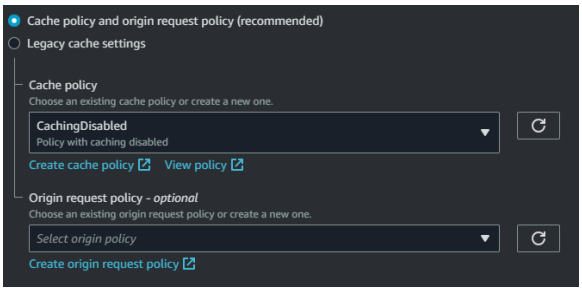
TEXAS A&M UNIVERSITY
Department of Computer Science & Engineering

Project 3 Phase 2 Documentation

Prerequisite

In CloudFront, navigate to the Distribution used for your S3 bucket.

Ensure that the Cache policy is “CachingDisabled.”

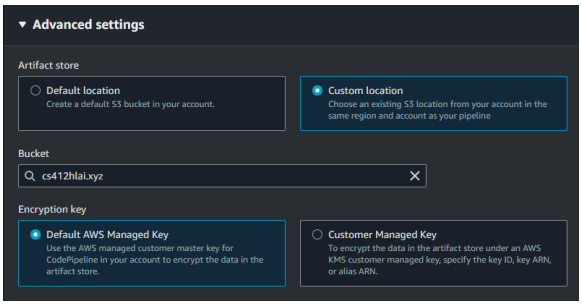


Creating a Code Pipeline

1. Navigate to AWS CodePipeline.

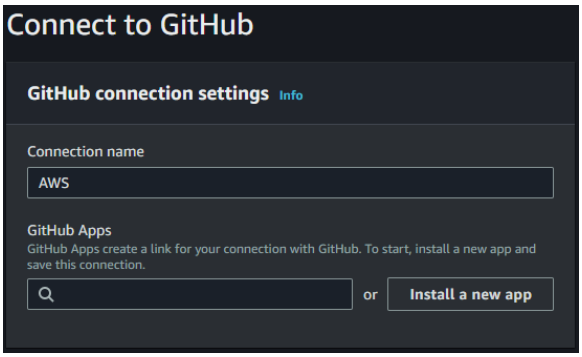
2. Click on the “Create pipeline” button.

3. Fill out the “Pipeline name”
4. Under “Advanced Settings” select “Custom location” from the “Artifact store” and fill out the bucket with the S3 bucket.

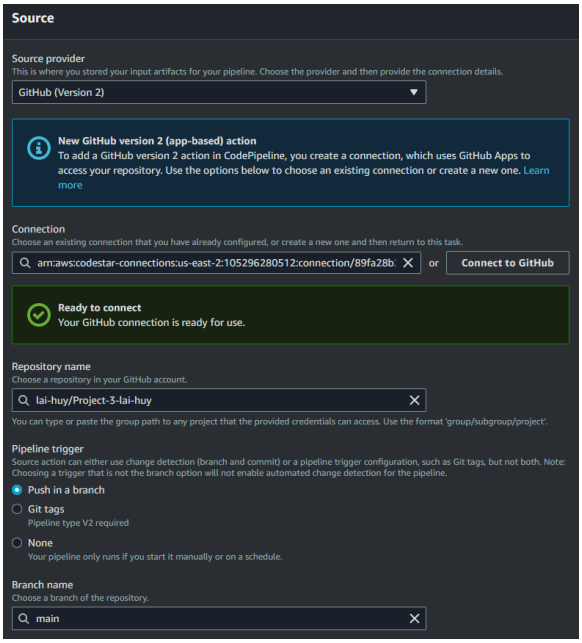


5. In the “Add source stage,” select “GitHub (Version 2)” from the “Source provider” dropdown menu and select your S3 bucket.
6. In the “Connection” category, select the “Connect to GitHub” button.
7. Give the connection name of your choosing and click the “Connect to GitHub” button.

