

Pull, Fork and Branch

What is Git pull?

If you are working on a project and collaborating with other people, you need to sync with remote repositories. Git Pull is a command used to update the local version of a repository from a remote repository. So, the first step is to 'fetch the remote repository and update the current local working branch (currently checked out branch) (i.e. HEAD is pointed at). After the content is downloaded, your local branch is compared to the remote-tracking branch and receives the new commits so it can catch up to the current state of the remote branch.

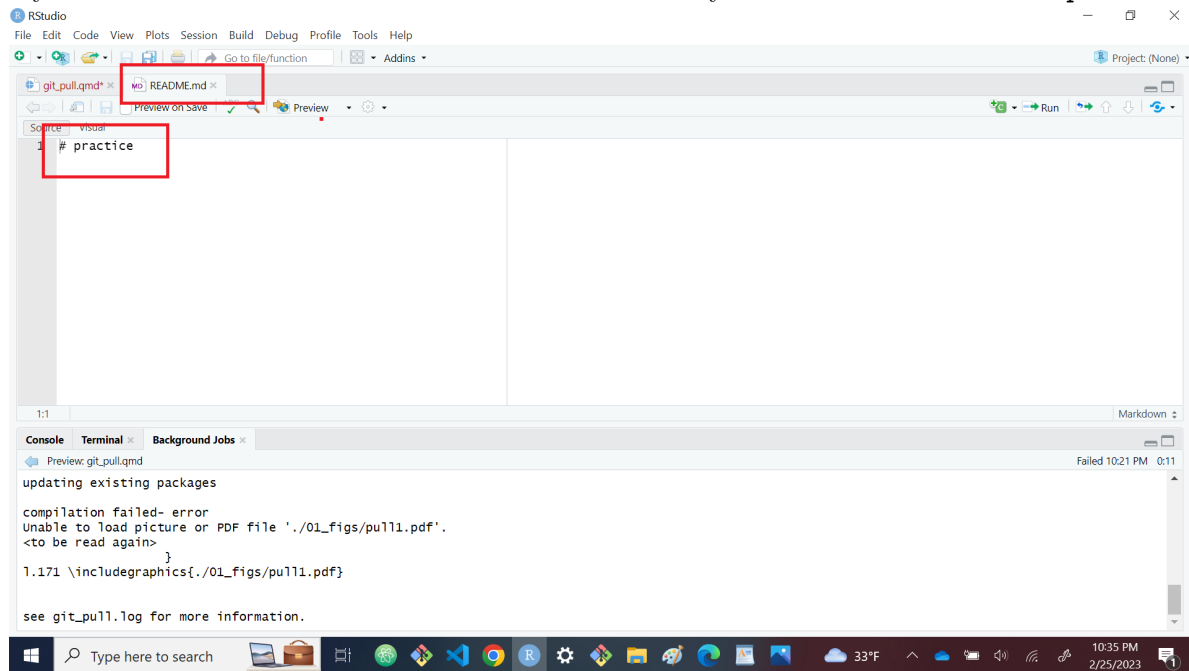
So `git pull` is a combination of `git fetch` and `git merge`. - Git Fetch: It is the process of updating your remote -tracking branches.

- Git Merge: It is used to merge the received commits into a local branch.
- Git Pull: It is a shortcut and does both `git fetch` and `git pull`. When you perform `git pull`, Git performs `git fetch` which downloads all the changes from your remote to your local repository. Then immediately performs a `git merge` which then applies those changes to your local repository.
- You need to run `git pull` at least once a day. Without running `git pull` your local repository will never be updated with changes from the remote. That's why `git pull` is one of the most used Git commands.
- **Question:** What is git rebase?

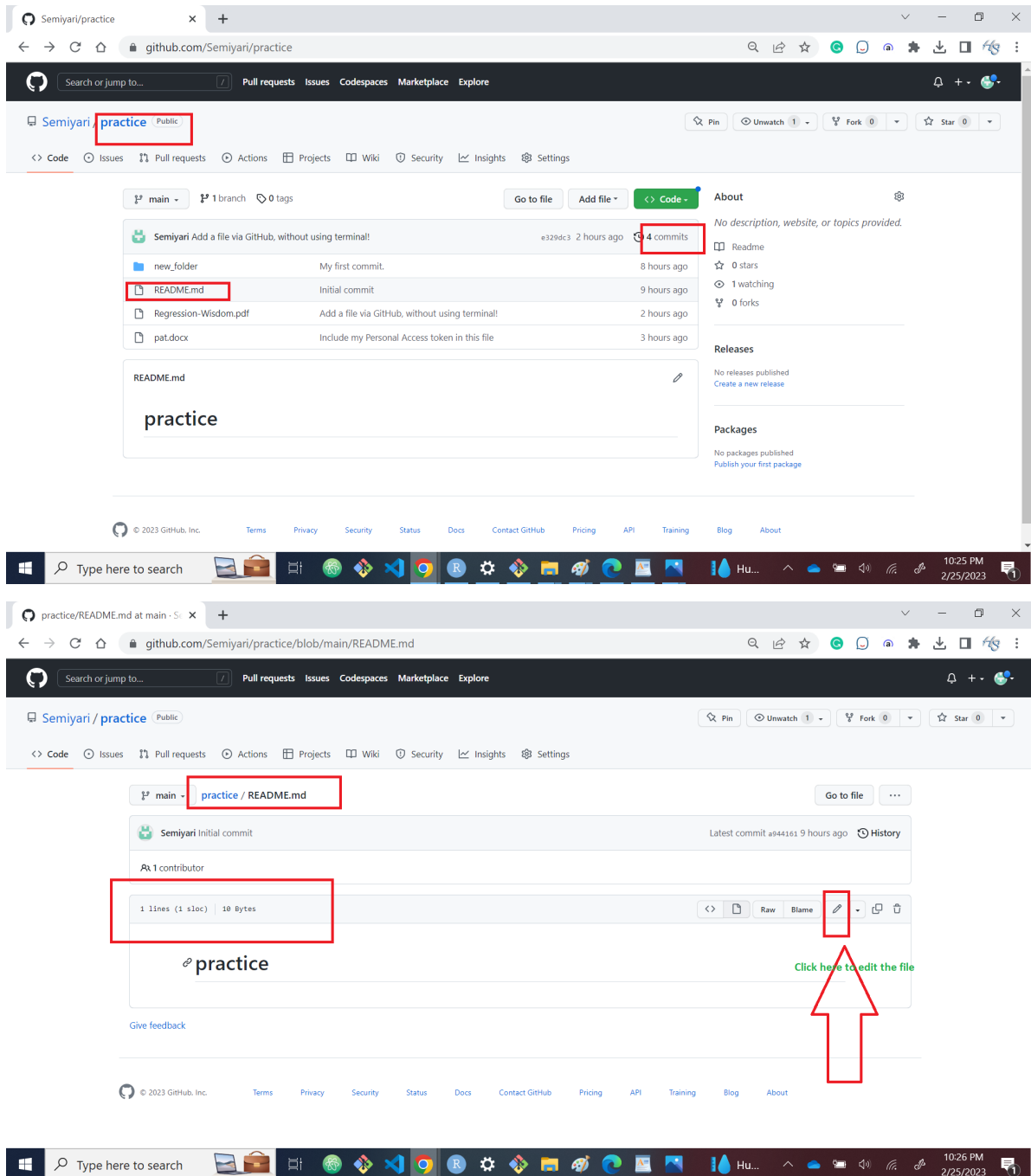
Rebase is one of two Git utilities that specializes in integrating changes from one branch onto another. The other change integration utility is `git merge`. Merge is always a forward moving change record. Alternatively, rebase has powerful history rewriting features.

