**Project: 🚀 Deploying a Portfolio on AWS S3 using GitHub Actions 🚀**

Welcome to this detailed guide on how to deploy your Portfolio app on AWS S3 using GitHub Actions. In this step-by-step tutorial, we will cover each concept and explain every single step to ensure a smooth deployment process. But first, let's understand the key concepts of GitHub Actions.

**What is GitHub Actions?**

GitHub Actions is a powerful continuous integration and continuous deployment (CI/CD) platform provided by GitHub. It enables you to automate your software development workflows, allowing you to build, test, and deploy code directly from your GitHub repositories. With GitHub Actions, you can create custom workflows using YAML-based configuration files, defining the sequence of steps that need to be executed.

**Key Benefits of GitHub Actions:**

**Integrated with GitHub:** GitHub Actions is tightly integrated with GitHub repositories, making it seamless to use for developers who already host their code on GitHub.

**YAML Configuration:** Workflows are defined using simple YAML files, making it easy to understand and version-controlled along with your codebase.

**Event-Driven:** Workflows can be triggered based on various events, such as code pushes, pull requests, or other custom events in your repository.

**Extensibility:** A rich ecosystem of community-created actions is available, allowing you to extend the functionality of your workflows.

**Scalability:** GitHub provides scalable infrastructure to run your workflows, ensuring high availability and performance.

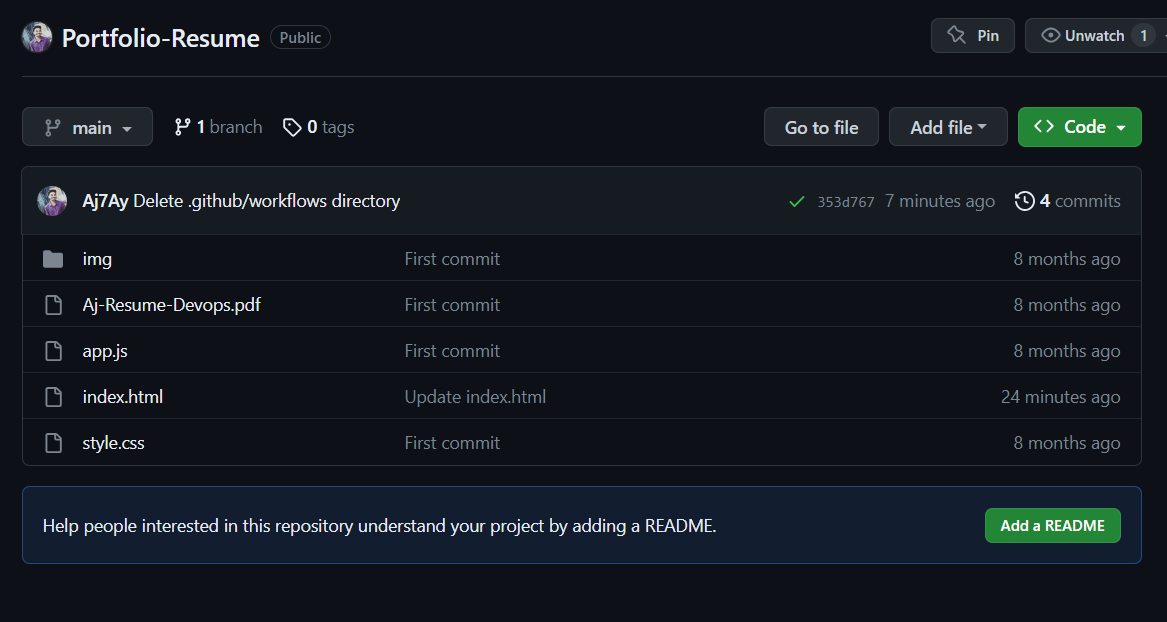
**Now, Let's Begin the Deployment Process! 🚀**

**Step 1: Create a New Repository - "my-portfolio" 📁**

To start the deployment process, create a new GitHub repository named "my-portfolio" where we will host our Portfolio app.

