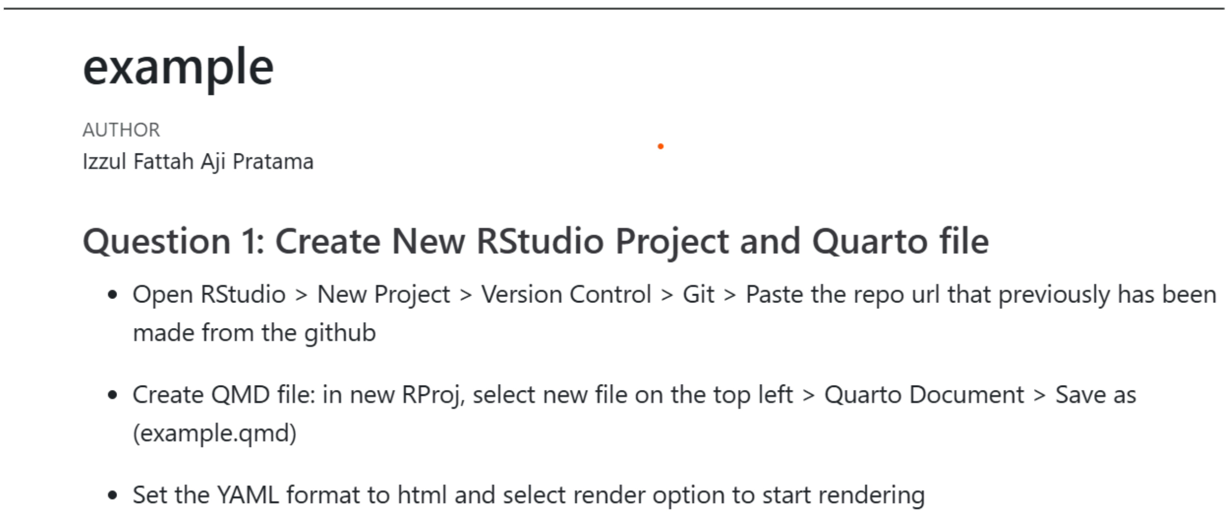
Assignment 2 RCP

AUTHOR
Izzul Fattah Aji Pratama

Question 1: Create New RStudio Project and Quarto file

- Open RStudio > New Project > Version Control > Git > Paste the repository URL that was previously created on GitHub.
- To create a Quarto (QMD) file, in the new RStudio Project, select **File > New File > Quarto Document > Save as** (example.qmd)
- Set the YAML format to html and select render option to start rendering
- 