Pull

Assume some one in our team has done some changes to the file README.md and push it to the `practice` repository. I have opened README.md and add a line. Why? Just to mimic that someone has changed the file and push it there. In my local machine the file README.md has only one line and it is `# practice`)



- Here are the steps that I took to imitate the action of a push file to the repository by a team member.



Editing practice/README.md at `main`

github.com/Semiyari/practice/edit/main/README.md

Search or jump to... Pull requests Issues Codespaces Marketplace Explore

Semiyari / practice Public Pin Unwatch 1 Fork 0 Star 0

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

practice / README.md in `main` Cancel changes

Edit file Preview Spaces 2 Soft wrap

```
1 # practice
2 This repository is just for practice!
```

Type here to search

10:27 PM 2/25/2023

Editing practice/README.md at `main`

github.com/Semiyari/practice/edit/main/README.md

Attach files by dragging & dropping, selecting or pasting them.

Commit changes

Add new line to practice pull.

Add an optional extended description...

☒ Commit directly to the `main` branch.

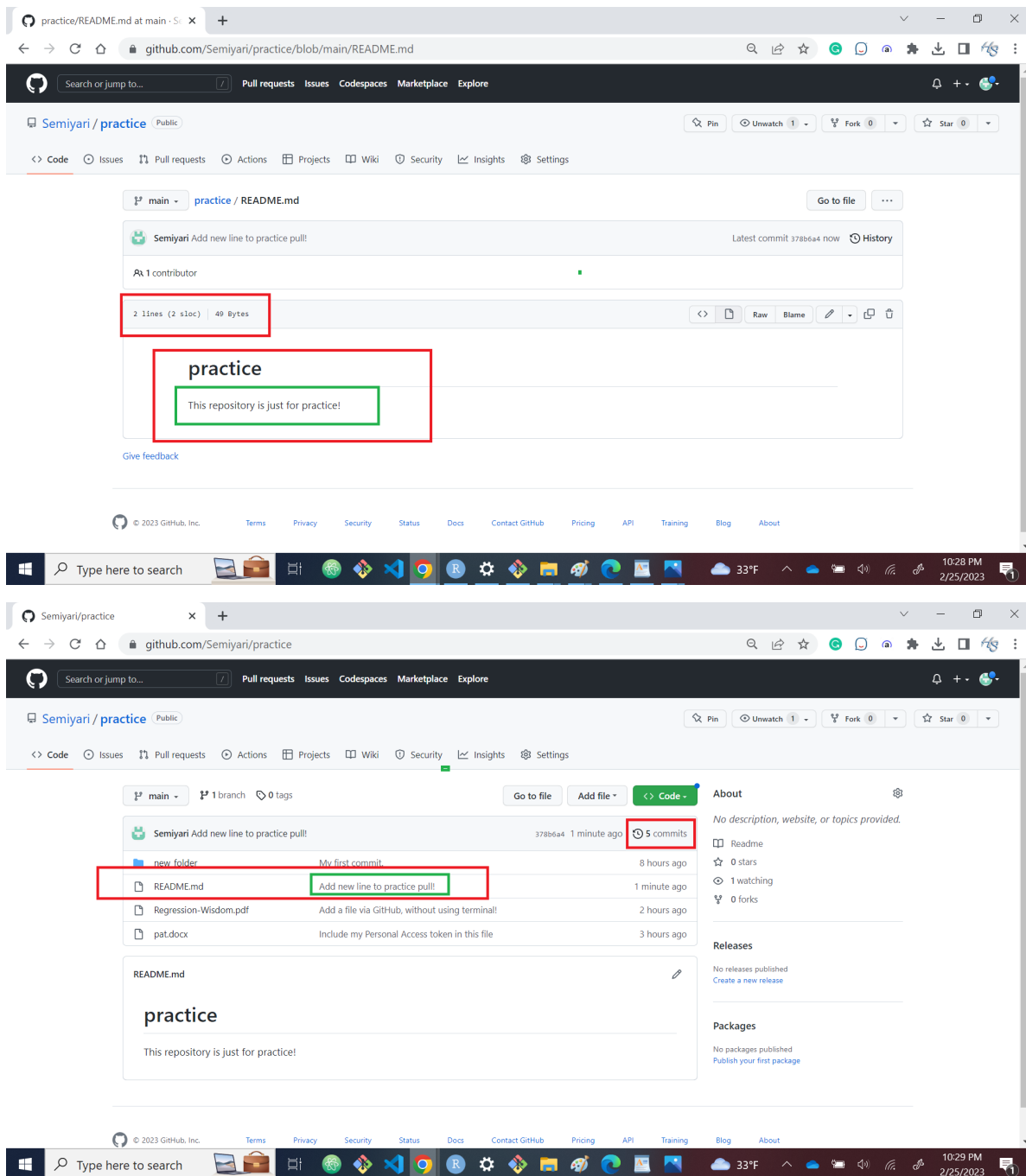
☐ Create a new branch for this commit and start a pull request. [Learn more about pull requests.](#)