8. Click on “Install a new app.”

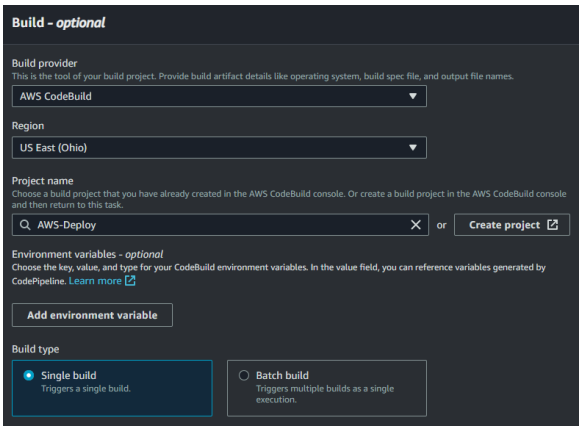


9. In the “Repository name,” select the appropriate GitHub repository you wish to push to the S3 bucket.
10. Select a branch you with to deploy from.
11. Leave the rest of the settings as is.

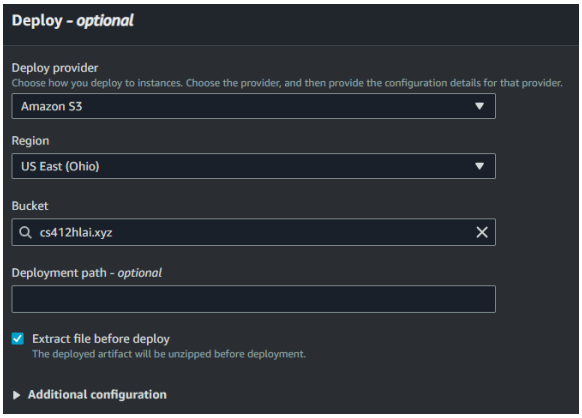


12. In “Add build stage” step, select “AWS CodeBuild” from the “Build provider” dropdown.

13. Select the “Create project” button and specify the parameters you want to.



14. In the “Deploy” step, select “Amazon S3” from the “Deploy provider”, select your appropriate S3 bucket, and check the “Extract file before deploy.”

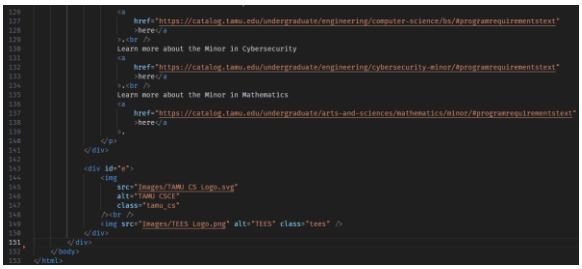


15. Click “Next” and “Create Pipeline”

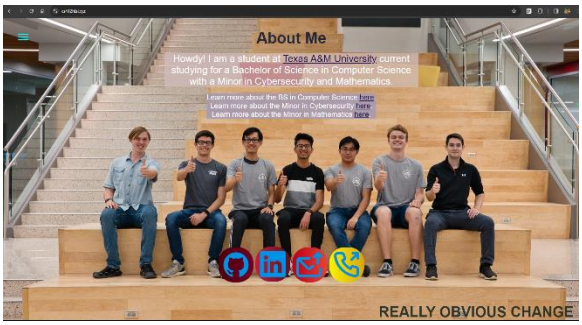
TEXAS A&M UNIVERSITY
Department of Computer Science & Engineering