example

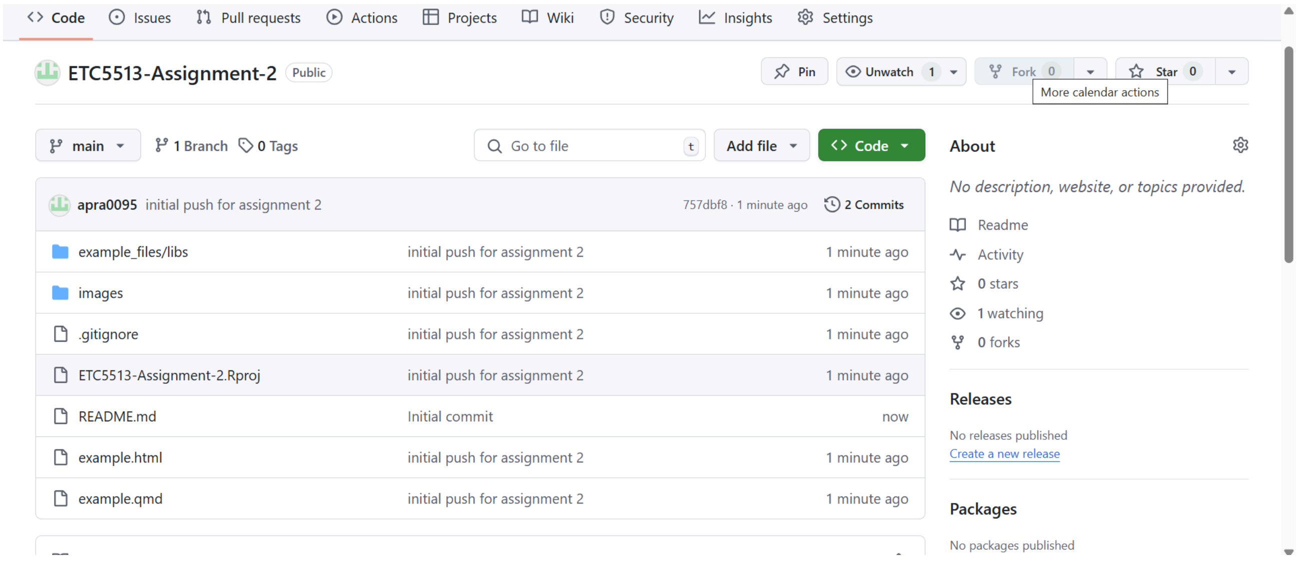
AUTHOR
Izzul Fattah Aji Pratama

Question 1: Create New RStudio Project and Quarto file

- Open RStudio > New Project > Version Control > Git > Paste the repo url that previously has been made from the github
- Create QMD file: in new RProj, select new file on the top left > Quarto Document > Save as (example.qmd)
- Set the YAML format to html and select render option to start rendering

Question 2: Initialize Git and Push to the Repo

- Previously we already created an R project. However, to make this project is reproducible we must place our file in a repository so the others could also see and work on this
- Open the RStudio terminal and initialize a Git repository by typing `git init`
- add all files by typing `git add .`
- Commit the file using `git commit -m "initial commit"`
- Up until this point, we could know that Git actually could work without GitHub to track changes of our file. However, since this demo is to show a reproducible practices, we need to set up a repo in GitHub then copy the SSH code and type `git remote add origin <paste the repo url>` to link the local repo to GitHub repo.
- After we connect the repo, we could see that the default name of the main branch is “master” . Because we already familiar with the name of “main” lets just change it by typing `git branch -M main`
- the last step is to push our changes by typing `git push -u origin main`



ETC5513-Assignment-2

apra0095 initial push for assignment 2		757dbf8 · 1 minute ago · 2 Commits
example_files/libs	initial push for assignment 2	1 minute ago
images	initial push for assignment 2	1 minute ago
.gitignore	initial push for assignment 2	1 minute ago
ETC5513-Assignment-2.Rproj	initial push for assignment 2	1 minute ago
README.md	Initial commit	now
example.html	initial push for assignment 2	1 minute ago
example.qmd	initial push for assignment 2	1 minute ago

About

No description, website, or topics provided.

Readme
Activity
0 stars
1 watching
0 forks

Releases

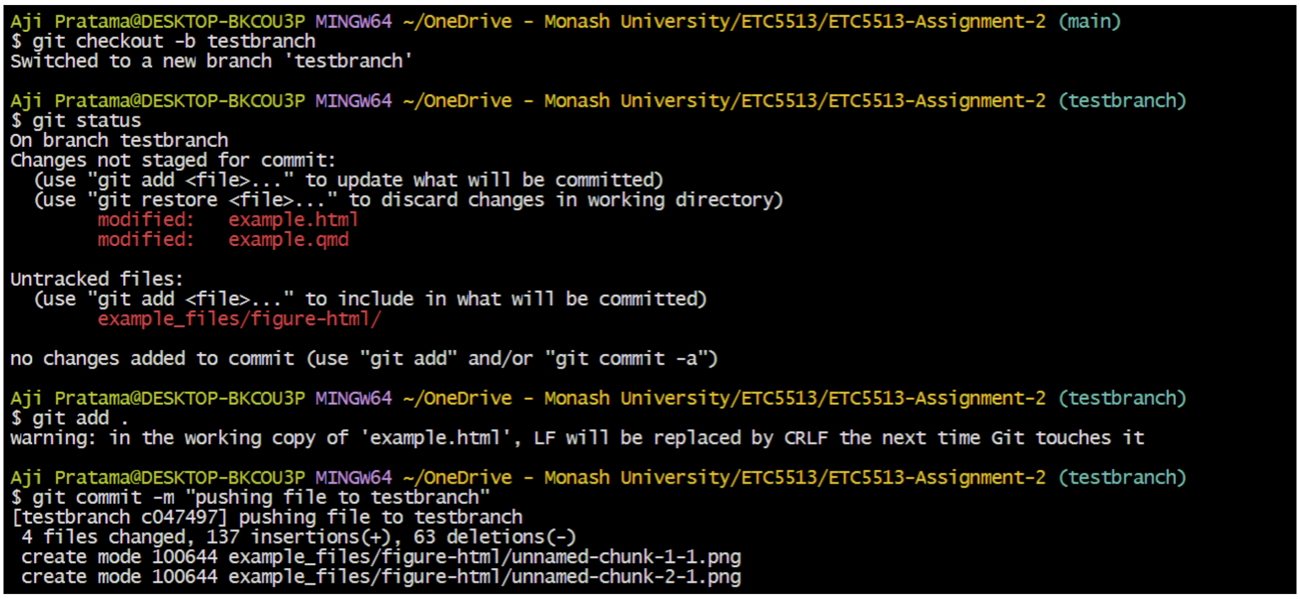
No releases published
Create a new release