Commit changes Cancel

© 2023 GitHub, Inc. Terms Privacy Security Status Docs Contact GitHub Pricing API Training Blog About

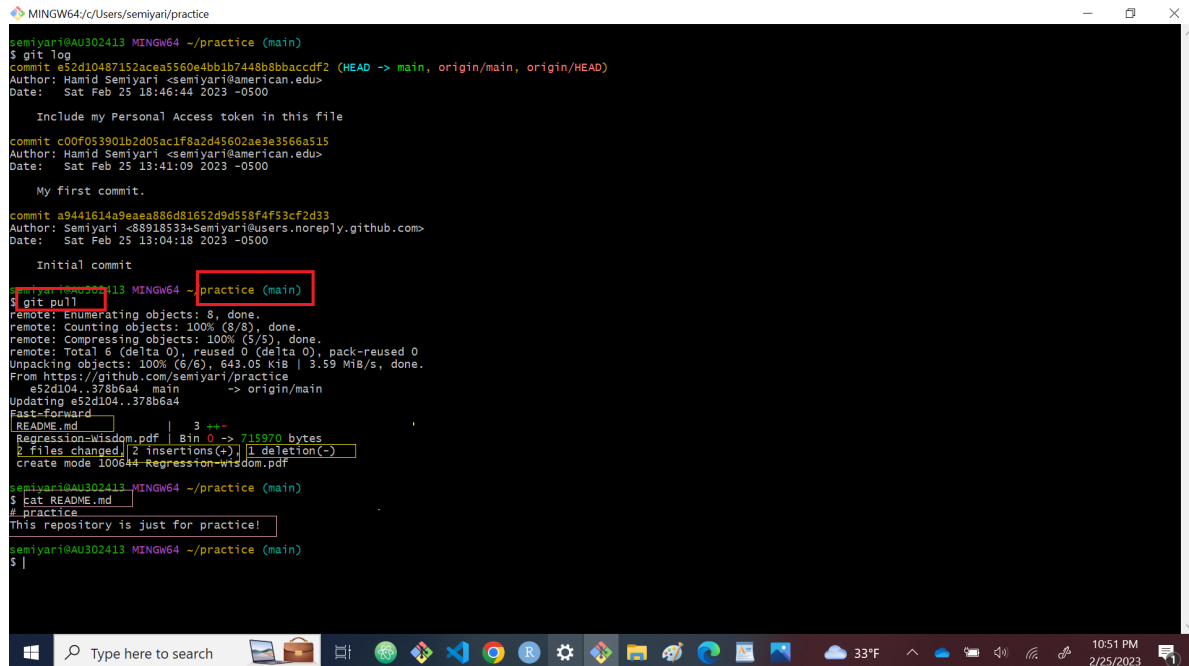
Type here to search

10:28 PM 2/25/2023



Pull A file

- Recently a team member has done some changes to a file and push it into the remote repository and we want to download that file with all of its history.
- Step 1. Open Terminal make sure you are in correct directory. I want to be on **practice**. If I am not there, then I just type `cd ~/practice`.
- Step 2. Type `git pull`. You are done, but if you want to check your work, go to the next step.



```
MINGW64~/c/Users/semiari/practice
semiari@AU302413 MINGW64 ~/practice (main)
$ git log
commit e52d10487152acea5560e4bb1b7448b8bbaccdf2 (HEAD -> main, origin/main, origin/HEAD)
Author: Hamid Semiari <semiari@american.edu>
Date: Sat Feb 25 18:46:44 2023 -0500

    Include my Personal Access token in this file

commit c00f053901b2d05ac1f8a2d45602ae3e3566a515
Author: Hamid Semiari <semiari@american.edu>
Date: Sat Feb 25 13:41:09 2023 -0500

    My first commit.

Initial commit
commit a9441614a9eaea886d81652d9d558f4f53cf2d33
Author: Semiari <88918533+Semiari@users.noreply.github.com>
Date: Sat Feb 25 13:04:18 2023 -0500

    Initial commit

semiari@AU302413 MINGW64 ~/practice (main)
$ git pull
remote: Enumerating objects: 8, done.
remote: Counting objects: 100% (8/8), done.
remote: Compressing objects: 100% (5/5), done.
remote: Total 6 (delta 0); reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (6/6), 643.05 KiB | 3.59 MiB/s, done.
From https://github.com/semiari/practice
e52d104..378b6a4 main -> origin/main
Updating e52d104..378b6a4
Fast-forward
 README.md | 3 ++-
 Regression-Wisdom.pdf | Bin 0 -> 715970 bytes
 2 files changed, 2 insertions(+), 1 deletion(-)
 create mode 100644 Regression-Wisdom.pdf
semiari@AU302413 MINGW64 ~/practice (main)
$ cat README.md
# practice
This repository is just for practice!
semiari@AU302413 MINGW64 ~/practice (main)
$
```

