



Connecting GitHub account to AWS Developer Tools

GitHub is a widely used web-based platform that facilitates software development and version control using Git. It provides a collaborative environment where developers can work together on projects, manage code, and track changes. Here's a comprehensive overview:

Key Components and Features

1. Version Control with Git:

- GitHub is built on Git, a distributed version control system that tracks changes in source code. This allows multiple developers to work on a project simultaneously without conflicts.
- Git keeps a complete history of changes, enabling developers to revert to previous versions if necessary.

2. Repositories (Repos):

- A repository is a centralized location where all the files related to a project are stored. Each repository contains the project's source code, documentation, and other resources.
- Repositories can be public (accessible to everyone) or private (restricted access).

3. Branches:

- Branches are parallel versions of a repository. They allow developers to work on new features, bug fixes, or experiments without affecting the main codebase (often called the "main" or "master" branch).
- Once the work on a branch is complete, it can be merged back into the main branch.

4. Pull Requests (PRs):

- Pull requests are a key feature for collaboration. They are used to propose changes from one branch to another. The proposed changes can be reviewed, discussed, and approved before being merged into the main branch.
- Pull requests facilitate code review and help maintain code quality.

5. Issues and Bug Tracking:

- GitHub provides an integrated issue tracker to manage bugs, tasks, and feature requests. Users can create, comment on, and track the status of issues.
- This feature helps teams organize and prioritize work.

6. Collaborative Features:

- GitHub supports collaboration through features like forking (creating a personal copy of someone else's repository), code review, and discussions.
- Team members can comment on code, discuss changes, and provide feedback directly within the platform.

7. GitHub Pages:

- GitHub Pages is a service that allows users to host static websites directly from a GitHub repository. It's commonly used for project documentation, personal portfolios, and more.

8. Continuous Integration/Continuous Deployment (CI/CD):

- GitHub integrates with various CI/CD tools, enabling automated testing, building, and deployment of projects. This helps ensure code quality and accelerates the development process.

9. Security and Insights:

- GitHub offers security features like vulnerability alerts, secret scanning, and dependency management.
- Insights and analytics tools provide metrics on project activity, code changes, and collaboration.

10. Community and Open Source:

- GitHub is a central hub for open-source projects. Developers can contribute to projects, collaborate with others, and discover new projects.
- The platform hosts millions of repositories, making it a valuable resource for learning and sharing code.

Use Cases

- **Open-Source Development:** GitHub is a go-to platform for open-source projects, where contributors from around the world can collaborate.
- **Team Collaboration:** Organizations and teams use GitHub to manage internal projects, track issues, and maintain a streamlined workflow.
- **Documentation and Knowledge Sharing:** GitHub Pages and wikis are used for project documentation and knowledge sharing.

Access

GitHub is accessible through its website (github.com) and offers both free and paid plans. The free tier provides ample functionality for most users, while paid plans offer additional features like private repositories and advanced security options.

In summary, GitHub is a powerful platform for version control, collaboration, and project management, making it a cornerstone of modern software development practices.

End Goal: In this lab, you will learn how to connect a GitHub repository with AWS to automate the deployment of a website. The end goal is to establish a connection between GitHub and the AWS Developer tool and create a code pipeline so that whenever you update your website's code on GitHub, the changes are automatically applied to your live website without you having to do anything manually. This setup saves time and ensures your site always has the latest updates.

😊 To begin with the Lab:

😊 Step 1: Creating GitHub Repo

1. In this lab we will learn how to connect our GitHub account with AWS Developer Tools. So, for that first, we need to create a private repository in our GitHub account.

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Required fields are marked with an asterisk ().*

Owner * Repository name *

 Pulkit-Kumar-0 / demo-website-repo
✓ demo-website-repo is available.

Great repository names are short and memorable. Need inspiration? How about [special-train](#) ?

Description (optional)

just a demo repository

 Public

Anyone on the internet can see this repository. You choose who can commit.

 Private

You choose who can see and commit to this repository.

Initialize this repository with:

Add a README file

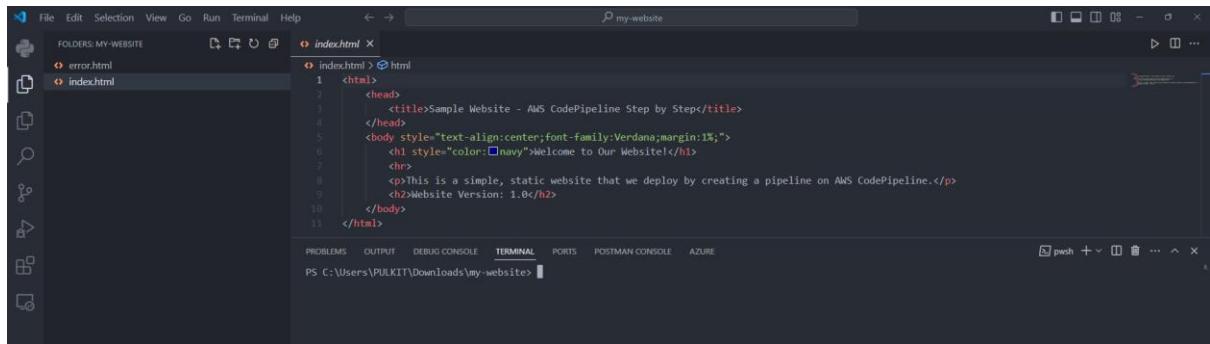
This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore