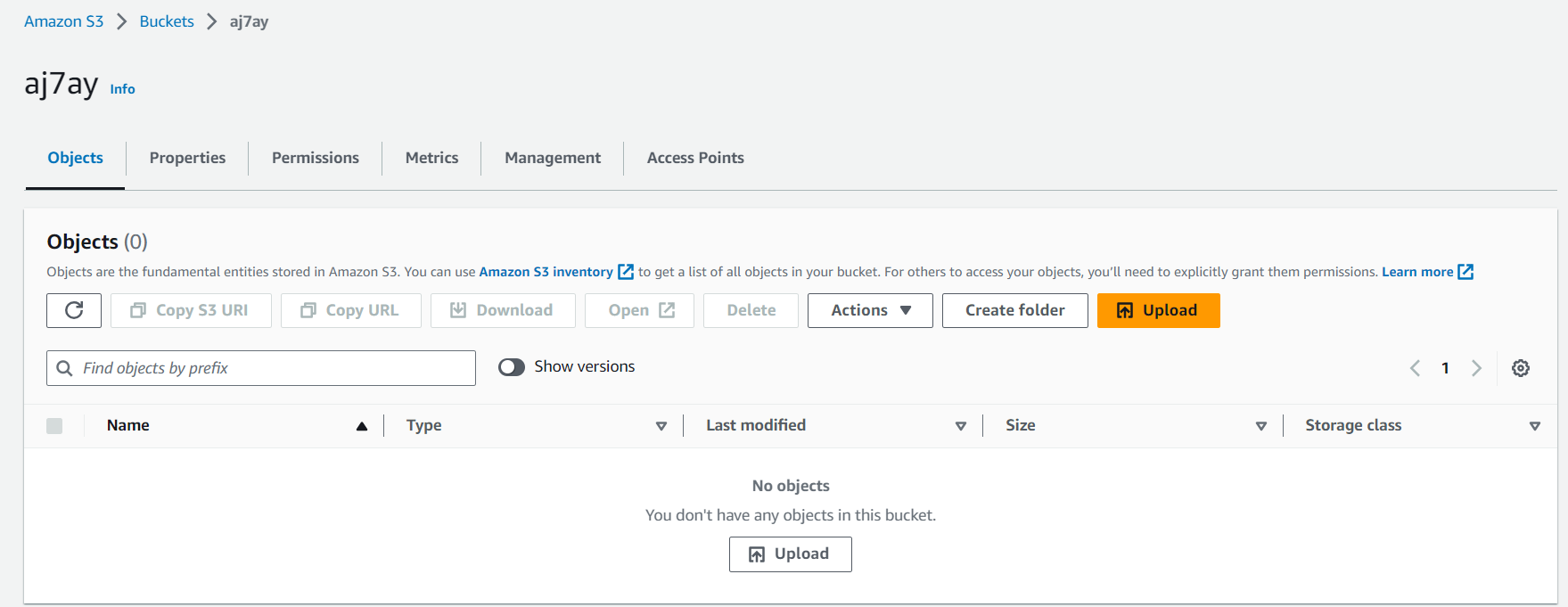
**Step 2: Add and Commit Code to GitHub Repository 📝**

Once the repository is created, add your Portfolio app's code to it. Use the git commands to commit the code and push it to your GitHub repository or use the github UI.



**Step 3: Create an S3 Bucket - "Aj7ay-portfolio-bucket" ☁️**

Now, head over to the AWS Management Console and create an S3 Bucket named "Aj7ay-portfolio-bucket" to store your Portfolio app's files. Remember to select a region for the bucket. Also allow public access.