- Step 3. Type `cat README.md` to see the content of the file.

```
MINGW64/C:/Users/semiylari/practice
Unpacking objects: 100% (6/6), 643.05 KiB | 3.59 MiB/s, done.
From https://github.com/semiylari/practice
e52d104..378b6a4  main    -> origin/main
Updating e52d104..378b6a4
Fast-forward
 README.md | 3 ++-
 Regression-wisdom.pdf | Bin 0 -> 715970 bytes
 2 files changed, 2 insertions(+), 1 deletion(-)
 create mode 100644 Regression-wisdom.pdf

semiylari@AU302413 MINGW64 ~/practice (main)
$ cat README.md
# practice
This repository is just for practice!

semiylari@AU302413 MINGW64 ~/practice (main)
$ git log
commit 378b6a4b6ad1ef50ae9758d96b0a1637169685d9 (HEAD -> main, origin/main, origin/HEAD)
Author: Semiylari <88918533+Semiylari@users.noreply.github.com>
Date: Sat Feb 25 22:28:20 2023 -0500

    Add new line to practice pull!

commit e329dc362353edc65ccbf9879e6f7aa830d96c08
Author: Semiylari <88918533+Semiylari@users.noreply.github.com>
Date: Sat Feb 25 20:27:39 2023 -0500

    Add a file via GitHub, without using terminal!

commit 6f2d10487152acea5560e4bb1b7448b8bbaccdf2
Author: Hamid Semiylari <semiylari@american.edu>
Date: Sat Feb 25 18:46:44 2023 -0500

    Include my Personal Access token in this file

commit c00f053901b2d05ac1f8a2d45602ae3e3566a515
Author: Hamid Semiylari <semiylari@american.edu>
Date: Sat Feb 25 13:41:09 2023 -0500

    My first commit.

commit a9441614a9eaea886d81652d9d558f4f53cf2d33
Author: Semiylari <88918533+Semiylari@users.noreply.github.com>
Date: Sat Feb 25 13:04:18 2023 -0500

    Initial commit

semiylari@AU302413 MINGW64 ~/practice (main)
$
```

What is a fork?

- It means copying a repository. But it is different than “cloning”.
 - When you fork a Repository, you create a *copy* of the original repository (upstream repository) but the repository remains *on your GitHub account*.
 - when you clone a repository, the repository is *copied on to your local machine* with the help of Git. ## How to Fork a Repository?
- There is no git fork command, if you’re working with a standard Git installation.
- You can fork any repository by clicking the fork button in the upper right corner of a repo page.

Screenshot of the GitHub interface showing the process of forking a repository.

Top Screenshot (Repository View):

- Repository: Semiyari/practice
- Buttons: Pin, Unwatch (1), Fork (0), Star (0)
- Dropdown menu for Fork: "Your existing forks" (You don't have any forks of this repository), "Create a new fork" (highlighted in red)
- Repository content: README.md, new_folder, Regression-Wisdom.pdf, pat.docx
- README content: "practice", "This repository is just for practice!"

Bottom Screenshot (Forking Process):

- Page: Fork Semiyari/practice
- Section: Create a new fork
- Text: "A fork is a copy of a repository. Forking a repository allows you to freely experiment with changes without affecting the original project."
- Owner: AU-DATA-413-613
- Repository name: practice
- Description (optional):
- Copy the main branch only (checked)
- Text: "You are creating a fork in the AU-DATA-413-613 organization."
- Button: Create fork (highlighted in red)

Branching

- Git doesn't store data as a series of changesets or differences, but instead as a series of snapshots.

When you make a commit, Git stores a commit object that contains a pointer to the snapshot of the content you staged. This object also contains the author's name and email address, the message that you typed, and pointers to the commit or commits that directly came before this commit (its parent or parents): zero parents for the initial commit, one parent for a normal commit, and multiple parents for a commit that results from a merge of two or more branches.

- Branches allow you to work on different parts of a project without impacting the main branch.
- When the work is complete, a branch can be merged with the main project.
- You can even switch between branches and work on different projects without them interfering with each other.
 - We are working on our local repository and we do not want to mess with the main project. Let's add something to our `new_folder\hw1.txt`.
 - Let's see what we have inside of the file `hw1.txt`

```
cat new_folder/hw1.txt
```

- Create a branch

```
git branch branch1
```

- Let us check if the branch has created

```
git branch
```

It returns

```
branch_1 * main
```

but the `*` beside `main` specifies that we are currently on that branch.

The prompt says, I am in main branch:

```
semiyari@AU302413 MINGW64 ~/practice (main)
```

- `checkout` is the command used to check out a branch. Moving us from the current branch, to the one specified at the end of the command:

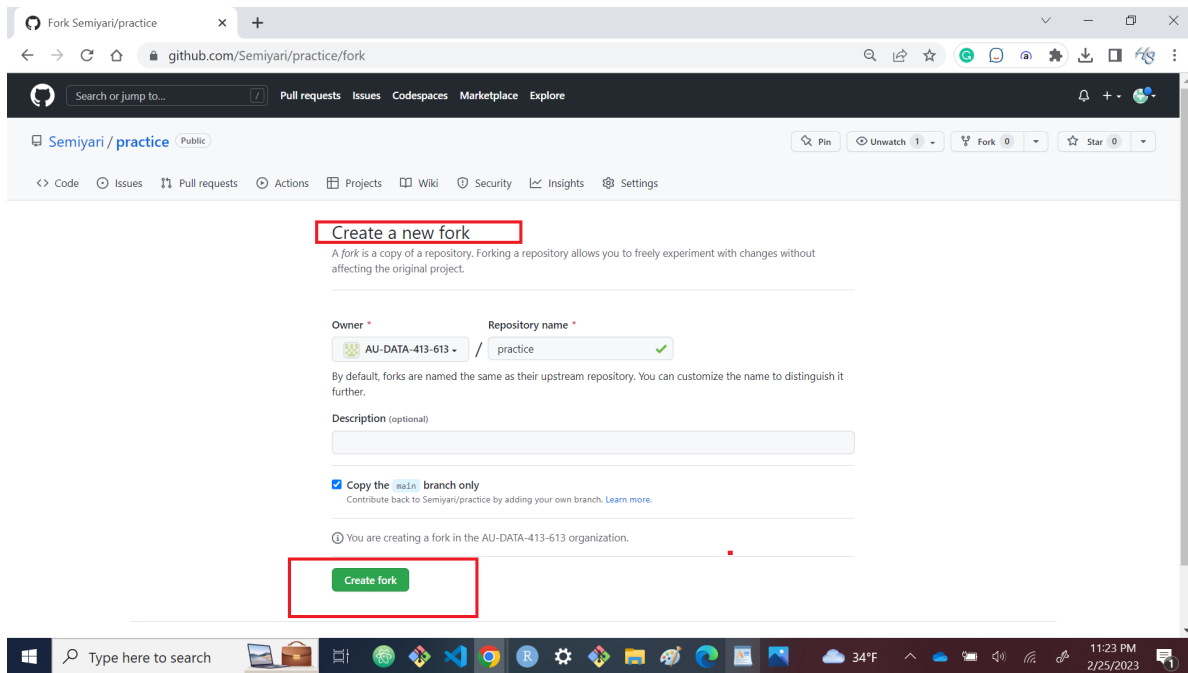
```
git checkout branch_1
```

It returns the following line

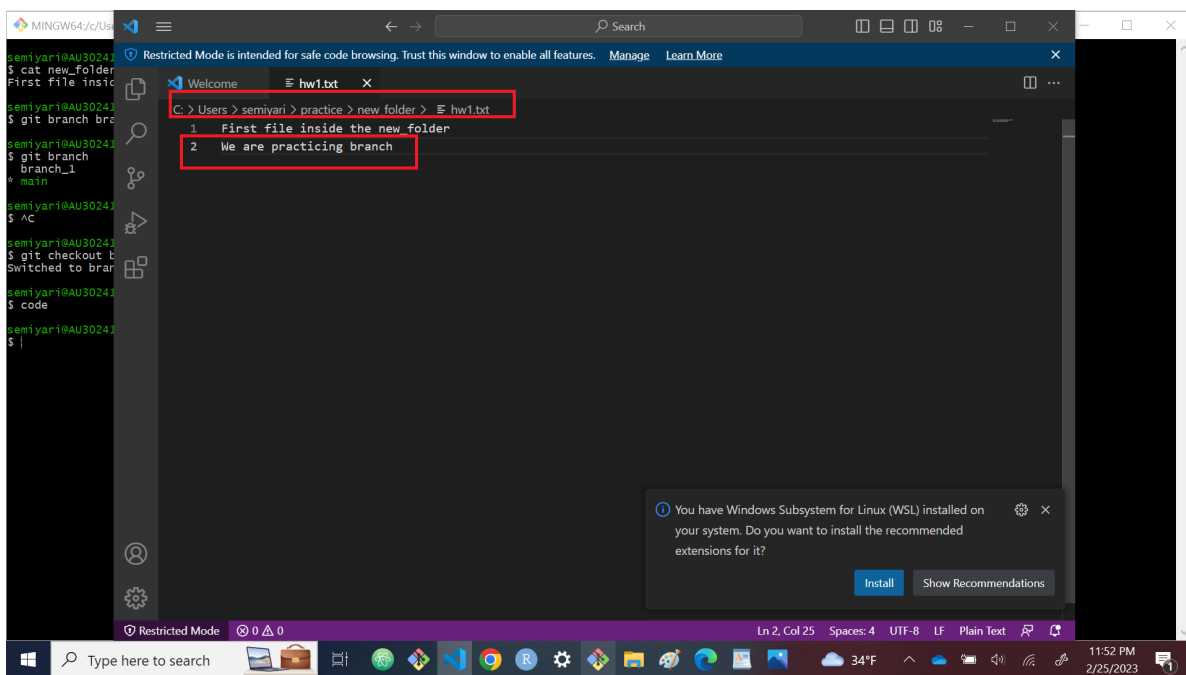
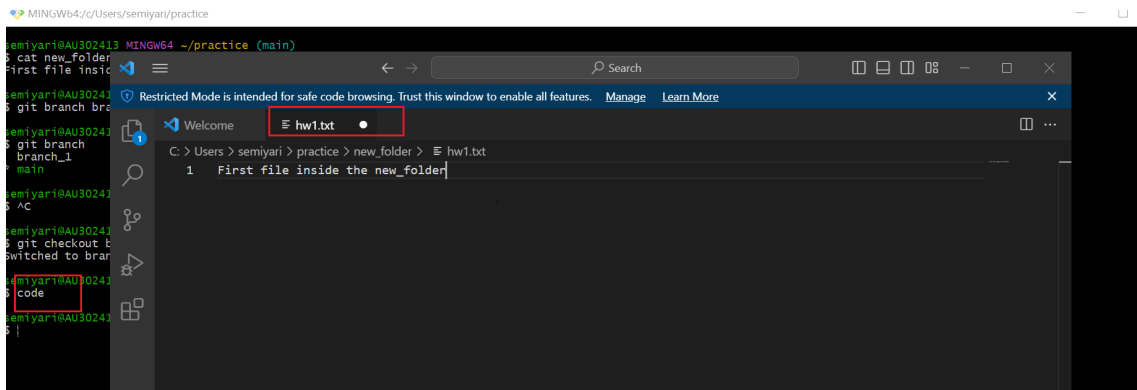
Switched to branch 'branch_1'