.gitignore template: None ▾

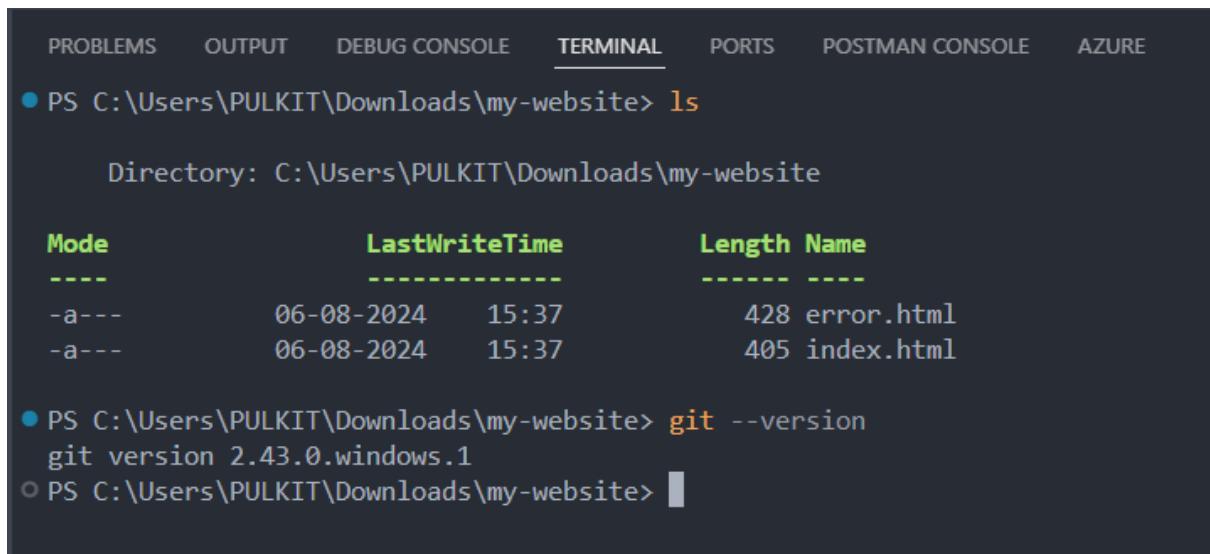
Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

2. Now you will find a zipped folder on our GitHub account with the name my-website. You need to download it and then unzip the content then open the folder in VS Code. Also, open the terminal, as you can see below in the snapshot.



```
index.html
1 <html>
2   <head>
3     <title>Sample Website - AWS CodePipeline Step by Step</title>
4   </head>
5   <body style="text-align:center;font-family:Verdana;margin:1%;">
6     <h1 style="color: navy">Welcome to Our Website!</h1>
7     <br>
8     <p>This is a simple, static website that we deploy by creating a pipeline on AWS CodePipeline.</p>
9     <h2>Website Version: 1.0</h2>
10    </body>
11  </html>
```

3. You need to check only these two files should be in your my-website folder and git should be installed on your local machine.



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS POSTMAN CONSOLE AZURE
● PS C:\Users\PULKIT\Downloads\my-website> ls
Directory: C:\Users\PULKIT\Downloads\my-website

Mode                LastWriteTime         Length Name
----                -----          428 error.html
-a---        06-08-2024 15:37           405 index.html

● PS C:\Users\PULKIT\Downloads\my-website> git --version
git version 2.43.0.windows.1
○ PS C:\Users\PULKIT\Downloads\my-website> █
```

4. To begin with, you must initialize a local Git repository under this folder. It's straightforward. You can do this using the 'git init' command. After that, you can check the status of your local Git repository using the 'git status' command.
5. As you can see, none of our files have been tracked by Git. It means they won't be included in our commits. So next, we will stage our files using the 'git add' command, and we will provide '.' to select all files under this folder.

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS POSTMAN CONSOLE AZURE

● PS C:\Users\PULKIT\Downloads\my-website> git init
Initialized empty Git repository in C:/Users/PULKIT/Downloads/my-website/.git/
● PS C:\Users\PULKIT\Downloads\my-website> git status
On branch master

No commits yet

Untracked files:
(use "git add <file>..." to include in what will be committed)
  error.html
  index.html

nothing added to commit but untracked files present (use "git add" to track)
● PS C:\Users\PULKIT\Downloads\my-website> git add .
warning: in the working copy of 'error.html', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'index.html', LF will be replaced by CRLF the next time Git touches it
● PS C:\Users\PULKIT\Downloads\my-website> git status
On branch master

No commits yet

Changes to be committed:
(use "git rm --cached <file>..." to unstage)
  new file:   error.html
  new file:   index.html

○ PS C:\Users\PULKIT\Downloads\my-website>
```

- After providing the commit command you can see that our first git commit has been created.

```
● PS C:\Users\PULKIT\Downloads\my-website> git commit -m "Version 1.0"
[master (root-commit) ee079cc] Version 1.0
 2 files changed, 22 insertions(+)
  create mode 100644 error.html
  create mode 100644 index.html
○ PS C:\Users\PULKIT\Downloads\my-website>
```

- However, it currently only exists in our local Git repository. We must push it to our GitHub repository to be able to create a pipeline from it later. First, let's add our GitHub repository as a remote repository using the 'git remote add' command.
- For that, first we need to copy the HTTPS URL from GitHub. You can get it from the quick setup as shown below.

