



Connect a GitHub Repo with AWS



naveen msd

```
Verifying      : perl-File-Find-1.37-477.amzn2023.0.7.noarch          5/8
Verifying      : perl-Git-2.50.1-1.amzn2023.0.1.noarch                6/8
Verifying      : perl-TermReadKey-2.38-9.amzn2023.0.2.x86_64          7/8
Verifying      : perl-lib-0.65-477.amzn2023.0.7.x86_64                8/8

Installed:
git-2.50.1-1.amzn2023.0.1.x86_64                               git-core-2.50.1-1.amzn2023.0.1.x86_64
git-core-doc-2.50.1-1.amzn2023.0.1.noarch                      perl-Error-1:0.17029-5.amzn2023.0.2.noarch
perl-File-Find-1.37-477.amzn2023.0.7.noarch                  perl-Git-2.50.1-1.amzn2023.0.1.noarch
perl-TermReadKey-2.38-9.amzn2023.0.2.x86_64                  perl-lib-0.65-477.amzn2023.0.7.x86_64

Complete!
● [ec2-user@ip-172-31-36-5 ~]$ git --version
git version 2.50.1
● [ec2-user@ip-172-31-36-5 ~]$
```



Introducing Today's Project!

In this project, I will demonstrate how to create a GitHub repo for a Web app and connect it with AWS, setting up the foundation for automated builds and deployments.

Key tools and concepts

The key services I used for this are GitHub. In this I learnt how to set up a GitHub repo, connect git to EC2 Instance, generate tokens and commands used in git and performed commit and push.

Project reflection

This project took me approximately took about an hour. It was most rewarding to see two git communicates and all the changes made were pushed, and I could see these reflecting on my GitHub repo

I did this project to understand how the Code is being maintained, stored, and updated automatically.



naveen msd
NextWork Student

nextwork.org

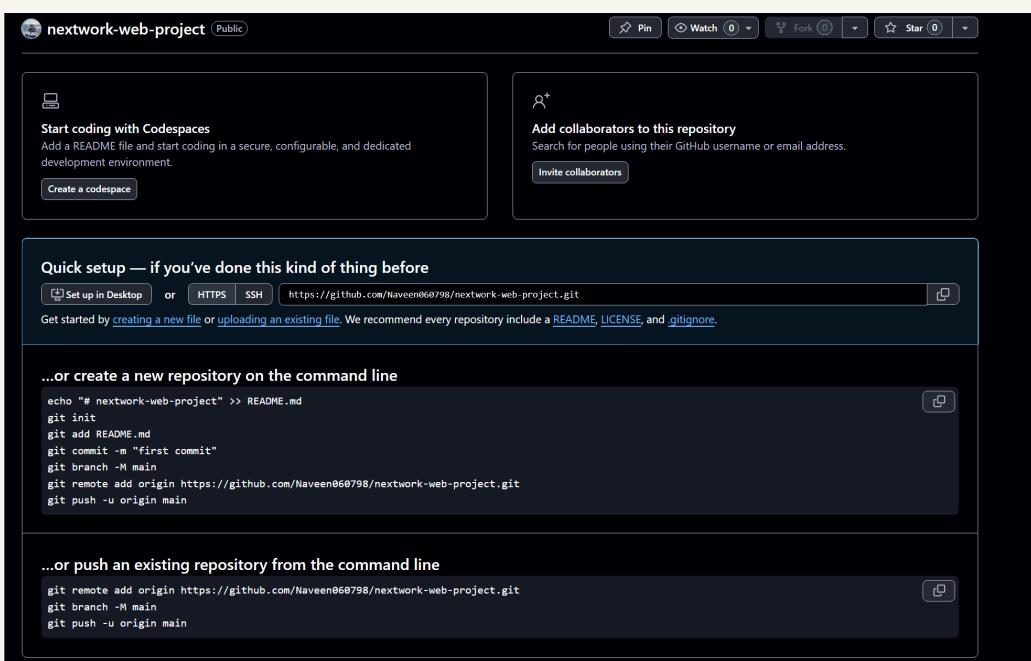
This project is part two of a series of DevOps projects where I'm building a CI/CD pipeline! I'll be working on the next project tomorrow.