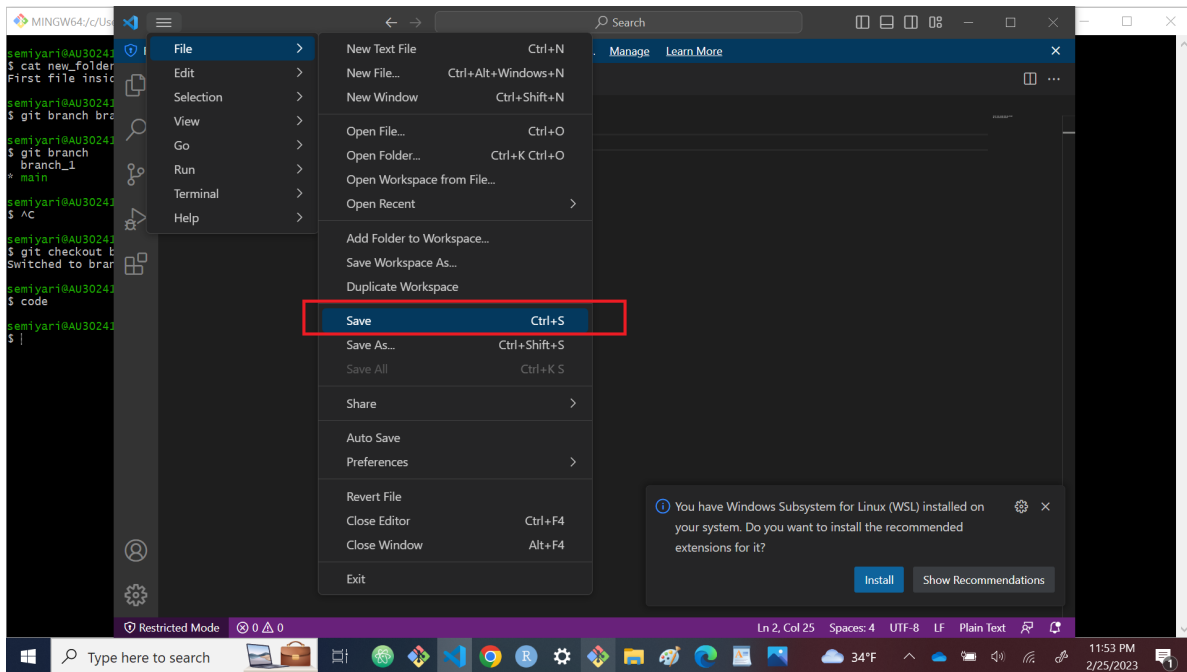
Now the prompt says

semiyari@AU302413 MINGW64 ~/practice (branch_1)

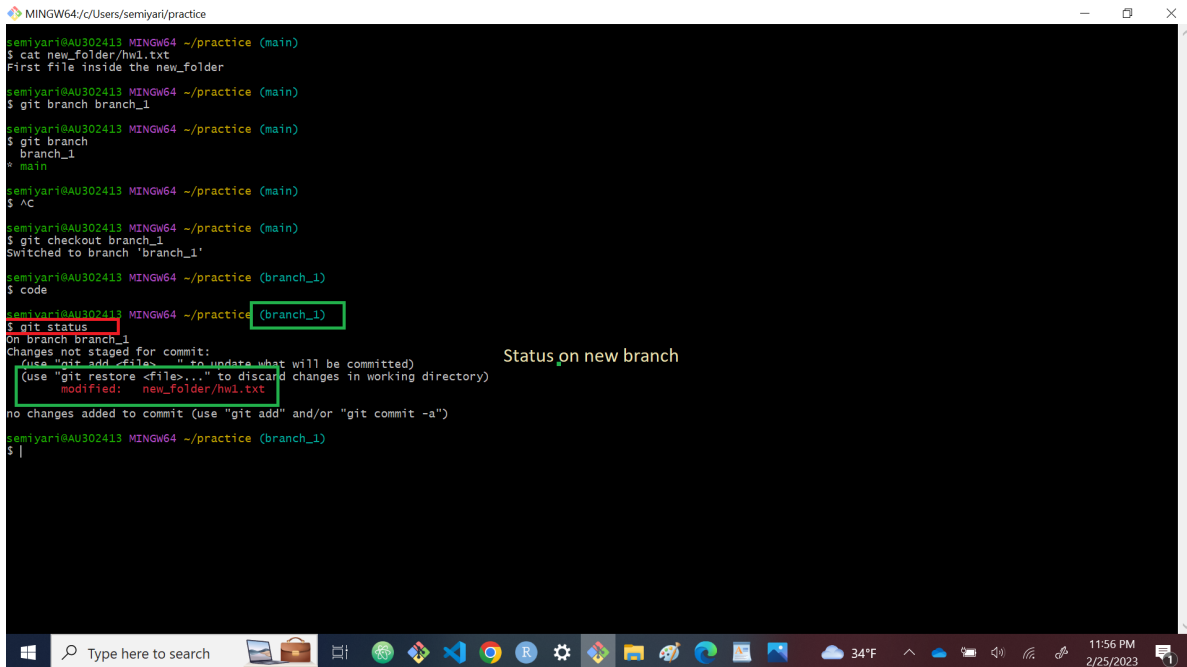


- Open the text editor and make some changes to the file `hw1.txt`





- Check the status of this current branch



- we need to add the file to the Staging area for this branch and commit the file. At the end we get the status of the branch.

```
MINGW64/c/Users/semiyan/practice
First file inside the new_folder

semiyan@AU302413 MINGW64 ~/practice (main)
$ git branch branch_1

semiyan@AU302413 MINGW64 ~/practice (main)
$ git branch
  branch_1
* main

semiyan@AU302413 MINGW64 ~/practice (main)
$ ^C

semiyan@AU302413 MINGW64 ~/practice (main)
$ git checkout branch_1
Switched to branch 'branch_1'

semiyan@AU302413 MINGW64 ~/practice (branch_1)
$ code

semiyan@AU302413 MINGW64 ~/practice (branch_1)
$ git status
On branch branch_1
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore --staged <file>..." to discard changes in working directory)
    modified:   new_folder/hw1.txt

no changes added to commit (use "git add" and/or "git commit -a")

semiyan@AU302413 MINGW64 ~/practice (branch_1)
$ git add --all
warning: in the working copy of 'new_folder/hw1.txt', LF will be replaced by CRLF the next time Git touches it

semiyan@AU302413 MINGW64 ~/practice (branch_1)
$ git status
On branch branch_1
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    modified:   new_folder/hw1.txt

semiyan@AU302413 MINGW64 ~/practice (branch_1)
$ git commit -m "Add new line to hw1.txt"
[branch_1 0fd4e2e] Add new line to hw1.txt
1 file changed, 1 insertion(+)

semiyan@AU302413 MINGW64 ~/practice (branch_1)
$ |
```