Quick setup — if you've done this kind of thing before

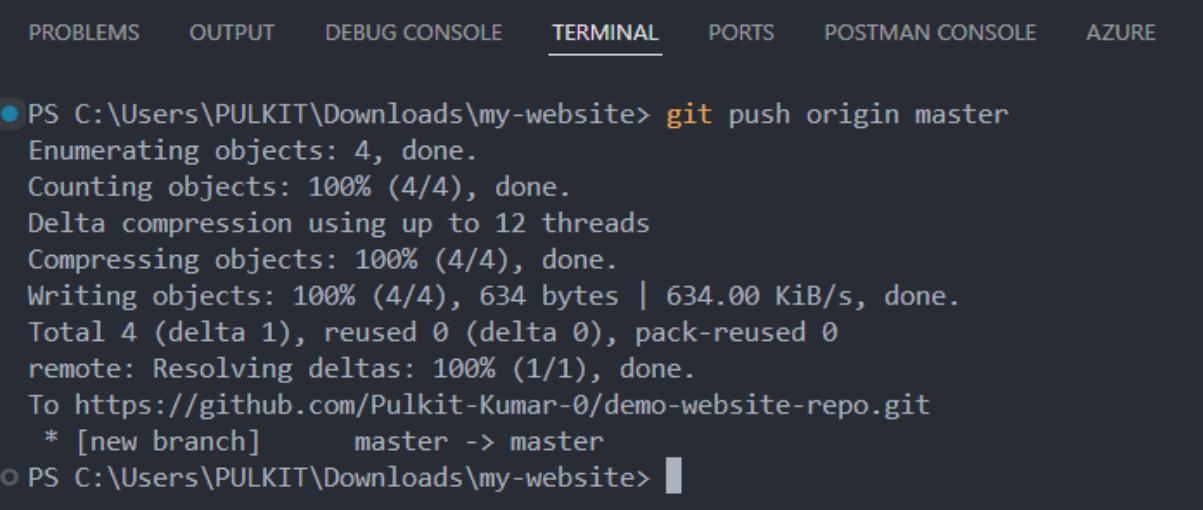
 Set up in Desktop or [HTTPS](#) [SSH](#) <https://github.com/Pulkit-Kumar-0/demo-website-repo.git>

Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

```
● PS C:\Users\PULKIT\Downloads\my-website> git remote add origin https://github.com/Pulkit-Kumar-0/demo-website-repo.git
● PS C:\Users\PULKIT\Downloads\my-website> git remote -v
origin https://github.com/Pulkit-Kumar-0/demo-website-repo.git (fetch)
origin https://github.com/Pulkit-Kumar-0/demo-website-repo.git (push)
○ PS C:\Users\PULKIT\Downloads\my-website>
```

- Now we have given the push command to push all our objects to the GitHub repository. Also if you are doing all this for the first time then VS Code will ask you to login first.

