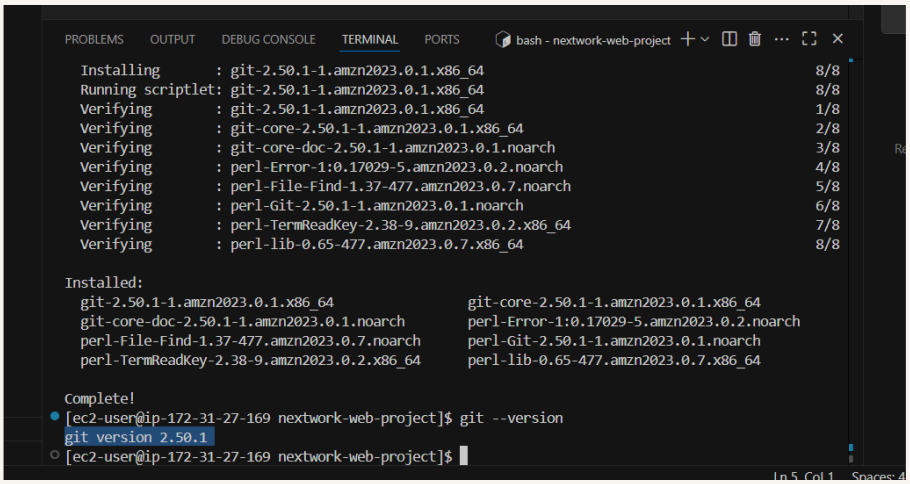
nextwork.org

Connect a GitHub Repo with AWS

S shilpa kale



Introducing Today's Project!

Today we are here to setup a Git repository for our web app's code. This is project TWO in our 7 Day DevOps challenge. By the end of the project, the code that we write for our Java web app will be stored securely in GitHub.

Key tools and concepts

Services we used were GitHub, Amazon EC2, which was our development instance. We also used Key pairs and VS Code. Key concepts we learnt include setting up a repository, the difference between Git and GitHub, the git commands for stagging, saving.

Project reflection

This project took us approximately two hours including Demo. The most challenging part was to learn all the different git commands. It was most rewarding to set up a README and seeing the code popup in GitHub repository was fun.

We did this project because to learn more about git and GitHub and it's Day 2 of 7 Day DevOps Challenge.

S

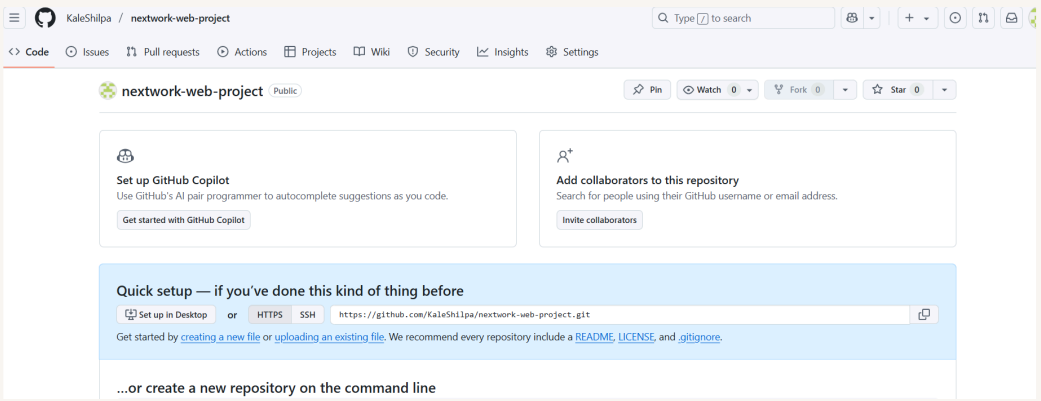
shilpa kale
NextWork Student

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

Git and GitHub

Git is a version control system which means it's used to track changes that we make to our code. It's aslo helpfull for collaboration. With Git you can see who made what change and handover from dev to deploy is a lot smoother. We used 'sudo update'

GitHub is a storage space for different version of your project that Git tracks. Since GitHub is a cloud service, it also lets you access your work from anywhere and collaborate with other developers over the internet.