Packages

No packages published

Question 3: Create Branch, Modify File, and Add the File

- Type `git checkout -b testbranch` in the terminal to create a new branch called `testbranch` and switch to it.
- Modify the file to test. For example, I am modifying the file by typing the Question 3 command.
- Type `git status` in the terminal to check whether Git is tracking the changes. Then type `git add .` to stage the file and `git commit -m "pushing file to testbranch"`.



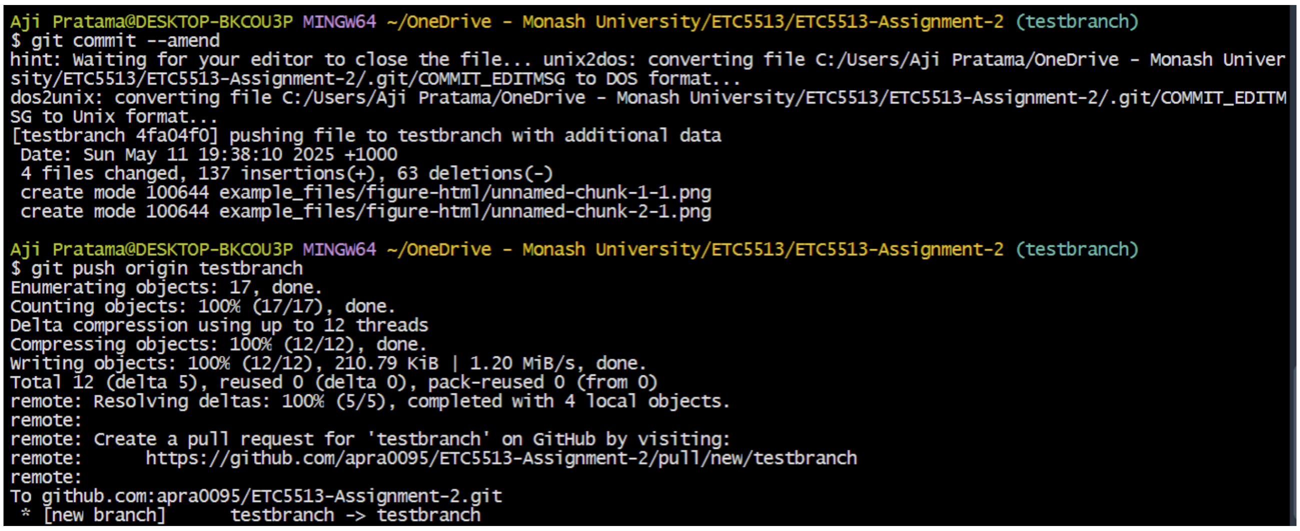
```
Aji Pratama@DESKTOP-BKCOU3P MINGW64 ~/OneDrive - Monash University/ETC5513/ETC5513-Assignment-2 (main)
$ git checkout -b testbranch
Switched to a new branch 'testbranch'

Aji Pratama@DESKTOP-BKCOU3P MINGW64 ~/OneDrive - Monash University/ETC5513/ETC5513-Assignment-2 (testbranch)
$ git status
On branch testbranch
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   example.html
        modified:   example.qmd

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        example_files/figure-html/

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

Aji Pratama@DESKTOP-BKCOU3P MINGW64 ~/OneDrive - Monash University/ETC5513/ETC5513-Assignment-2 (testbranch)
$ git add .
warning: in the working copy of 'example.html', LF will be replaced by CRLF the next time Git touches it

Aji Pratama@DESKTOP-BKCOU3P MINGW64 ~/OneDrive - Monash University/ETC5513/ETC5513-Assignment-2 (testbranch)
$ git commit -m "pushing file to testbranch"
[testbranch c047497] pushing file to testbranch
 4 files changed, 137 insertions(+), 63 deletions(-)
 create mode 100644 example_files/figure-html/unnamed-chunk-1-1.png
 create mode 100644 example_files/figure-html/unnamed-chunk-2-1.png
```

