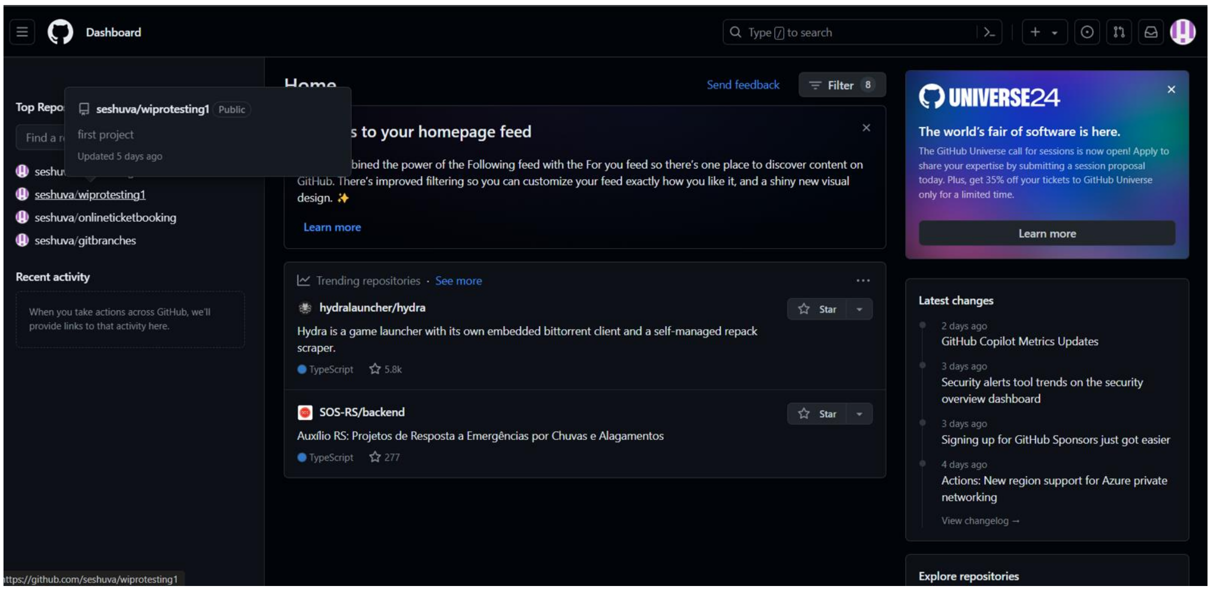
Day 1 Assignments:

Assignment 1: Initialize a new Git repository in a directory of your choice. Add a simple text file to the repository and make the first commit.


Dashboard

Type / to search

Top Repo

Find a r

seshuva/wiprotesting1 Public

first project

Updated 5 days ago

seshuva/wiprotesting1

seshuva/onlineticketbooking

seshuva/gitbranches

Recent activity

When you take actions across GitHub, we'll provide links to that activity here.

Home

Send feedback

Filter 8

s to your homepage feed

bined the power of the Following feed with the For you feed so there's one place to discover content on GitHub. There's improved filtering so you can customize your feed exactly how you like it, and a shiny new visual design. ✨

Learn more

Trending repositories · See more

hydralauncher/hydra

Hydra is a game launcher with its own embedded bittorrent client and a self-managed repack scraper.

TypeScript ☆ 5.8k

Star

SOS-RS/backend

Auxílio RS: Projetos de Resposta a Emergências por Chuvas e Alagamentos

TypeScript ☆ 277

Star

UNIVERSE24

The world's fair of software is here.

The GitHub Universe call for sessions is now open! Apply to share your expertise by submitting a session proposal today. Plus, get 35% off your tickets to GitHub Universe only for a limited time.

Learn more

Latest changes

2 days ago
GitHub Copilot Metrics Updates

3 days ago
Security alerts tool trends on the security overview dashboard

3 days ago
Signing up for GitHub Sponsors just got easier

4 days ago
Actions: New region support for Azure private networking

View changelog →

Explore repositories

ttps://github.com/seshuva/wiprotesting1

Figure 1/Created a new repository

Open terminal (Command Prompt or Git Bash on Windows, Terminal on macOS and Linux).