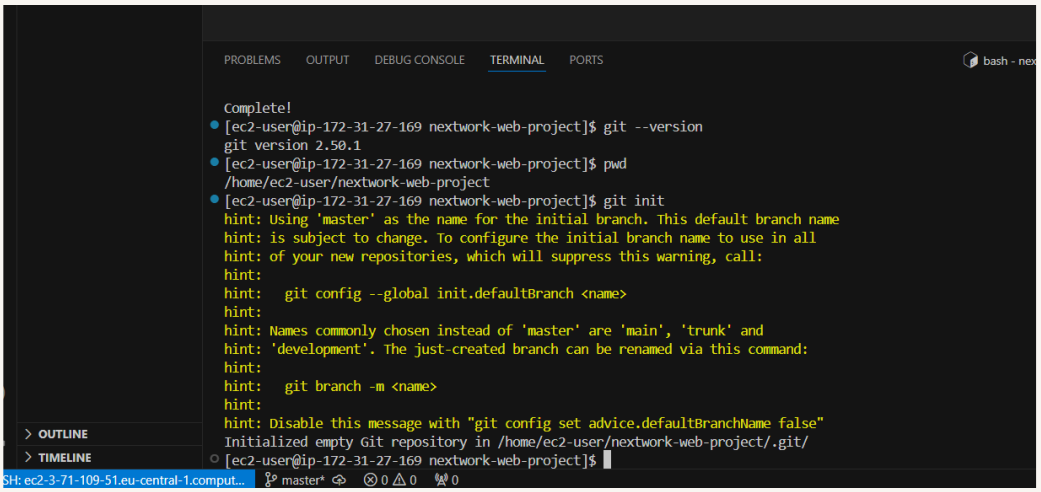


My local repository

A Git repository is a like a online folder that you can use to store your web apps code and all the versions of that code. Think of it as the go to place for all the code and updates and changes of specific project.

Git init is a command that initializes git in our local repo. We ran git init in our web app project folder, which tells our terminal that we want to start tracking changes locally.

After running git init, the response from the terminal was that we initialized git and by default we were using the main branch. A branch is like a version of your code. You make changes to your code over a branch and then merge it into main branch.


PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
bash - nex
Complete!
[ec2-user@ip-172-31-27-169 nextwork-web-project]$ git --version
git version 2.50.1
[ec2-user@ip-172-31-27-169 nextwork-web-project]$ pwd
/home/ec2-user/nextwork-web-project
[ec2-user@ip-172-31-27-169 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-172-31-27-169 nextwork-web-project]$
OUTLINE
TIMELINE
SH: ec2-3-71-109-51.eu-central-1.comput...
master*
0 0
0

S
shilpa kale
NextWork Student

To push local changes to GitHub, I ran three commands

git add

The first command we ran was git add . , which adds changes to stagging area. A staging area is place to review all changes made to the code, so that you can decide what changes you like to save.

git commit

The second command we ran was git commit, which is the command for saving the change in our staging area. Using '-m' means we are also leaving a message for that commit.

git push

The third command we ran was git push -u origin master. This command pushes the changes to our remote origin. Using '-u' means' a special flag that makes our remote origin as default upstream remote repository.

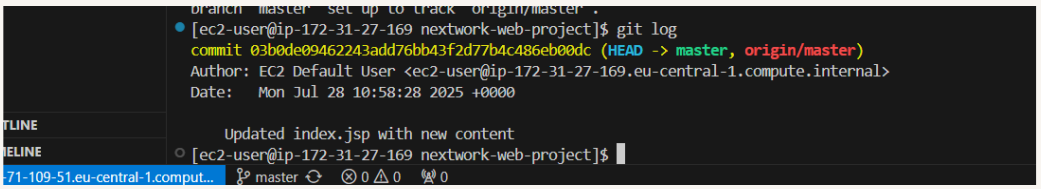
Authentication

When we commit changes to GitHub, Git asks for my credentials because it needs to authenticate you before letting us make code changes to the repo. It needs to know that you have the right to do changes

Local Git identity

Git needs our name and email because it's a version control system which means it's used to tracking changes to code and really who made what changes to code. To really identify the people that made x changes, git will need to know name and email.

Running git log showed us that git is saving our code changes to a user name "EC2 Default User" instead of our actual name in details.

