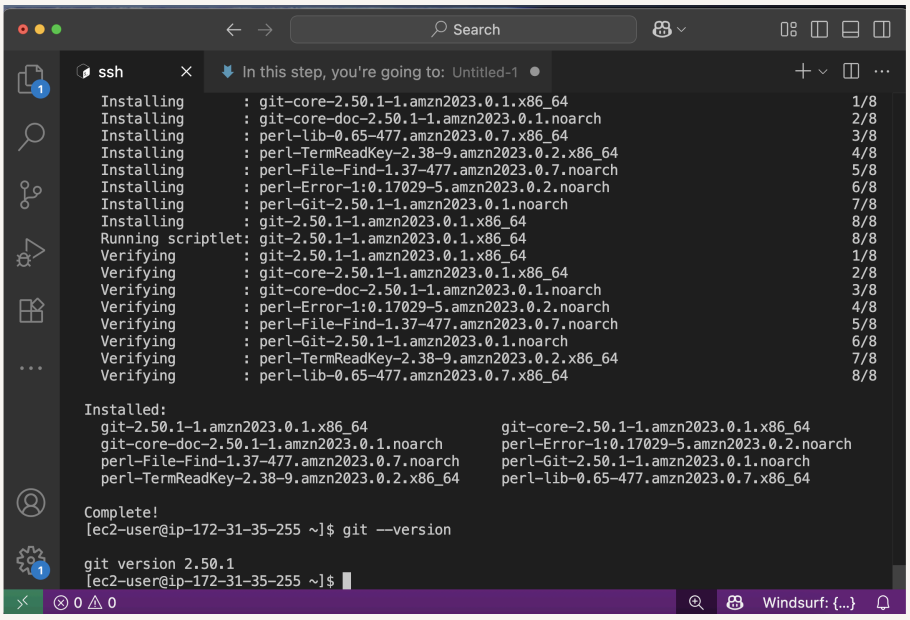
nextwork.org

Connect a GitHub Repo with AWS

Kehinde Abiuwa



Kehinde Abiuwa
NextWork Student

Introducing Today's Project!

In this project, I will demonstrate how to set up Git and GitHub, link a local web app project to a GitHub repository, track and push code changes, and create a README file to document the project. I'm doing this project to learn how version control with Git and GitHub works, how to manage my code efficiently, and how to collaborate or share my web app code using GitHub.

Key tools and concepts

Services I used were Amazon EC2 for hosting my web app environment, GitHub for remote version control and code hosting, and VS Code as my local development editor with Remote - SSH for connecting to EC2. Key concepts I learnt include how to set up and manage a Git repository, connect local and remote repositories, use Git commands to track and push changes, authenticate securely with GitHub tokens, and collaborate on code through GitHub.

Project reflection

This project took me approximately 40 minutes to complete. The most challenging part was setting up the GitHub authentication with a personal access token because password authentication was no longer supported, which required learning a new process. It was most rewarding to see my changes successfully pushed and reflected in my GitHub repository, knowing I had full version control and a reliable backup for my web app code.

Kehinde Abiuwa
NextWork Student

I did this project because I wanted to learn how to use Git and GitHub for version control, improve my workflow by managing code changes efficiently, and understand how to connect my local projects with remote repositories for better collaboration and backup.

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 right away!