**Step 4: Configure Bucket Policy for Public Access 🔒**

To allow public access to the objects in your S3 Bucket, you need to set up a bucket policy. Navigate to the bucket's properties, select "Permissions," and click on "Bucket Policy." Add the following policy:

{

"Version": "2012-10-17",

"Statement": [

{

"Sid": "PublicReadForGetBucketObjects",

"Effect": "Allow",

"Principal": "\*",

"Action": "s3:GetObject",

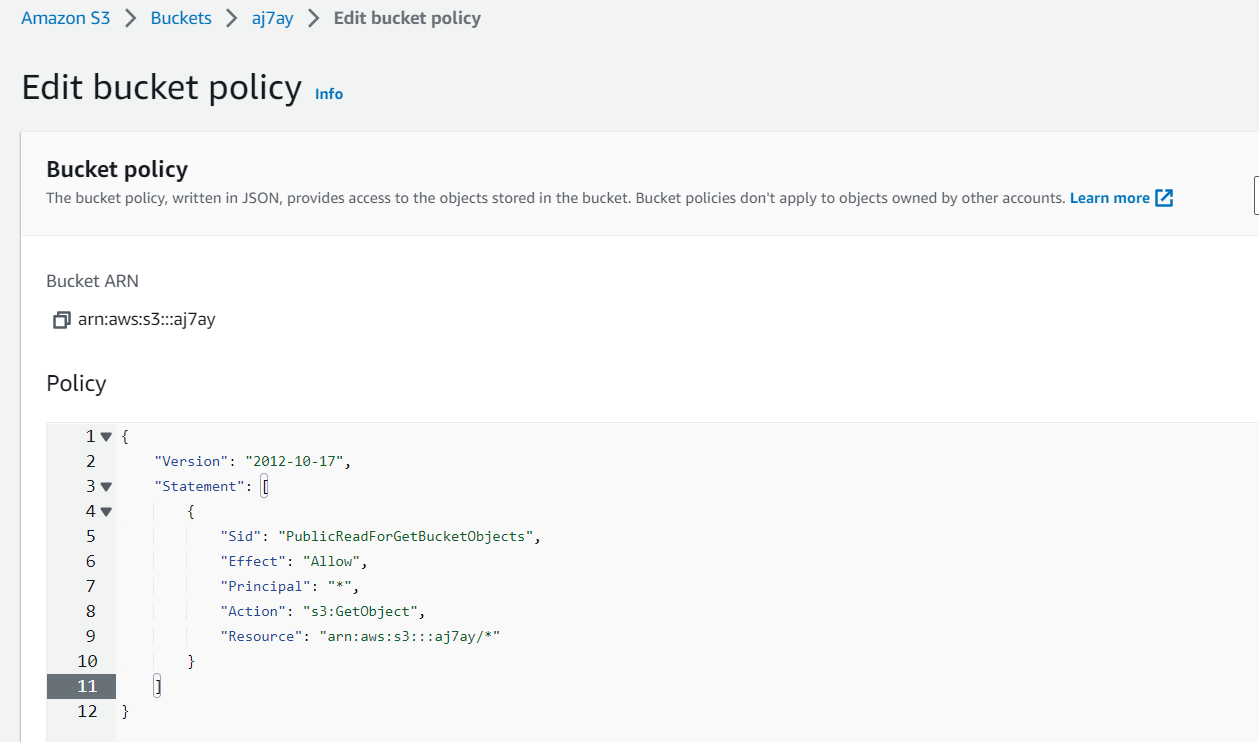
"Resource": "arn:aws:s3:::<bucket name>/\*"

}

]