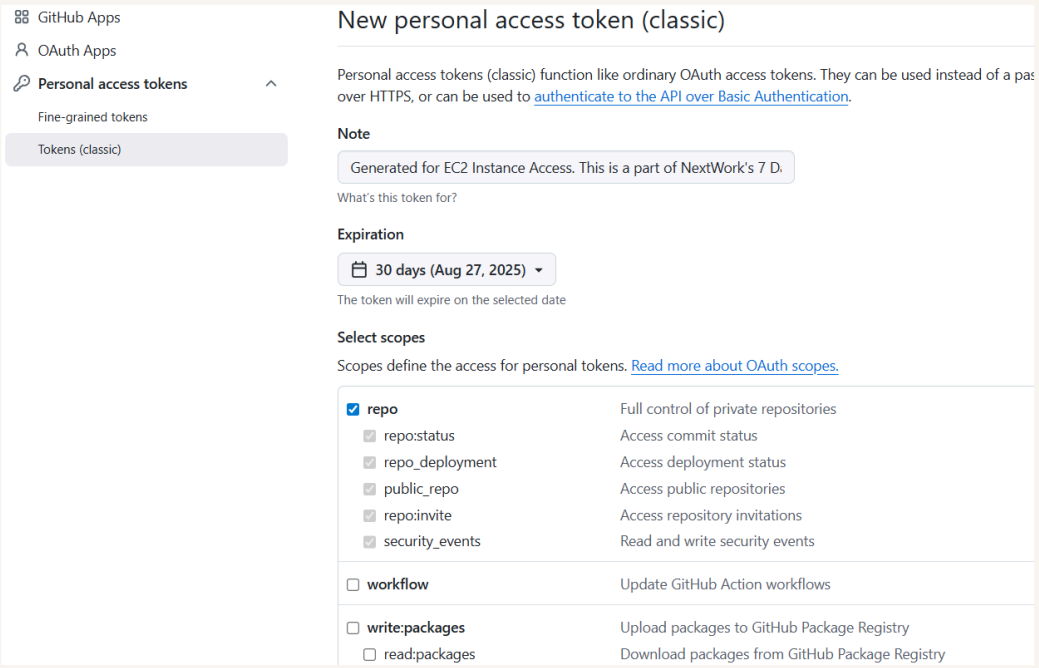


GitHub tokens

GitHub authentication failed when we entered my password because password authentication was removed. Because there are too many risks over the terminal to GitHub. Now more secure ways are available.

A GitHub token is like a temporary password that grants access to your GitHub account. We're using one in this project because it let's safely authenticate to our GitHub repositories while in our EC2 Instance.

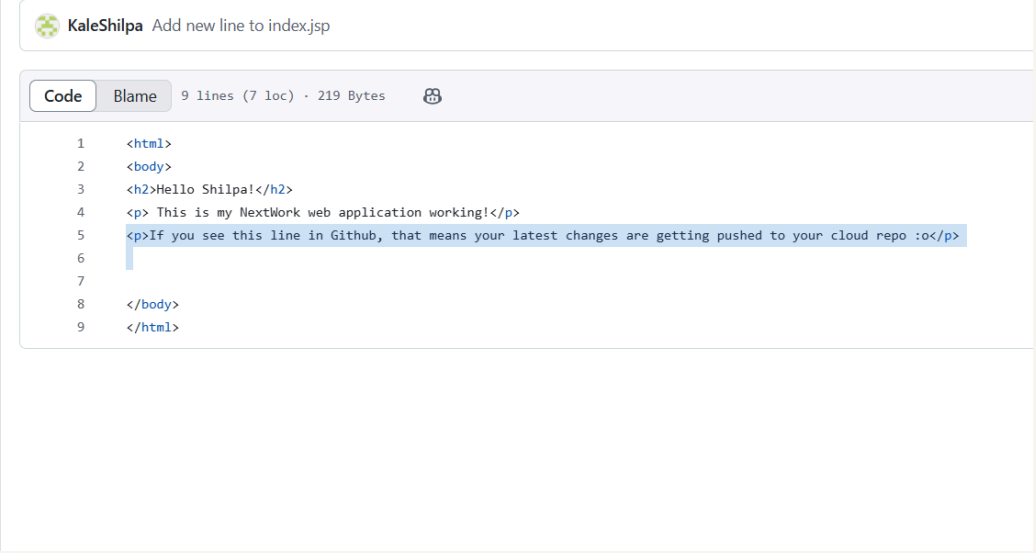
We could set up a GitHub token by visiting Developer Setting in GitHub and we setup a token that expires in 7 days and only allows persmission to repo.


GitHub Apps
OAuth Apps
Personal access tokens
Fine-grained tokens
Tokens (classic)
New personal access token (classic)
Personal access tokens (classic) function like ordinary OAuth access tokens. They can be used instead of a pas
over HTTPS, or can be used to authenticate to the API over Basic Authentication.
Note
Generated for EC2 Instance Access. This is a part of NextWork's 7 D
What's this token for?
Expiration
30 days (Aug 27, 2025)
The token will expire on the selected date
Select scopes
Scopes define the access for personal tokens. Read more about OAuth scopes.
repo
Full control of private repositories
repo:status
Access commit status
repo_deployment
Access deployment status
public_repo
Access public repositories
repo:invite
Access repository invitations
security_events
Read and write security events
workflow
Update GitHub Action workflows
write:packages
Upload packages to GitHub Package Registry
read:packages
Download packages from GitHub Package Registry

Making changes again

We wanted to see Git working in action, so we made another change to our index.jsp file. We couldn't see the changes in our GitHub repo initially because we had not added, commited or pushed those changes. After running the 3 commands, we could see:o

We finally saw the changes in my GitHub repo after running the same three commands and then refreshing index.jsp in our GitHub repository.



The place to learn & showcase your skills

Check out nextwork.org for more projects