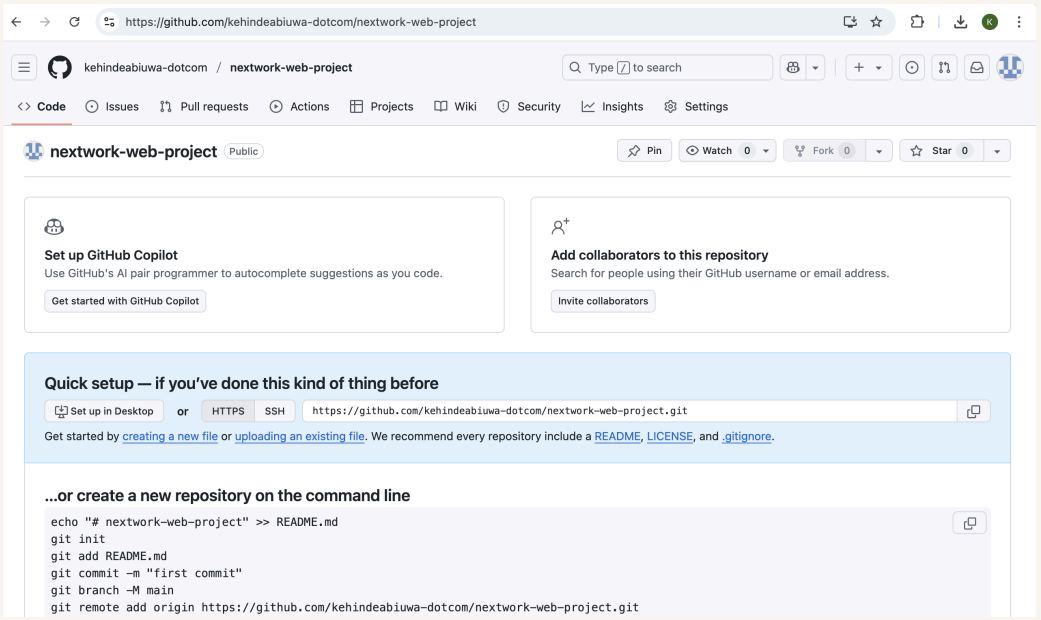
Kehinde Abiuwa
NextWork Student

Git and GitHub

Git is a version control system that helps developers track and manage changes to their code over time. It lets you save snapshots of your project at different points, collaborate with others without overwriting each other's work, and easily revert to earlier versions if needed. In simple terms, Git helps you keep a history of your code and work together with others efficiently. I installed Git using the commands -- 'sudo dnf update -y sudo dnf install git -y'

GitHub is a web-based platform that hosts Git repositories, making it easy to store, share, and collaborate on code projects online. I'm using GitHub in this project to keep my web app code safely stored in the cloud, track changes with Git, collaborate with others if needed, and showcase my work publicly or privately.

Kehinde Abiuwa
NextWork Student



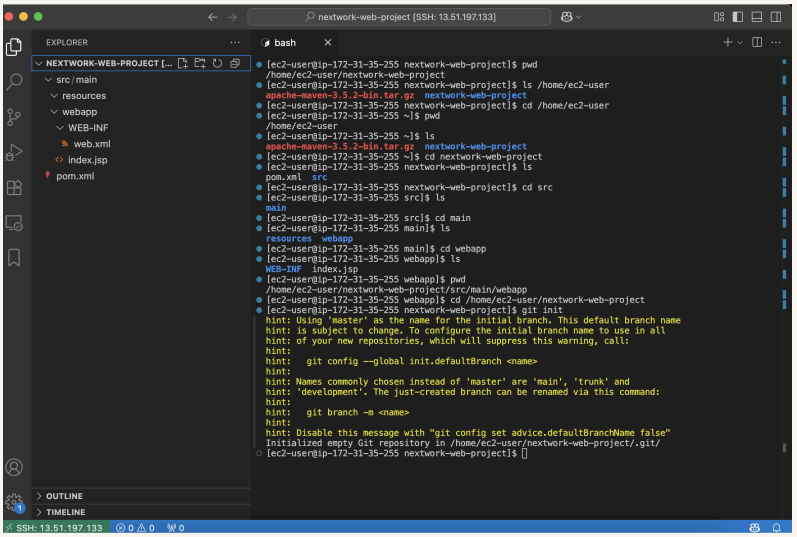
My local repository

A Git repository is a special folder in your project that stores all the files and the full history of changes made to those files using Git. It lets you track, manage, and revert changes, making version control possible.

git init is a command that initializes a new Git repository in a folder, setting up all the necessary files Git needs to start tracking changes. I ran git init in my project folder on the EC2 instance to begin version controlling my web app code there.

A branch in Git is a separate line of development within a repository that lets you work on different features or fixes without affecting the main codebase. After running git init, the response from the terminal was something like: “Initialized empty Git repository in /home/ec2-user/nextwork-web-project/.git/” — which means Git successfully set up the repository and is ready to track my changes.

Kehinde Abiuwa
NextWork Student



Kehinde Abiuwa
NextWork Student

To push local changes to GitHub, I ran three commands

git add

The first command I ran was git remote add origin https://github.com/kehindeabiuwa-dotcom/nextwork-web-project.git which links my local Git repository to my GitHub repository so I can push and pull changes between them. A staging area is a place in Git where you prepare changes before committing them. It lets you review and select which changes you want to include in the next commit, acting like a buffer between your working files and the repository history.

git commit

The second command I ran was git add . and then git commit -m "Updated index.jsp with new content" The purpose of these commands is to first stage all the changes in the project (git add .) and then save them as a new snapshot in the repository history (git commit). Using -m means you can provide a commit message directly in the command, describing what changes you made—in this case, updating the index.jsp file with new content.

git push

The third command I ran was git push -u origin master This command uploads my local master branch to the remote repository named origin on GitHub. Using -u means I'm setting the upstream branch, so in the future I can simply run git push or git pull without having to specify the remote and branch name every time.