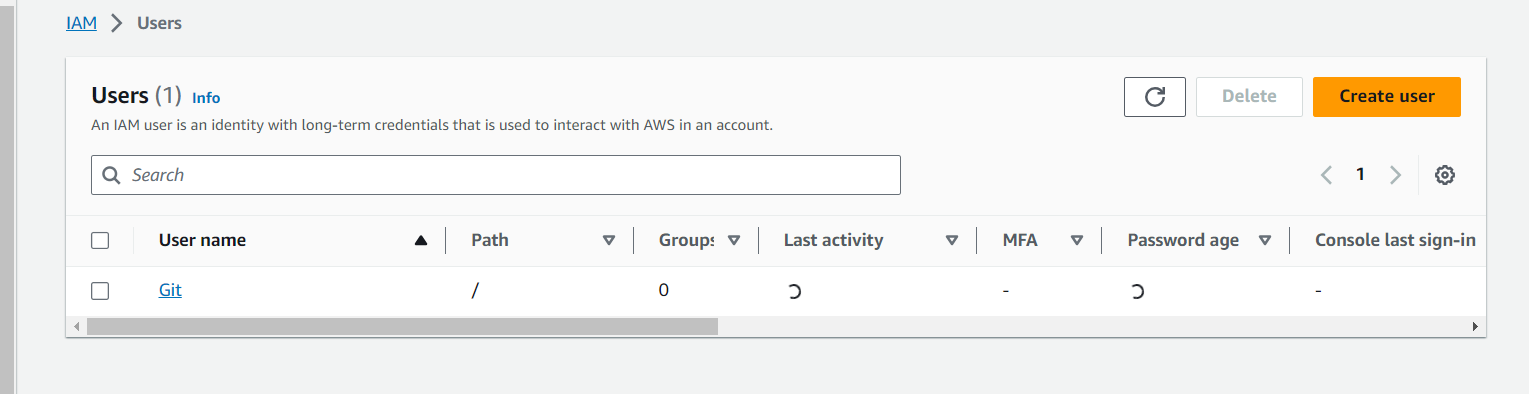
}

Remember to replace <bucket name> with the actual name of your S3 Bucket.