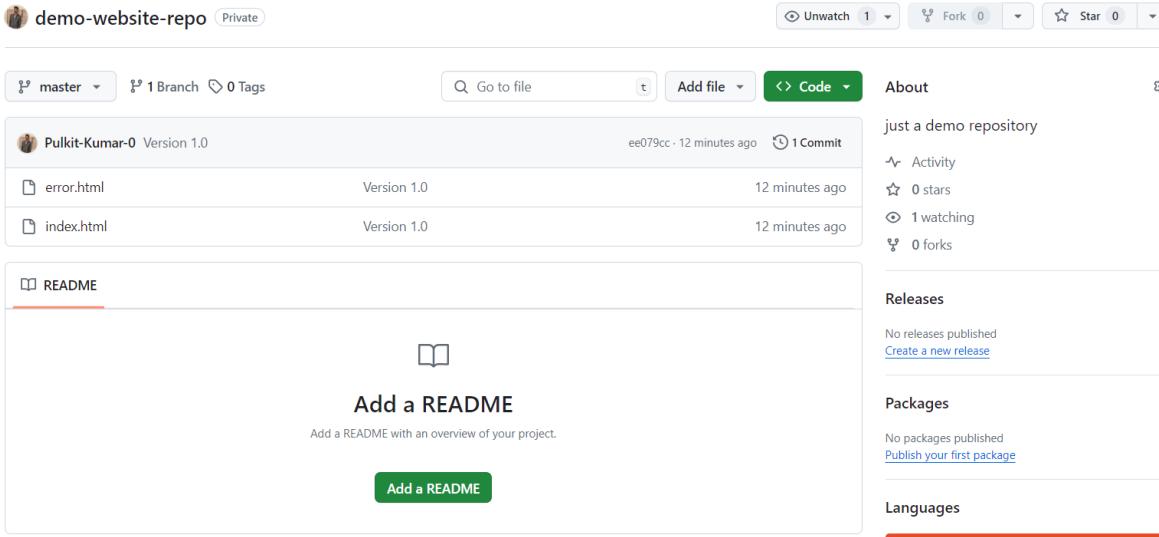
git push origin master



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS POSTMAN CONSOLE AZURE

● PS C:\Users\PULKIT\Downloads\my-website> git push origin master
  Enumerating objects: 4, done.
  Counting objects: 100% (4/4), done.
  Delta compression using up to 12 threads
  Compressing objects: 100% (4/4), done.
  Writing objects: 100% (4/4), 634 bytes | 634.00 KiB/s, done.
  Total 4 (delta 1), reused 0 (delta 0), pack-reused 0
  remote: Resolving deltas: 100% (1/1), done.
  To https://github.com/Pulkit-Kumar-0/demo-website-repo.git
    * [new branch]      master -> master
○ PS C:\Users\PULKIT\Downloads\my-website>
```

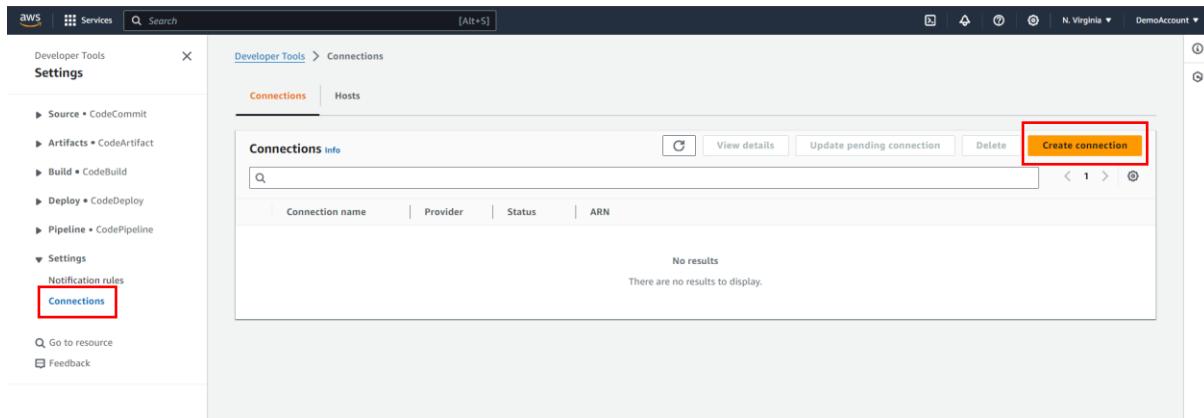
- Now go to your Repository and click on refresh, you will see your resources there.



The screenshot shows a GitHub repository page for 'demo-website-repo'. The repository is private. It contains one branch ('master'), one file ('error.html'), and one folder ('index.html'). The 'About' section describes it as 'just a demo repository'. The 'Activity' section shows 1 commit by 'Pulkit-Kumar-0' from 12 minutes ago. The 'Releases' section indicates 'No releases published' and 'Create a new release'. The 'Packages' section shows 'No packages published' and 'Publish your first package'. The 'Languages' section shows 'HTML 100.0%'.

😊 Step 2: Establishing connection between GitHub and AWS

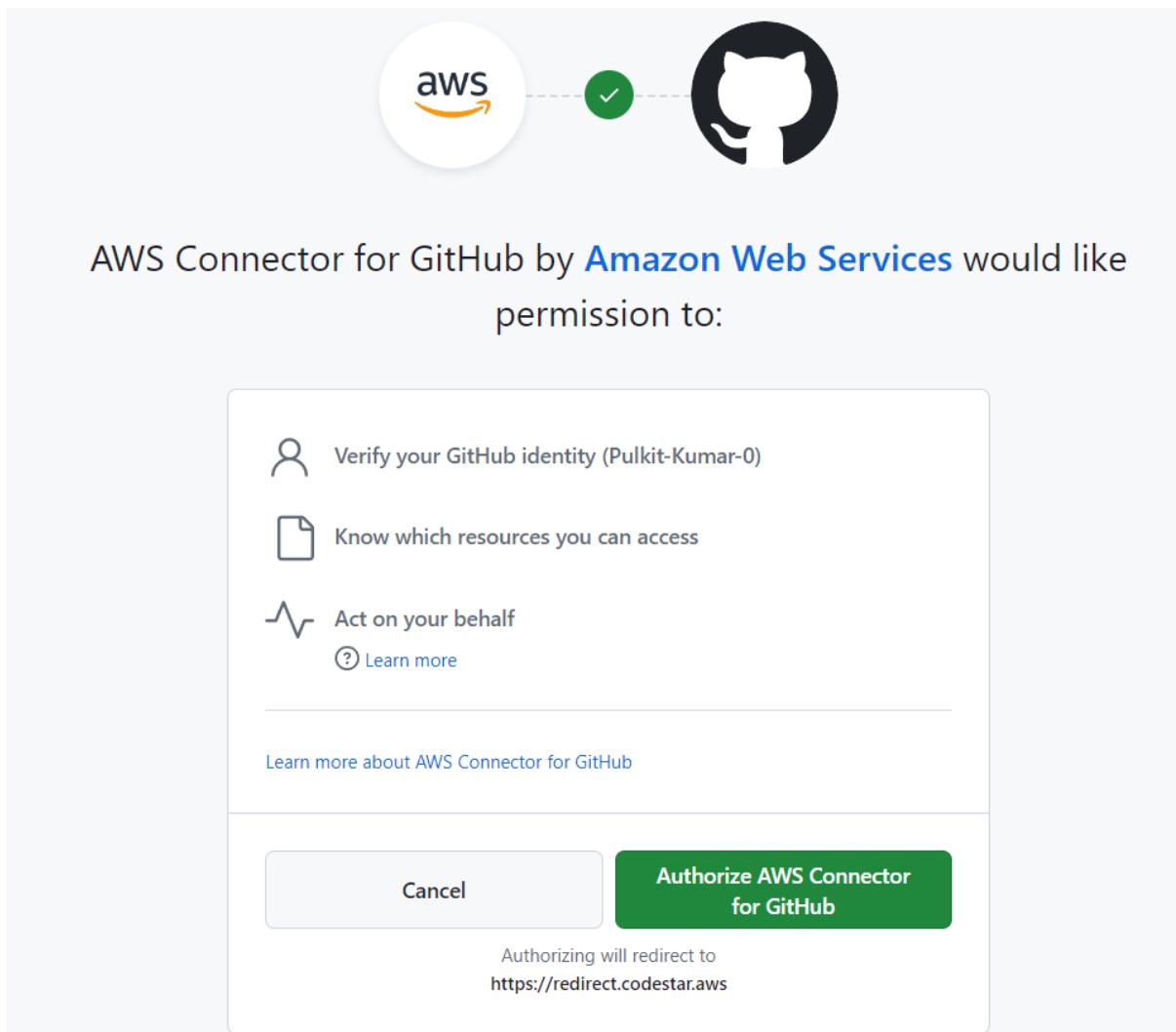
- Now, before creating a pipeline for Git repositories in any third-party provider, such as GitHub, you must first configure a connection between your AWS account and that provider. So, next, let's go back to the CodePipeline Console to configure the GitHub connection.
- So, in AWS Developer Tools expand the settings option and choose Connection. Now you need to create a Connection.



3. Then you need to choose GitHub and give your connection a name, now just click on Connect to GitHub.

A screenshot of the 'Create a connection' form for GitHub. The 'Provider' section shows 'GitHub' selected. The 'Connection name' field contains 'GitHub Connection'. At the bottom right, there's a 'Connect to GitHub' button.

4. As you see, it redirected us to GitHub. This process configures the AWS Connector for the GitHub app. First, you must authorize the AWS connector to access your GitHub account.



5. Next, you must install the AWS Connector into the GitHub account you will use with your pipelines. If you installed it before, you can choose it from the list here. Otherwise, you can install the app by clicking the button next to it. And, it redirected us to GitHub again.

Beginning July 1, 2024, the console will create connections with codeconnections in the resource ARN. Resources with both service prefixes will continue to display in the console. [Learn more](#) 



Connect to GitHub

GitHub connection settings Info

Connection name

GitHub Apps

GitHub Apps create a link for your connection with GitHub. Install a new app and save this connection.



or

Install a new app

► Tags - optional