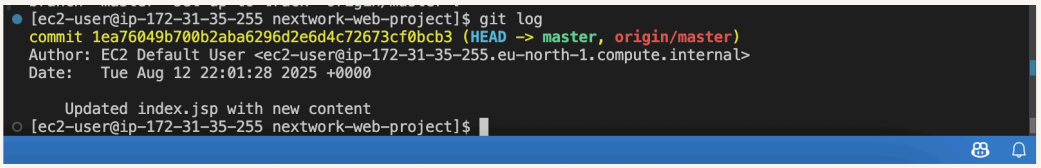
Authentication

When I commit changes to GitHub, Git asks for my credentials because GitHub needs to verify my identity before allowing me to push changes to a repository. Providing my username and password (or personal access token) ensures that only authorized users can update the code.

Local Git identity

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

Running git log showed me my history of commits, including details such as the commit hash, date, commit message, and the author’s name. This information helps track who made each change and when it was made.

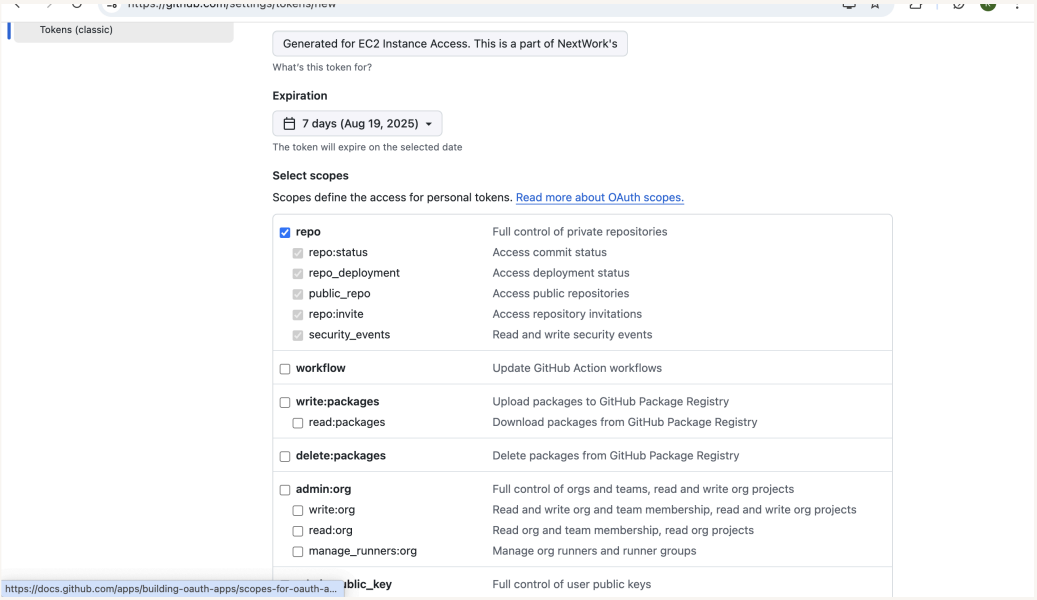

[ec2-user@ip-172-31-35-255 nextwork-web-project]$ git log
commit 1ea76049b700b2aba6296d2e6d4c72673cf0bcb3 (HEAD -> master, origin/master)
Author: EC2 Default User <ec2-user@ip-172-31-35-255.eu-north-1.compute.internal>
Date: Tue Aug 12 22:01:28 2025 +0000
Updated index.jsp with new content
[ec2-user@ip-172-31-35-255 nextwork-web-project]$

GitHub tokens

GitHub authentication failed when I entered my password because git has recently phased out password authentication to connect with repositories over HTTPS as there are alot of security risks involved. I need to use a personal access token instead, which is a more secure method for logging in and interacting with your repos.

A GitHub token is a secure, unique string of characters generated by GitHub that works like a password but can be limited in scope and revoked at any time. It's used to authenticate you when accessing GitHub from the command line or other tools. I'm using one in this project because GitHub no longer accepts passwords for HTTPS connections, and a personal access token provides a secure way to push and pull code between my local repository and my GitHub repository.