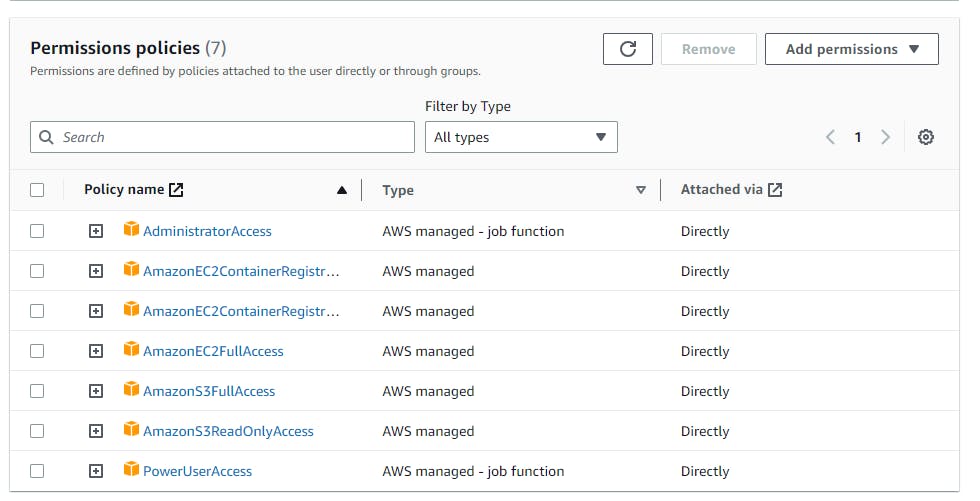


**Step 5: Create an IAM User and Generate Security Credentials 🔑**

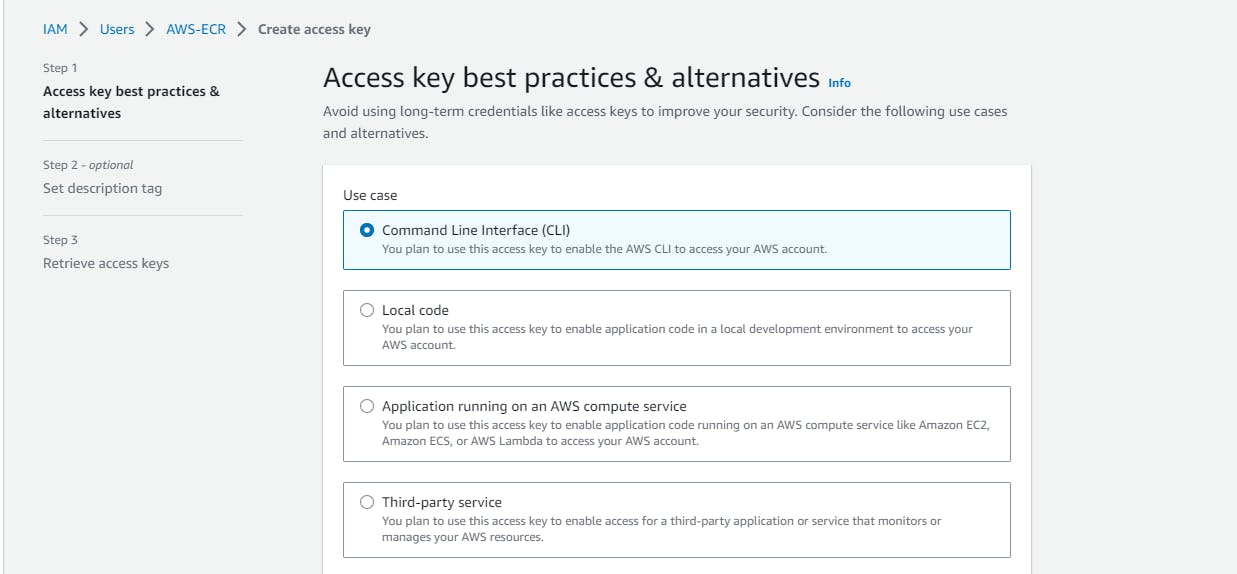
To securely interact with AWS services from GitHub Actions, we need an IAM User with sufficient permissions. If you already have an IAM User, you can use it; otherwise, create a new one.