Connect

6. Now click on install. After that, it will ask you to enter your GitHub account password.



Install AWS Connector for GitHub

Install on your personal account Pulkit Kumar



for these repositories:

All repositories

This applies to all current *and* future repositories owned by the resource owner.
Also includes public repositories (read-only).

Only select repositories

Select at least one repository.
Also includes public repositories (read-only).

with these permissions:

- Read access to issues and metadata**
- Read and write access to administration, code, commit statuses, pull requests, and repository hooks**

Install

Cancel

Next: you'll be directed to the GitHub App's site to complete setup.

7. The configuration on the GitHub side has been finished. And we are now back to the AWS Developer Tools connections. As you can see, we have a new ID in the 'GitHub Apps' field.

Beginning July 1, 2024, the console will create connections with codeconnections in the resource ARN. Resources with both service prefixes will continue to display in the console. [Learn more](#)

Connect to GitHub

GitHub connection settings [Info](#)

Connection name

GitHub Apps

GitHub Apps create a link for your connection with GitHub. Install a new app and save this connection.



or

[Install a new app](#)

► Tags - optional

[Connect](#)

- Below you can see that our connection was successful. Now, we can use this connection to create a pipeline from the GitHub repositories it has access to.

Connection GitHub Connection created successfully

Developer Tools > Connections > ec300734-7071-436f-a6e7-1fdc389af15a

GitHub Connection

| Connection settings | | | |
|---------------------|----------|--|--|
| Name | Provider | Status | ARN |
| GitHub Connection | GitHub | Available | arn:aws:codeconnections:us-east-1:533267094905:connection/ec300734-7071-436f-a6e7-1fdc389af15a |

- Below you can see your connection and the provider is GitHub.

Developer Tools > Connections

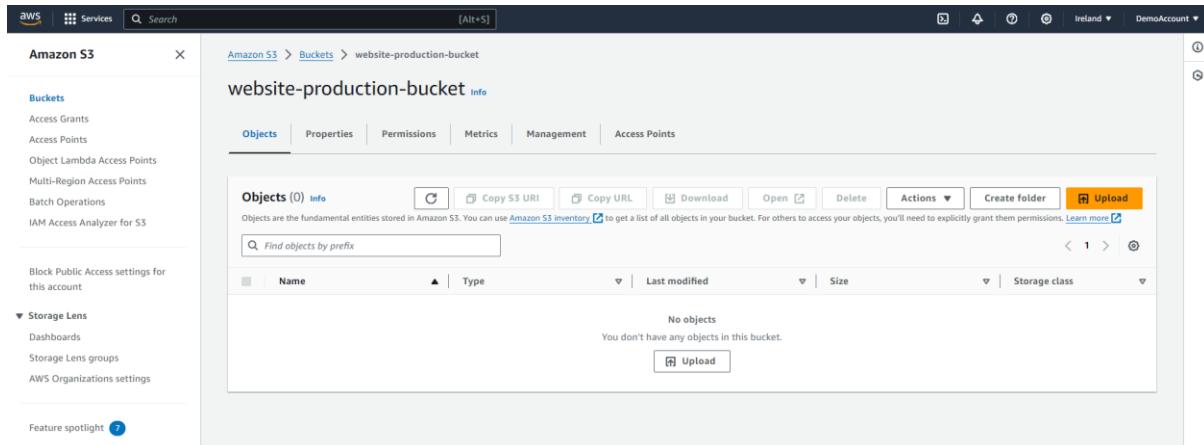
Connections [Hosts](#)

| Connections Info | | | |
|--|----------|--|--|
| Connection name | Provider | Status | ARN |
| <input checked="" type="radio"/> GitHub Connection | GitHub | Available | arn:aws:codeconnections:us-east-1:533267094905:connection/ec300734-7071-436f-a6e7-1fdc389af15a |

😊 Step 3: Creating Pipeline

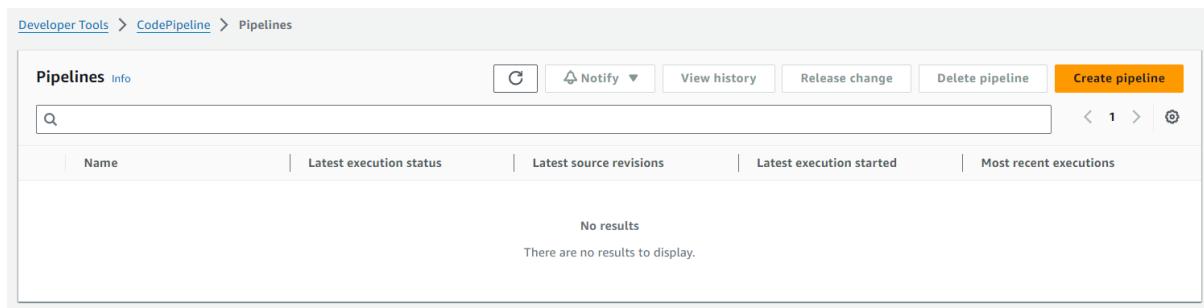
- Now we are going to use the same public bucket that we had used earlier and clear its content. If you don't have a bucket, then you need to create a bucket in the same region

where you will create your pipeline. Make your bucket public update the bucket policy and then enable static website hosting.



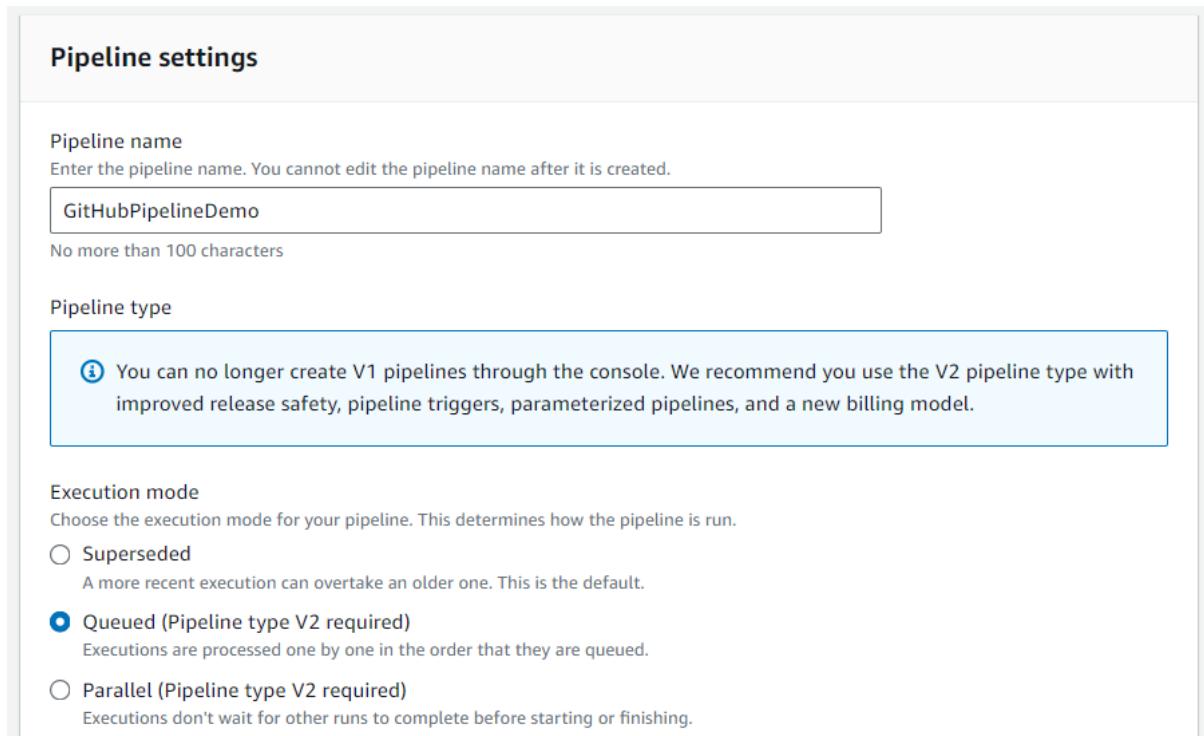