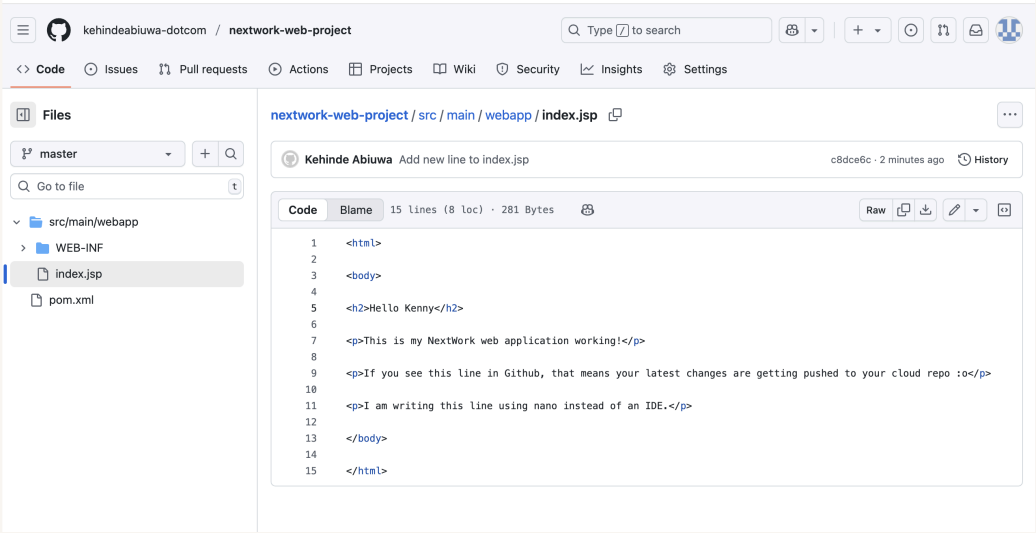
I could set up a GitHub token by logging into my GitHub account, going to Settings → Developer settings → Personal access tokens → Tokens (classic), clicking Generate new token, selecting the scopes/permissions I need (such as repo for full repository access), generating the token, and then copying it to use in place of a password when pushing or pulling code via Git.


Tokens (classic)
Generated for EC2 Instance Access. This is a part of NextWork's
What's this token for?
Expiration
7 days (Aug 19, 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
delete:packages
Delete packages from GitHub Package Registry
admin:org
Full control of orgs and teams, read and write org projects
write:org
Read and write org and team membership, read and write org projects
read:org
Read org and team membership, read org projects
manage_runners:org
Manage org runners and runner groups
Full control of user public keys
https://docs.github.com/apps/building-oauth-apps/scopes-for-oauth-a...

Making changes again

I wanted to see Git working in action, so I edited the index.jsp file, staged, committed, and then pushed the updates to my GitHub repository. I couldn't see the changes in my GitHub repo at first because I hadn't pushed them yet—committing only saves changes locally, and pushing sends them to the remote repository.

I finally saw the changes in my GitHub repo after running the git push command, which uploaded my committed changes from the local repository to the remote repository on GitHub.



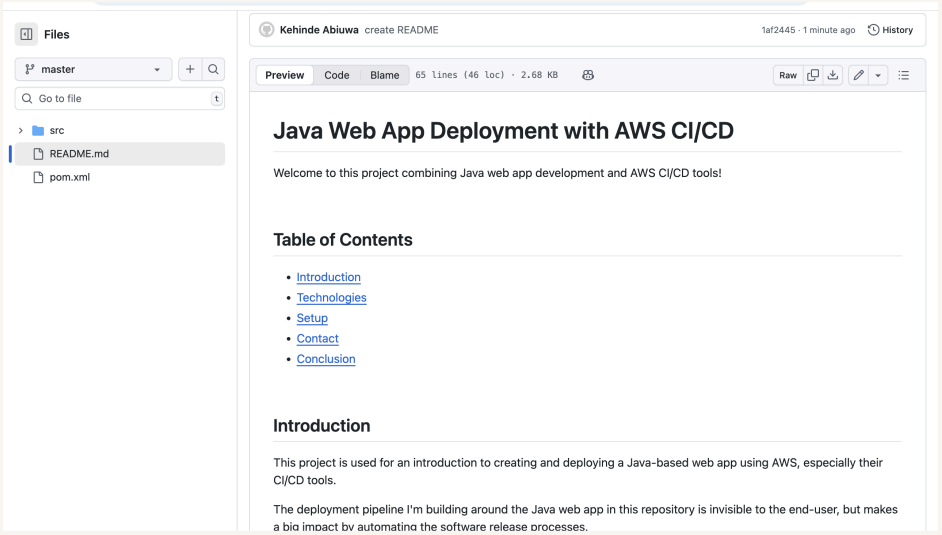
Setting up a README file

As a finishing touch to my GitHub repository, I added a README file, which is a markdown document that provides important information about the project, such as what it does, how to use it, and any other relevant details. I added a README file by creating a new README.md file in my project directory, writing the content, then staging, committing, and pushing it to the GitHub repository.

My README is written in Markdown because it's a simple, lightweight markup language that lets you easily format text with headings, lists, links, and more — all in plain text that's easy to read and write. Special characters can help you format text in Markdown, such as: # for headings * or - for bullet points ` for inline code [text](url) for links

My README file has 6 sections that outline the Table of Contents, Introduction, Technologies, Setup ,Contact & Conclusion.

Kehinde Abiuwa
NextWork Student



The place to learn & showcase your skills

Check out nextwork.org for more projects