- Let's add a new file in this branch file2.txt

```
MINGW64/c/Users/semiyan/practice
File Edit View Settings Tools Help
echo New File > file2.html

semiyan@AU302413 MINGW64 ~/practice (branch_1)
$ git status
On branch branch_1
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    file2.html

nothing added to commit but untracked files present (use "git add" to track)

semiyan@AU302413 MINGW64 ~/practice (branch_1)
$ git add .html
fatal: pathspec '.html' did not match any files

semiyan@AU302413 MINGW64 ~/practice (branch_1)
$ git add file2.html
warning: in the working copy of 'file2.html', LF will be replaced by CRLF the next time Git touches it

semiyan@AU302413 MINGW64 ~/practice (branch_1)
$ git status
On branch branch_1
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   file2.html

semiyan@AU302413 MINGW64 ~/practice (branch_1)
$ git commit -m "Add new file"
[branch_1 2c4f491] Add new file
1 file changed, 1 insertion(+)
create mode 100644 file2.html

semiyan@AU302413 MINGW64 ~/practice (branch_1)
$ ls
README.md  Regression-Wisdom.pdf  file2.html  new_folder/  pat.docx

semiyan@AU302413 MINGW64 ~/practice (branch_1)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

semiyan@AU302413 MINGW64 ~/practice (main)
$ ls
README.md  Regression-Wisdom.pdf  new_folder/  pat.docx

semiyan@AU302413 MINGW64 ~/practice (main)
$ |
```

Switch Between Branches

We are currently on the branch `branch_1`. Let's list the files in the current directory:

```
ls
```

It returns

README.md Regression-Wisdom.pdf file2.html new_folder/ pat.docx

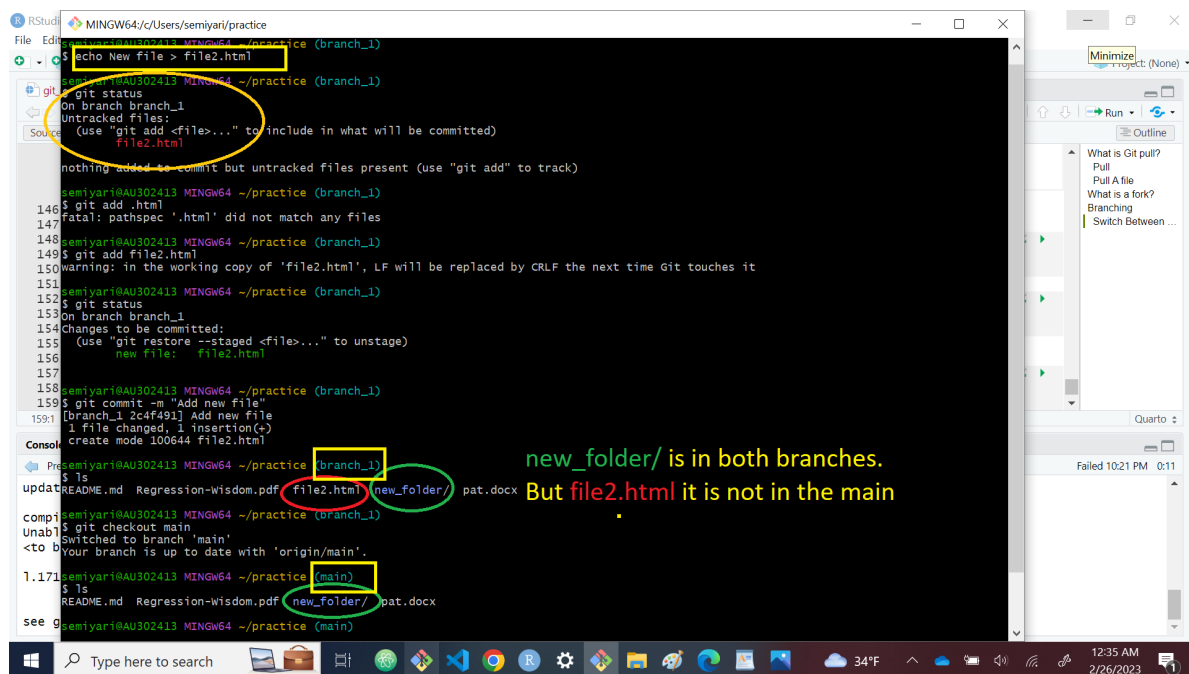
- Now, let's see what happens when we change branch to main

```
git checkout main
```

```
ls
```

It returns

README.md Regression-Wisdom.pdf new_folder/ pat.docx



Merge branches We are done with our work in our new branch and we now want to merge to the main so the changes that we have done to the files can be accessible to everyone. - We want to merge to main, so we need to make sure we are in the main branch. Use `git branch` to see if you are where you should be. Then use the following

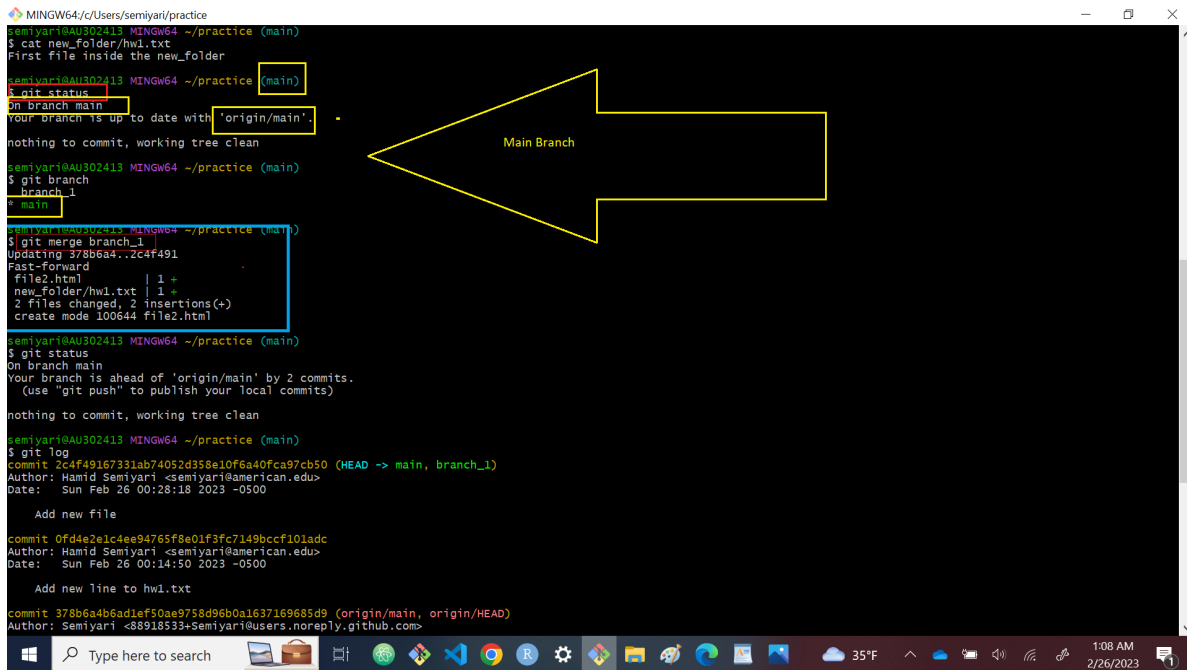
```
git merge <branch_that_need_to_be_merged>
```