The screenshot shows the AWS S3 console with the path 'Amazon S3 > Buckets > website-production-bucket'. The left sidebar includes options like 'Buckets', 'Access Grants', 'Access Points', 'Object Lambda Access Points', 'Multi-Region Access Points', 'Batch Operations', 'IAM Access Analyzer for S3', 'Block Public Access settings for this account', 'Storage Lens', 'Dashboards', 'Storage Lens groups', and 'AWS Organizations settings'. The main content area shows the 'website-production-bucket' info page with tabs for 'Objects (0) Info', 'Properties', 'Permissions', 'Metrics', 'Management', and 'Access Points'. A prominent 'Upload' button is visible at the top right of the object list area.

2. So, now go to pipelines and click on Create Pipeline.



The screenshot shows the AWS CodePipeline console with the path 'Developer Tools > CodePipeline > Pipelines'. The left sidebar lists 'Pipelines' and 'Info'. The main content area shows the 'Pipelines' list page with columns for 'Name', 'Latest execution status', 'Latest source revisions', 'Latest execution started', and 'Most recent executions'. A message 'No results' and 'There are no results to display.' is displayed. A 'Create pipeline' button is highlighted at the top right.

3. Now give your pipeline a name and then keep the rest of the settings to default.



The screenshot shows the 'Pipeline settings' configuration page. The 'Pipeline name' field is set to 'GitHubPipelineDemo'. The 'Pipeline type' section contains a note: '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.' The 'Execution mode' section shows 'Queued (Pipeline type V2 required)' selected. Other options include 'Superseded' and 'Parallel (Pipeline type V2 required)'. A note for 'Parallel' says 'Executions don't wait for other runs to complete before starting or finishing.'

4. Then it will create a service role also. Now move to the next step.

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

AWSCodePipelineServiceRole-us-east-1-GitHubPipelineDemo

Type your service role name

- Allow AWS CodePipeline to create a service role so it can be used with this new pipeline

5. In the next step we need to choose our source provider which is GitHub, so choose GitHub (Version 2) and then choose your connection as we have created it earlier.

Source

Source provider

This is where you stored your input artifacts for your pipeline. Choose the provider and then provide the connection details.

GitHub (Version 2)



New GitHub version 2 (app-based) action

To add a GitHub version 2 action in CodePipeline, you create a connection, which uses GitHub Apps to access your repository. Use the options below to choose an existing connection or create a new one. [Learn more](#)

Connection

Choose an existing connection that you have already configured, or create a new one and then return to this task.



arn:aws:codeconnections:us-east-1:533267094905:connection/ec300734-70



or

[Connect to GitHub](#)



Ready to connect

Your GitHub connection is ready for use.

6. After that you need to choose your repository and the default branch. Then in the trigger choose No filter.

Repository name
Choose a repository in your GitHub account.

X

You can type or paste the group path to any project that the provided credentials can access. Use the format 'group/subgroup/project'.

Default branch