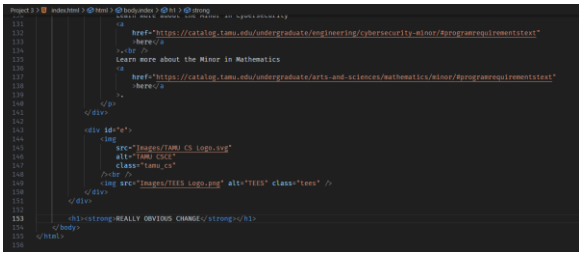
Workflow in Action

Pre-commit



Post-commit



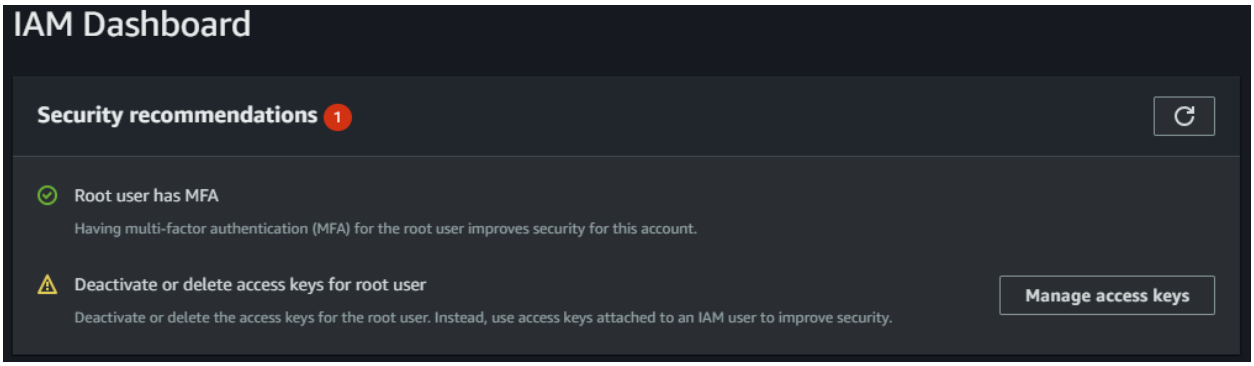


TEXAS A&M UNIVERSITY
Department of Computer Science & Engineering

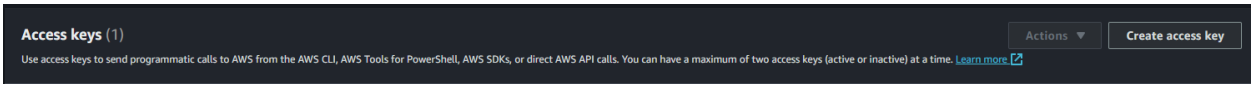
Connect a GitHub repository to AWS S3 Bucket

1. Create a GitHub repository.
    a. Add the contents of the static webpage to the repository and commit your changes.
2. Create AWS Access Keys
    a. Navigate to AWS Identity and Access Management (IAM)

    b. In the IAM Dashboard, click on “Manage access keys”.

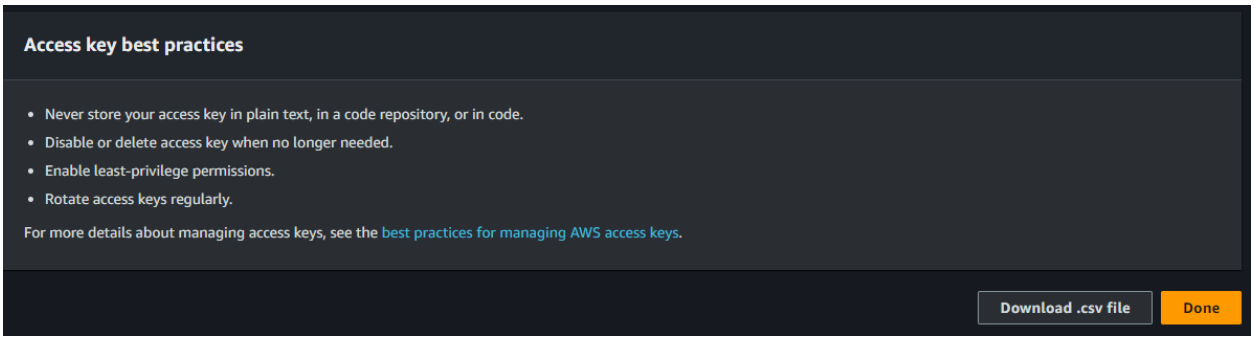


    c. Scroll down and click on “Create access key”.



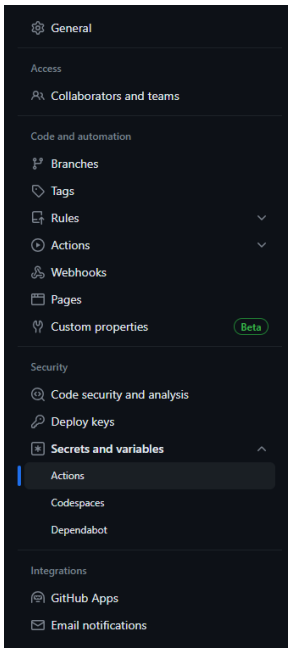
    The Security Manager will warn you about creating an access key to the root user. After carefully reading the warning, acknowledge that you read the warning.
    d. Click on “Create access key”.

e. Once you have created an Access Key, click the “Download .csv file” button.