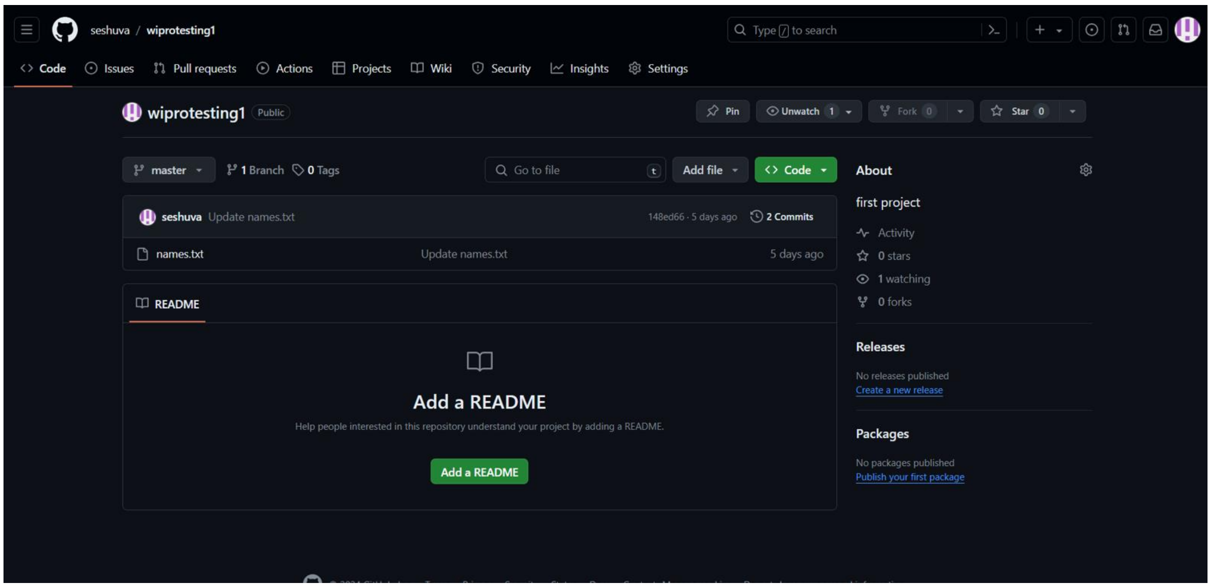
1. Navigate to the directory where you want to create the repository using the `cd` command. For example:
2. `cd path/to/your/directory`
3. Once you're in the desired directory, **initialize a new Git repository** with the following command:
4. `git init`
5. Create a new text file in the repository. You can do this from the terminal or your file explorer. If using the terminal, you can use the `touch` command (on macOS and Linux) or `type>` command (on Windows). For example:
6. `touch names.txt`
7. `# or on Windows`
8. `type> nemes.txt`
9. Add some content to your text file using a text editor of your choice.

10. After saving your changes, **add the file to the staging area** with the following command:

```
11.  git add names.txt
```

12. Now, **commit the file** to the repository with a commit message. The message should describe the changes you've made. Here's how you can make the commit:

```
13. git commit -m "initial commit"
```



Help people interested in this repository understand your project by adding a README.

Figure 2/Created a file inside a new repository

Assignment 2: Branch Creation and Switching

Create a new branch named 'feature' and switch to it. Make changes in the 'feature' branch and commit them.

1. *Create and Switch to the New Branch:*

git checkout -b feature

This command creates a new branch named 'feature' and switches to it. Any changes you make from now on will be in this branch.

2. *Make Changes*: Now you can make any necessary changes to your files. Let's say you want to modify the 'names.txt' file:

nano names.txt

This command opens the 'names.txt' file in the nano text editor where you can make your modifications. Once you're done, save and exit the editor.