Git and GitHub

Git is a Version Control it tracks changes by taking snapshots of what files look like at specific moments, and each snapshot is considered a 'version'. I installed Git using the command "sudo dnf install git -y"

GitHub is a storage space for different versions of the project that Git tracks. It lets me access the work from anywhere over the internet. GitHub helps me use Git and see my file changes in a more user-friendly way.





My local repository

A Git repository is like a folder that stores all the project files and their entire version history. Also, lets me collaborate with other developers over the internet.

Git init is a command that initiates to creation of a local git repo on the EC2 instance, where changes are now tracked for version control.

A branch in Git is a parallel version or 'alternate universe' of the same project. After running git init, the response from the terminal was successful in creating an empty repo and suggested changing the Main branch name.

```
[ec2-user@ip-172-31-36-5 nextwork-web-project]$ git init
hint: Using 'master' as the name for the initial branch. This default branch name
hint: is subject to change. To configure the initial branch name to use in all
hint: of your new repositories, which will suppress this warning, call:
hint:
hint:   git config --global init.defaultBranch <name>
hint:
hint: Names commonly chosen instead of 'master' are 'main', 'trunk' and
hint: 'development'. The just-created branch can be renamed via this command:
hint:
hint:   git branch -m <name>
hint:
hint: Disable this message with "git config set advice.defaultBranchName false"
Initialized empty Git repository in /home/ec2-user/nextwork-web-project/.git/
[ec2-user@ip-172-31-36-5 nextwork-web-project]$
```

To push local changes to GitHub, I ran three commands

git add

The first command I ran was `git diff --staged`. A staging area tells Git to put together all your modified files for a final review before you commit them. This is incredibly handy because all the edits are in one spot.

git commit

The second command I ran was `git commit -m "message"`. Using '`-m`' means leaving a message describing what the commit is about, making it easier to review what changed in this version.

git push

The third command I ran was `git push -u origin master`. Using '`-u`' means setting an 'upstream' for the local branch, which means you're telling Git to remember to push to master by default.



Authentication

When I commit changes to GitHub, Git asks for my credentials because Git needs to double-check that I have the right to push any changes to the remote origin your local repo is connected with.

Local Git identity

Git needs my name and email because Git needs author information for commits to track who made what change. If not set it manually, Git uses the system's default username, which might not accurately represent your identity in the project's.

Running git log showed me the changes that have been committed along with the commit ID, and also it had information like who made the changes, with the date and time.

```
[ec2-user@ip-172-31-36-5 ~]$ git log
bash: sit: command not found
[ec2-user@ip-172-31-36-5 .git]$ git log
commit 132090b8d101693c9f0c3714c0c5da446155bb92 (HEAD -> master, origin/master)
Author: EC2 Default User <ec2-user@ip-172-31-36-5.us-east-2.compute.internal>
Date:   Thu Sep 4 02:08:41 2025 +0000

    Updated index.jsp with new content
[ec2-user@ip-172-31-36-5 .git]$ git push -u origin master
branch 'master' set up to track 'origin/master'.
Everything up-to-date
[ec2-user@ip-172-31-36-5 .git]$ ]
```

GitHub tokens

GitHub authentication failed when I entered my password because GitHub phased out password authentication to connect with repositories over HTTPS - there are too many security risks and passwords can get intercepted over the internet.

A GitHub token is a secure string of characters (like a password) that GitHub generates for to authenticate and interact with GitHub services. I'm using one in this project to gain control.

I could set up a GitHub token by going to the settings page in GitHub> developer options> select Tokens> generate Token(classic)>Token expiration in 7days>select repo>and generate token.



New personal access token (classic)

Personal access tokens (classic) function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to [authenticate to the API over Basic Authentication](#).

Note

Devops_Nextwork

What's this token for?

Expiration

7 days (Sep 11, 2025) - The token will expire on the selected date

Select scopes

Scopes define the access for personal tokens. [Read more about OAuth scopes](#).