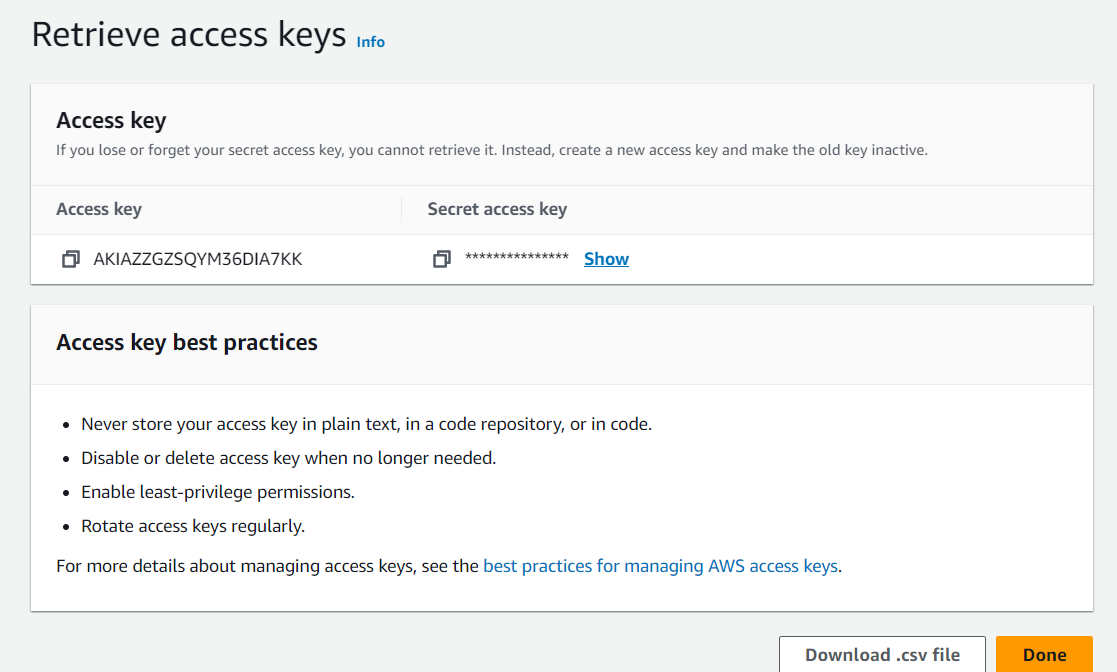


Assign the necessary IAM policies to this user, allowing access to S3 and any other services required for your deployment.



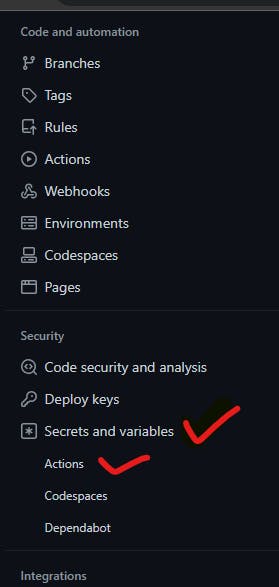
Generate the IAM User's access key and secret key, as we will need them later to configure GitHub Secrets.