3. *Add Changes:*

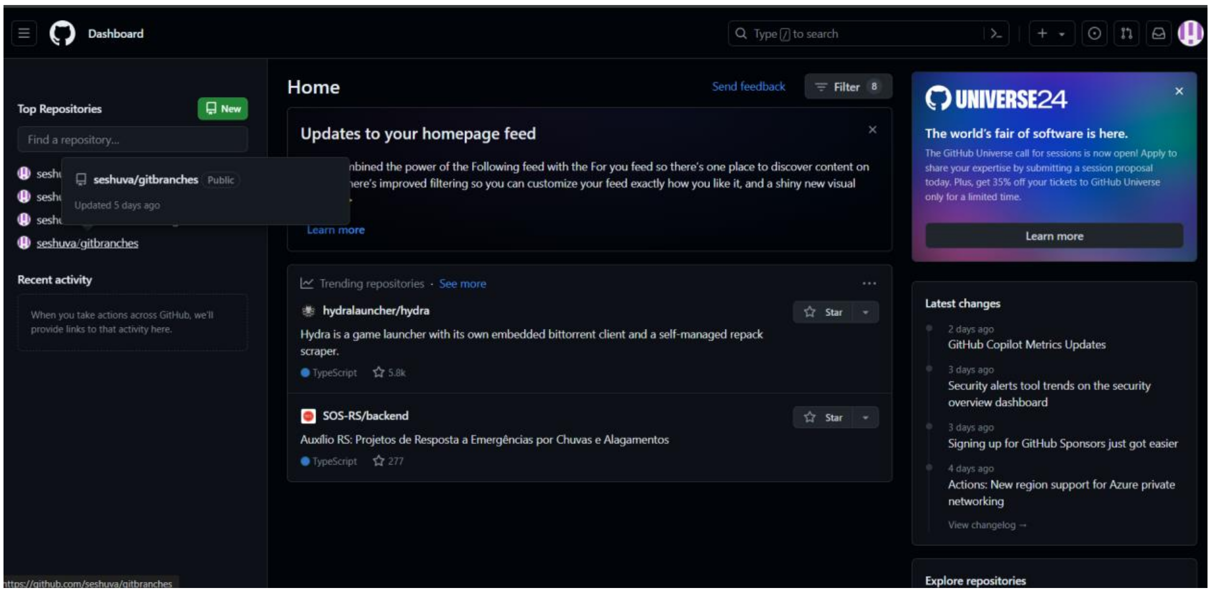
git add names.txt

This command stages the changes you made to the 'names.txt' file, preparing them to be committed.

4. *Commit Changes:*

git commit -m "Add new names to names.txt in feature branch"

This command commits the changes you made in the 'feature' branch with a descriptive message.



Top Repositories

Find a repository...

seshuva/gitbranches Public

Updated 5 days ago

seshuva/gitbranches

Recent activity

When you take actions across GitHub, we'll provide links to that activity here.

Home

Updates to your homepage feed

nbined the power of the Following feed with the For you feed so there's one place to discover content on here's improved filtering so you can customize your feed exactly how you like it, and a shiny new visual

Trending repositories

hydralauncher/hydra

Hydra is a game launcher with its own embedded bittorrent client and a self-managed repack scraper.

TypeScript 5.8k

SOS-RS/backend

Auxílio RS: Projetos de Resposta a Emergências por Chuvas e Alagamentos

TypeScript 277

The world's fair of software is here.

The GitHub Universe call for sessions is now open! Apply to share your expertise by submitting a session proposal today. Plus, get 35% off your tickets to GitHub Universe only for a limited time.

Latest changes

2 days ago
GitHub Copilot Metrics Updates

3 days ago
Security alerts tool trends on the security overview dashboard

3 days ago
Signing up for GitHub Sponsors just got easier

4 days ago
Actions: New region support for Azure private networking

Explore repositories

https://github.com/seshuva/gitbranches

Figure 2/gitbranches repository is created

Assignment 3: Feature Branches and Hotfixes

Create a 'hotfix' branch to fix an issue in the main code. Merge the 'hotfix' branch into 'main' ensuring that the issue is resolved.

1. Create and Switch to the Hotfix Branch:

git checkout -b hotfix

This command creates a new branch named 'hotfix' and switches to it. You'll address the issue in this branch.

2. Make the Necessary Fixes: Locate and fix the issue in your codebase. This might involve modifying one or more files depending on the nature of the problem.
3. Add and Commit the Fixes: Once you've made the necessary changes, add them to the staging area and commit them:

git add <file(s) with fixes> git commit -m "Fix issue in main code"

4. Switch Back to the Main Branch:

git checkout main

This command switches back to the 'main' branch where you'll merge the hotfix.

5. Merge the Hotfix Branch into Main:

git merge hotfix

This command merges the changes from the 'hotfix' branch into the 'main' branch. If there are no conflicts, the changes will be applied automatically.

6. Resolve any Conflicts (if Necessary): If there are conflicts between the changes made in the 'hotfix' branch and the 'main' branch, you'll need to resolve them manually. Git will prompt you to resolve conflicts using a text editor or a merge tool.
7. Commit the Merge: After resolving any conflicts, commit the merge:

git commit

Git will open your default text editor for you to enter a merge commit message. Save and close the editor when you're done.

8. Push the Changes to Remote Repository (if needed): If you're working with a remote repository and want to push the changes to it:

git push origin main

Now, the 'hotfix' branch has been merged into the 'main' branch, resolving the issue in the main code. You can delete the 'hotfix' branch if it's no longer needed:

git branch -d hotfix

This command deletes the 'hotfix' branch locally. If you've already pushed it to a remote repository and want to delete it there as well:

```
git push origin --delete hotfix
```