Default branch will be used only when pipeline execution starts from a different source or manually started.

X

Output artifact format
Choose the output artifact format.

CodePipeline default
AWS CodePipeline uses the default zip format for artifacts in the pipeline. Does not include Git metadata about the repository.

Full clone
AWS CodePipeline passes metadata about the repository that allows subsequent actions to do a full Git clone. Only supported for AWS CodeBuild actions.

Trigger

Trigger type
Choose the trigger type that starts your pipeline.

No filter
Starts your pipeline on any push and clones the HEAD.

Specify filter
Starts your pipeline on a specific filter and clones the exact commit. Pipeline type V2 is required.

Do not detect changes
Don't automatically trigger the pipeline.

i You can add additional sources and triggers by editing the pipeline after it is created.

7. Now we will skip the build stage.

Developer Tools > CodePipeline > Pipelines > Create new pipeline

Step 1 Choose pipeline settings Step 3 of 5

Step 2 Add source stage

Step 3 Add build stage

Step 4 Add deploy stage

Step 5 Review

Add build stage Info

Build - optional

Build provider
This is the tool of your build project. Provide build artifact details like operating system, build spec file, and output file names.

▼

Cancel Previous Skip build stage Next

8. In the last step we need to choose where to deploy our content, so choose S3 here and then choose your production bucket. Then move to the review page and create your pipeline.

Deploy

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

Amazon S3

Region
Europe (Ireland)

Bucket
website-production-bucket

Deployment path - *optional*

Extract file before deploy
The deployed artifact will be unzipped before deployment.

► Additional configuration

Configure automatic rollback on stage failure

Cancel Previous Next

9. Below you can see that our pipeline has been created and deployed successfully.

GitHubPipelineDemo

Pipeline type: V2 Execution mode: QUEUED

Source Succeeded
Pipeline execution ID: 1096bc73-a857-461e-b32e-9de449891d96

Source
Github (Version 2) [View details](#)
Succeeded - Just now
[ee079cc7](#)

ee079cc7 Source: Version 1.0

Disable transition

Deploy Succeeded
Pipeline execution ID: 1096bc73-a857-461e-b32e-9de449891d96

Deploy
Amazon S3 [View details](#)
Succeeded - Just now
[ee079cc7](#) Source: Version 1.0

Start rollback

10. Also, if you click on view details from the source then you can see the details regarding your source. Also if you click on view in Code Star Source Connection, then you will be redirected to the commit on your GitHub repository that triggered this action.

Action execution details

Action name: Source Status: Succeeded

Summary **Input** **Output**

| | |
|--|---------------|
| Status | Last updated |
| ✓ Succeeded | 3 minutes ago |
| Action execution ID | |
| b8686cad-b9b5-48c2-b9bb-0bee59047cc2 | |
| Message | |
| { "ProviderType": "GitHub", "CommitMessage": "Version 1.0" } | |
| Execution details | |
| View in CodeStarSourceConnection | |

Done