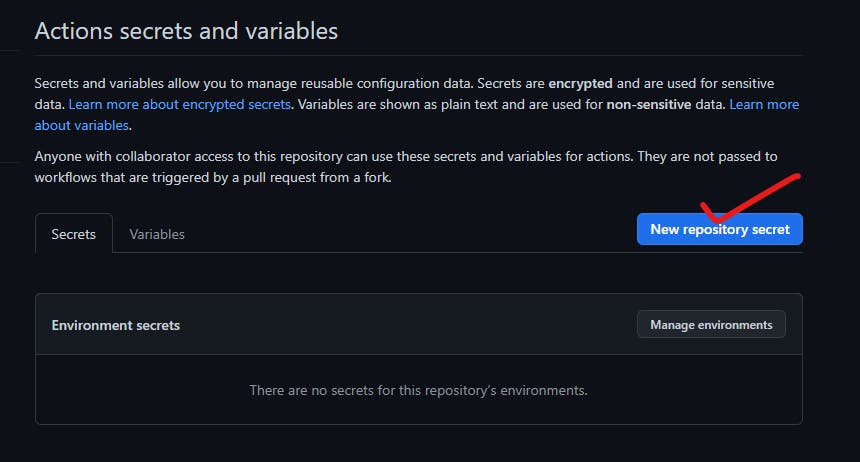


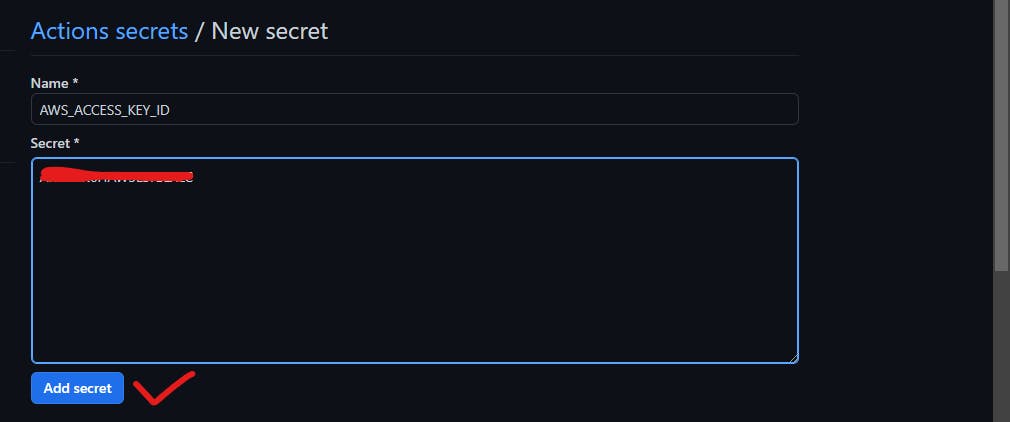


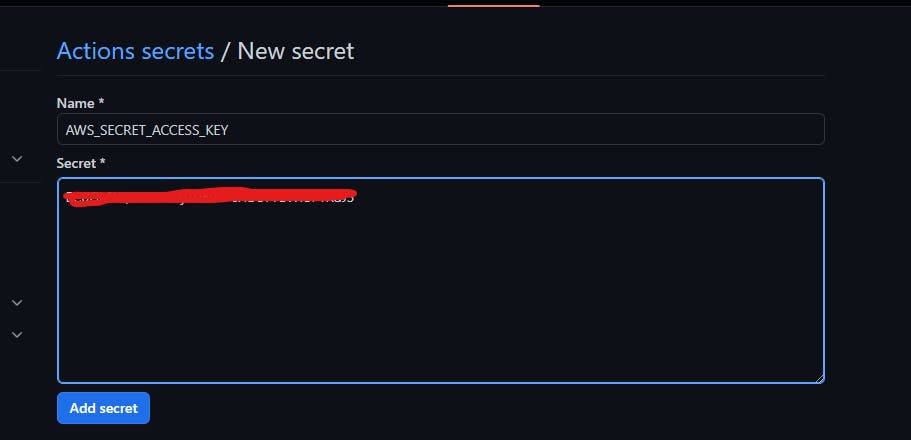
**Step 6: Set Up GitHub Secrets for AWS Credentials 🔒**

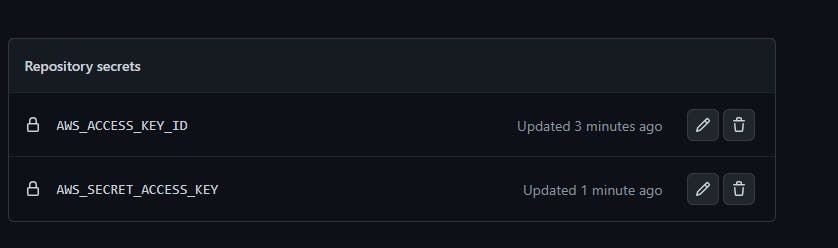
Go to your GitHub repository, click on "Settings," then "Secrets." Here, click on "New Repository Secret" to add your AWS access key and secret key as secrets named AWS\_ACCESS\_KEY\_ID and AWS\_SECRET\_ACCESS\_KEY, respectively.











**Step 7: Create the GitHub Actions Workflow 🔄**

In your GitHub repository, navigate to the "Actions" tab and click on "Set up a workflow yourself."

Create a new file named .github/workflows/main.yml, and add the following content:

name: my-portfolio-deployment # Name of the deployment

on:

push: # Trigger the workflow when changes are pushed

branches:

- main

jobs: # Any name can be provided

build-and-deploy:

runs-on: ubuntu-latest # Latest version of Ubuntu

steps:

- name: Checkout # Check out the repository's code into the workflow's execution environment.