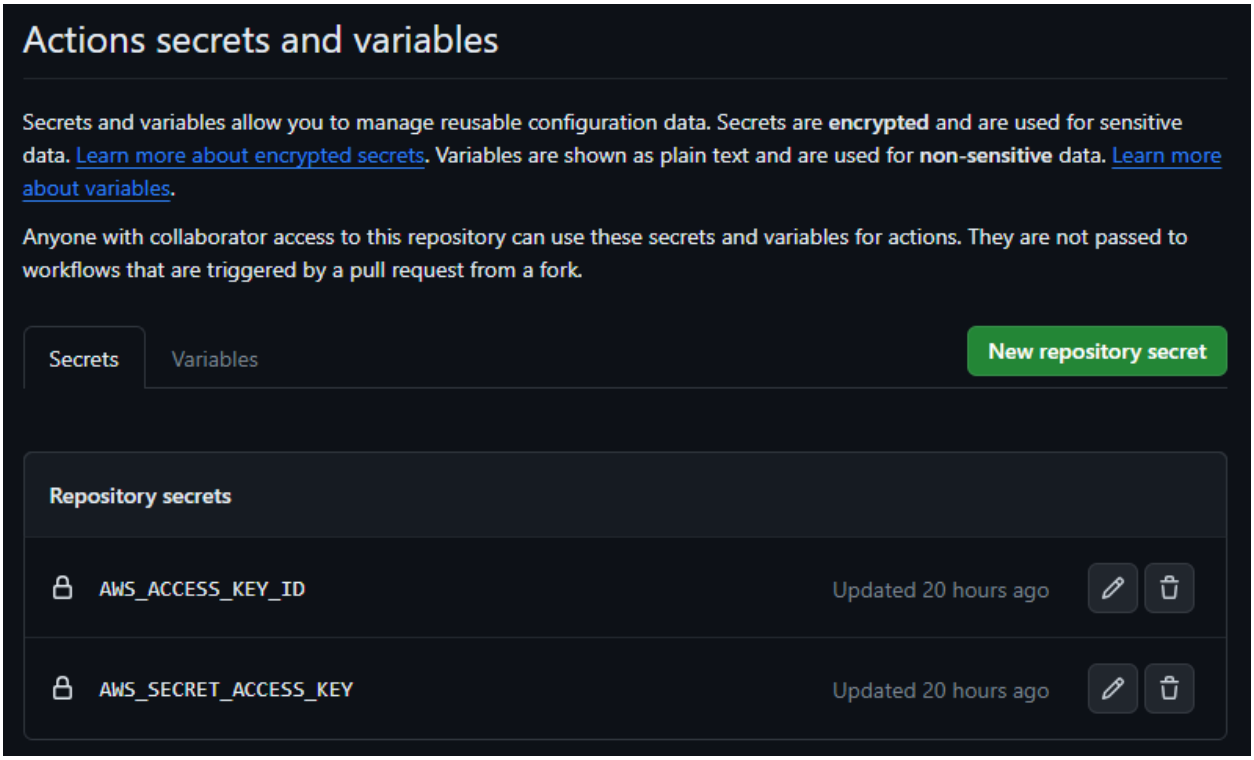
Save this file somewhere as we will require the credentials within the file.

3. Add AWS Keys to GitHub
   a. Navigate to your remote repository on GitHub.
   b. Go to the “Settings” tab on the right.
   c. Scroll down to the “Secrets and variables” option and click on it.



   d. Click on “Actions” in the submenu.

e. Click on the “New repository secret” button and add the AWS Credentials from the .csv file we downloaded earlier.



4. Add GitHub Workflow
    a. Put the following code in the file ".github/workflow/aws.yml"

```
name: Push to S3


on: [push]


jobs:
    deploy:
        runs-on: ubuntu-latest
        steps:
            - name: Checkout
              uses: actions/checkout@v2
            - name: Setup AWS CLI
              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: Sync files to S3 bucket
              run: |
                  aws s3 sync ./ s3://cs412hlai.xyz --delete
```