In this case

```
git merge branch_1
```

- Check the status

```
git status
```



```
MINGW64/c/Users/semiyari/practice
semiyari@AU302413 MINGW64 ~/practice (main)
$ cat new_folder/hw1.txt
First file inside the new_folder

semiyari@AU302413 MINGW64 ~/practice (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.
nothing to commit, working tree clean

semiyari@AU302413 MINGW64 ~/practice (main)
$ git branch
* main

semiyari@AU302413 MINGW64 ~/practice (main)
$ git merge branch_1
Updating 378b6a4..2c4f491
Fast-forward
 file2.html | 1 +
 new_folder/hw1.txt | 1 +
 2 files changed, 2 insertions(+)
 create mode 100644 file2.html

semiyari@AU302413 MINGW64 ~/practice (main)
$ git status
On branch main
Your branch is ahead of 'origin/main' by 2 commits.
  (Use 'git push' to publish your local commits)
nothing to commit, working tree clean

semiyari@AU302413 MINGW64 ~/practice (main)
$ git log
commit 2c4f49167331ab74052d358e10f6a40fca97cb50 (HEAD -> main, branch_1)
Author: Hamid Semiyari <semiyari@american.edu>
Date: Sun Feb 26 00:28:18 2023 -0500

    Add new file

commit 0fd4e2e1c4ee94765f8e01f3fc7149bccf101adc
Author: Hamid Semiyari <semiyari@american.edu>
Date: Sun Feb 26 00:14:50 2023 -0500

    Add new line to hw1.txt

commit 378b6a4b6ad1ef50ae9758d96b0a1637169685d9 (origin/main, origin/HEAD)
Author: Semiyari <88918533+Semiyari@users.noreply.github.com>
```

- Use `git log` to see if you have everything.

```
nothing to commit, working tree clean
semi@AU302413 MINGW64 ~/practice (main)
$ git log
commit 2c4f49167331ab7405d358e10f6a40fca97cb50 (HEAD -> main, branch_1)
Author: Hamid Semi@american.edu
Date: Sun Feb 26 00:28:18 2023 -0500

    Add new file

commit 0f4e2e1c4ee94765f8e01f3f7149bccf101adc
Author: Hamid Semi@american.edu
Date: Sun Feb 26 00:14:50 2023 -0500

    Add new line to hw1.txt

commit 378b6a4b6ad1ef50ae9758d96b0a1637169685d9 (origin/main, origin/HEAD)
Author: Semi@users.noreply.github.com
Date: Sat Feb 25 22:28:20 2023 -0500

    Add new line to practice pull!

commit e329dc362353edc65ccb9879e6f7aa830d96c08
Author: Semi@users.noreply.github.com
Date: Sat Feb 25 20:27:39 2023 -0500

    Add a file via GitHub, without using terminal!

commit e52d10487152acea5560e4bb1b7448bbaccdf2
Author: Hamid Semi@american.edu
Date: Sat Feb 25 18:46:44 2023 -0500

    Include my Personal Access token in this file

commit c00f053901b2d05ac1f8a2d45602ae3e3566a515
Author: Hamid Semi@american.edu
Date: Sat Feb 25 13:41:09 2023 -0500

    My first commit.

commit a9441614a9eaea886d81652d9d558f4f53cf2d33
Author: Semi@users.noreply.github.com
Date: Sat Feb 25 13:04:18 2023 -0500

    Initial commit
semi@AU302413 MINGW64 ~/practice (main)
$
```

- Use `git push` to publish your local commits

Delete a Branch

You had a branch to work on side, You are done and you do not need it any longer

```
git branch -d branch_1
```

You may then check by following code to see if the branch has been deleted.

```
git branch
```



```
MINGW64/c/Users/semiari/practice

commit 378b6a4b6ad1ef50ae9758d96b0a1637169685d9 (origin/main, origin/HEAD)
Author: Semiari <88918533+Semiari@users.noreply.github.com>
Date: Sat Feb 25 22:28:20 2023 -0500

    Add new line to practice pull!

commit e329dc362353edc65c3cbf9879e6f7a8830d96c08
Author: Semiari <88918533+Semiari@users.noreply.github.com>
Date: Sat Feb 25 20:27:39 2023 -0500

    Add a file via GitHub, without using terminal!

commit e52d10487152acea5560e4bb1b7448b8bbaccdf2
Author: Hamid Semiari <semiari@american.edu>
Date: Sat Feb 25 18:46:44 2023 -0500

    Include my Personal Access token in this file

commit c00f053901b2d05ac1f8a2d45602ae3e3566a515
Author: Hamid Semiari <semiari@american.edu>
Date: Sat Feb 25 13:41:09 2023 -0500

    My first commit.

commit a9441614a9e8ea886d81652d9d558f4f53cf2d93
Author: Semiari <88918533+Semiari@users.noreply.github.com>
Date: Sat Feb 25 13:04:18 2023 -0500

    Initial commit

semiari@AU302413 MINGW64 ~/practice (main)
$ git branch -d branch_1
Deleted branch branch_1 (was 2c4f491).

semiari@AU302413 MINGW64 ~/practice (main)
$ git branch
* main

semiari@AU302413 MINGW64 ~/practice (main)
$
$ You had a branch to work on side, you are done and you do not need it any longer
Error: unexpected symbol in "You had"
> {bash, eval=FALSE}
git checkout main
```