uses: actions/checkout@v1 # Script actions

- 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

- name: Deploy static site to S3 bucket

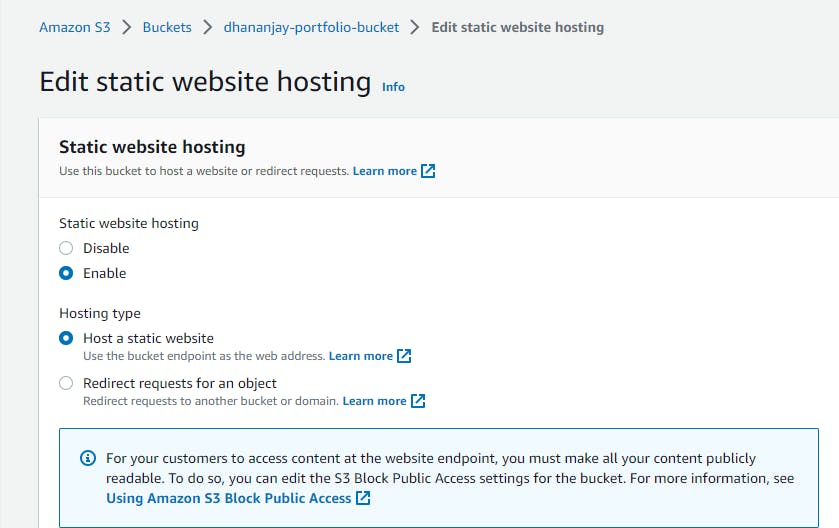
run: aws s3 sync . s3://<bucket-name> --delete # Change bucket name

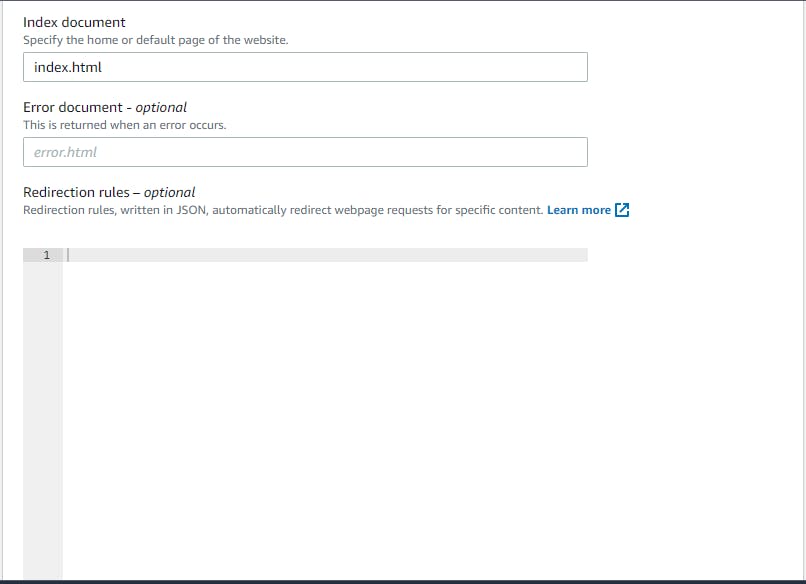
**Step 8: Commit Changes and Trigger Deployment 🚀**

Commit the .github/workflows/main.yml file to your repository and push the changes. This will trigger the GitHub Actions workflow to deploy your Portfolio app to the S3 Bucket.

**Step 9: Enable Static Website Hosting for the S3 Bucket 🏠**

Once the deployment is complete, go to your S3 Bucket's properties, select "Static website hosting," and enable it. Specify the "Index document" as index.html.





**Step 10: Access Your Portfolio Website 🌐**

Now, visit the URL endpoint provided by S3 Static Website Hosting, and you'll see your deployed Portfolio app live and accessible to the public!

Congratulations! You have successfully deployed your Portfolio app on AWS S3 using GitHub Actions.

{

    "Version": "2012-10-17",

    "Statement": [

        {

            "Sid": "PublicReadForGetBucketObjects",

            "Effect": "Allow",

            "Principal": "\*",

            "Action": "s3:GetObject",

            "Resource": "arn:aws:s3:::<bucket name>/\*"

        }

    ]

}

name: my-portfolio-deployment # Name of the deployment

on:

  push: # Trigger the workflow when changes are pushed

    branches:

      - main

jobs: # Any name can be provided

  build-and-deploy:

    runs-on: ubuntu-latest # Latest version of Ubuntu

    steps:

      - name: Checkout # Check out the repository's code into the workflow's execution environment.

        uses: actions/checkout@v1 # Script actions

      - 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

      - name: Deploy static site to S3 bucket

        run: aws s3 sync . s3://<bucket-name> --delete # Change bucket name