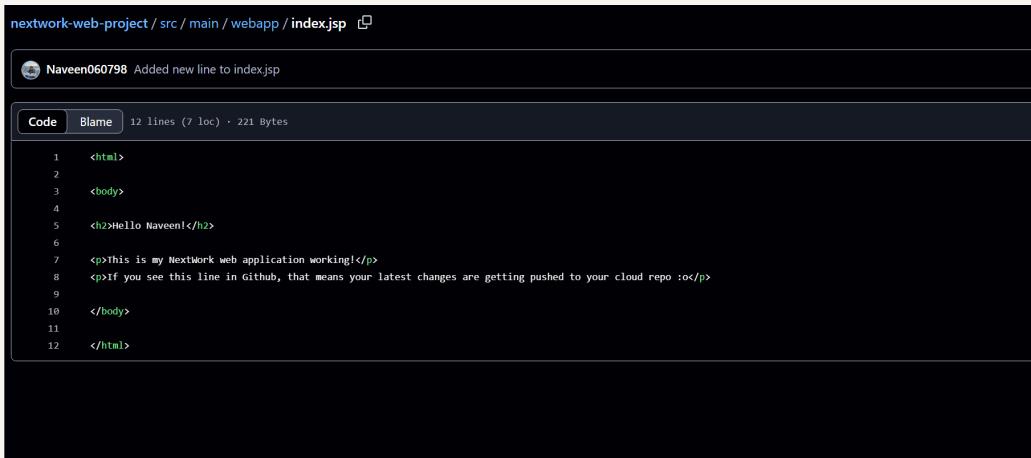
<input checked="" type="checkbox"/> repo	Full control of private repositories
<input type="checkbox"/> repostatus	Access commit status
<input type="checkbox"/> repo_deployment	Access deployment status
<input type="checkbox"/> public_repo	Access public repositories
<input type="checkbox"/> repo_invite	Access repository invitations
<input type="checkbox"/> security_events	Read and write security events
<hr/>	
<input type="checkbox"/> workflow	Update GitHub Action workflows
<input type="checkbox"/> write_packages	Upload packages to GitHub Package Registry
<input type="checkbox"/> read_packages	Download packages from GitHub Package Registry
<input type="checkbox"/> delete_packages	Delete packages from GitHub Package Registry
<hr/>	
<input type="checkbox"/> admin:org	Full control of orgs and teams; read and write org projects
<input type="checkbox"/> write:org	Read and write org and team membership; read and write org projects
<input type="checkbox"/> read:org	Read org and team membership; read org projects
<input type="checkbox"/> manage:runners:org	Manage org runners and runner groups
<hr/>	
<input type="checkbox"/> admin:public_key	Full control of user public keys
<input type="checkbox"/> write:public_key	Write user public keys
<input type="checkbox"/> read:public_key	Read user public keys
<input type="checkbox"/> admin:repo_hook	Full control of repository hooks
<input type="checkbox"/> write:repo_hook	Write repository hooks
<input type="checkbox"/> read:repo_hook	Read repository hooks
<input type="checkbox"/> admin:org_hook	Full control of organization hooks
<input type="checkbox"/> gist	Create gists
<input type="checkbox"/> notifications	Access notifications



Making changes again

I wanted to see Git working in action, so I updated the index.jsp where I added a new line.

I finally saw the changes in my GitHub repo after adding the file, committing the changes, and pushing them. I could see the changes reflecting on the repository.



A screenshot of a GitHub code diff interface. The title bar says "nextwork-web-project / src / main / webapp / index.jsp". Below it, a commit message from "Naveen060798" states "Added new line to index.jsp". The "Code" tab is selected, showing 12 lines of JavaServer Pages (JSP) code. The code contains two new lines added by the commit:

```
1 <html>
2
3 <body>
4
5 <h2>Hello Naveen!</h2>
6
7 <p>This is my NextWork web application working!</p>
8 <p>If you see this line in GitHub, that means your latest changes are getting pushed to your cloud repo :o</p>
9
10 </body>
11
12 </html>
```



nextwork.org

The place to learn & showcase your skills

Check out nextwork.org for more projects