```

11 └── error.html
...
... @@ -0,0 +1,11 @@
1 + <html>
2 +   <head>
3 +     <title>Error! - AWS CodePipeline Step by Step</title>
4 +   </head>
5 +   <body style="text-align:center;font-family:Verdana;margin:1%;">
6 +     <h1 style="color:red">Sorry! We have an error on our website.</h1>
7 +     <h2>Please come back later.</h2>
8 +     <hr>
9 +     <p>This is a simple error page for the static website we deploy by creating a pipeline on AWS CodePipeline.</p>
10 +    </body>
11 +   </html>

```



```

11 └── index.html
...
... @@ -0,0 +1,11 @@
1 + <html>
2 +   <head>
3 +     <title>Sample Website - AWS CodePipeline Step by Step</title>
4 +   </head>
5 +   <body style="text-align:center;font-family:Verdana;margin:1%;">
6 +     <h1 style="color:navy">Welcome to Our Website!</h1>
7 +     <hr>
8 +     <p>This is a simple, static website that we deploy by creating a pipeline on AWS CodePipeline.</p>
9 +     <h2>Website Version: 1.0</h2>
10 +    </body>
11 +   </html>

```

11. Then go to S3 and open your bucket and go to properties, scroll down to bottom click on the static website URL provided and you can see your website.

Welcome to Our Website!

This is a simple, static website that we deploy by creating a pipeline on AWS CodePipeline.

Website Version: 1.0

12. In the objects section you can see your files.

Amazon S3 > Buckets > website-production-bucket

website-production-bucket [info](#)

Objects (2) [Info](#)

| Name | Type | Last modified | Size | Storage class |
|----------------------------|------|--------------------------------------|---------|---------------|
| error.html | html | August 6, 2024, 16:52:17 (UTC+05:45) | 428.0 B | Standard |
| index.html | html | August 6, 2024, 16:52:17 (UTC+05:45) | 405.0 B | Standard |

😊 Step 4: Triggering pipeline with GitHub pushes

- Now go to your local project folder to release a new version as you can see our index.html file and here we just changed the Website version to 2.0 from 1.0 and save our changes.

```

index.html M X
index.html > html > body > h2
1 <html>
2   <head>
3     <title>Sample Website - AWS CodePipeline Step by Step</title>
4   </head>
5   <body style="text-align:center;font-family:Verdana;margin:1%;">
6     <h1 style="color:#00008B">Welcome to Our Website!</h1>
7     <hr>
8     <p>This is a simple, static website that we deploy by creating a pipeline on AWS CodePipeline.</p>
9     <h2>Website Version: 2.0</h2>
10    </body>
11  </html>

```

- Next, we will use git commands to create a new commit from this change and push it to the master branch of our GitHub repository. And this should trigger our pipeline.
- In the below snapshot we used two commands first to check the status and we saw that one file needed to be committed, so we issued the commit command.

git status
git commit -a -m "Version 2.0"

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS POSTMAN CONSOLE AZURE

● PS C:\Users\PULKIT\Downloads\my-website> git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
    (use "git restore <file>..." to discard changes in working directory)
      modified:   index.html

no changes added to commit (use "git add" and/or "git commit -a")
● PS C:\Users\PULKIT\Downloads\my-website> git commit -a -m "Version 2.0"
warning: in the working copy of 'index.html', LF will be replaced by CRLF the next time Git touches it
[master a0d8fc1] Version 2.0
 1 file changed, 1 insertion(+), 1 deletion(-)
○ PS C:\Users\PULKIT\Downloads\my-website>

```

- Then we first checked the status after that we gave the push command.

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS POSTMAN CONSOLE AZURE

```

● PS C:\Users\PULKIT\Downloads\my-website> git status
On branch master
nothing to commit, working tree clean
● PS C:\Users\PULKIT\Downloads\my-website> git push origin master
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 12 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 315 bytes | 315.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/Pulkit-Kumar-0/demo-website-repo.git
  ee079cc..a0d8fc1  master -> master
○ PS C:\Users\PULKIT\Downloads\my-website>

```

- After that if you go back to your pipeline here you can see that your pipeline just finished running and you can see the version is 2.0.

GitHubPipelineDemo

Pipeline type: V2 Execution mode: QUEUED

Source Succeeded
Pipeline execution ID: [0c4b5169-3bb7-433c-9c3e-5af8da629046](#)

Source
[GitHub \(Version 2\)](#)
Succeeded - Just now
[a0d8fc11](#)
[View details](#)

[a0d8fc11](#) Source: Version 2.0

[Disable transition](#)

Deploy Succeeded
Pipeline execution ID: [0c4b5169-3bb7-433c-9c3e-5af8da629046](#)

Deploy
[Amazon S3](#)
Succeeded - Just now
[a0d8fc11](#)
[View details](#)

[a0d8fc11](#) Source: Version 2.0

[Start rollback](#)

- Now if you click on view details from the source then you will see the message.

Action execution details

X

Action name: Source Status: Succeeded

Summary

Input

Output

Status

 **Succeeded**

Last updated

1 minute ago

Action execution ID

 9bd66dbe-f5fc-460f-9be0-d48c6afacf29

Message

{"ProviderType":"GitHub","CommitMessage":"Version 2.0"}

Execution details

[View in CodeStarSourceConnection](#) 

Done

7. Also, if you refresh the website page you will see that the version has been changed to 2.0.

Welcome to Our Website!

This is a simple, static website that we deploy by creating a pipeline on AWS CodePipeline.

Website Version: 2.0

8. Now if you go to the history section from the left pane then you will see the Triggers and the source revisions.

S | Services | Search | [Alt+S] | X | A | ? | ⓘ | Ireland | DemoAccount |

Developer Tools | CodePipeline | Pipelines | GitHubPipelineDemo | Execution history

Execution history | Info | Rerun | Stop execution | View details | Release change | ⌂ | 1 | ⌂ | ⓘ | ⓘ

Execution ID | Status | Source revisions | Trigger | Started

0c4b5169 | Succeeded | Source - a0d8fc11 | Commit a0d8fc11 | Aug 6, 2024 5:08 PM
Version 2.0 | pushed in | Pulkit-Kumar-0/demo-website-repo/master | Version 2.0

1096bc73 | Succeeded | Source - ee079cc7 | CreatePipeline - root | Aug 6, 2024 4:52 PM
Version 1.0 | Version 1.0 |

Go to resource | Feedback | ⓘ | ⓘ