Question 4: Add Data Folder and Amend the Previous Commit

- Previously we already make some changes in the qmd file and committed to the `testbranch` . However, we also want to add another file called “data” to our repository in that specific commit.
- To do that, first, Create a `data` folder in the working directory and fill it with the Assignment 1 data.
- after that, we could amend the previous commit by typing this command `git commit --amend` and Git will open a text editor stating our last commit and the untracked files. All we need to do is to adjust the commit message following what we have been added, in this case data file, close and save the file.

![](images/clipboard-51895079.png)

- After that, we just need to push the changes to the `testbranch` by typing `git push origin testbranch` . This push is already included the data file that we recently added by doing a commit amend from previous commit.



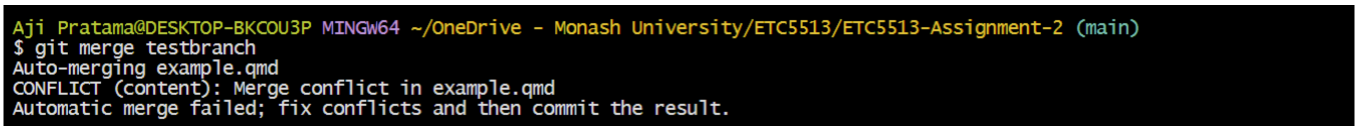
```
Aji Pratama@DESKTOP-BKCOU3P MINGW64 ~/OneDrive - Monash University/ETC5513/ETC5513-Assignment-2 (testbranch)
$ git commit --amend
hint: Waiting for your editor to close the file... unix2dos: converting file C:/Users/Aji Pratama/OneDrive - Monash Univer
sity/ETC5513/ETC5513-Assignment-2/.git/COMMIT_EDITMSG to DOS format...
dos2unix: converting file C:/Users/Aji Pratama/OneDrive - Monash University/ETC5513/ETC5513-Assignment-2/.git/COMMIT_EDITM
SG to Unix format...
[testbranch 4fa04f0] pushing file to testbranch with additional data
 Date: Sun May 11 19:38:10 2025 +1000
 4 files changed, 137 insertions(+), 63 deletions(-)
 create mode 100644 example_files/figure-html/unnamed-chunk-1-1.png
 create mode 100644 example_files/figure-html/unnamed-chunk-2-1.png

Aji Pratama@DESKTOP-BKCOU3P MINGW64 ~/OneDrive - Monash University/ETC5513/ETC5513-Assignment-2 (testbranch)
$ git push origin testbranch
Enumerating objects: 17, done.
Counting objects: 100% (17/17), done.
Delta compression using up to 12 threads
Compressing objects: 100% (12/12), done.
Writing objects: 100% (12/12), 210.79 KiB | 1.20 MiB/s, done.
Total 12 (delta 5), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (5/5), completed with 4 local objects.
remote:
remote: Create a pull request for 'testbranch' on GitHub by visiting:
remote:      https://github.com/apra0095/ETC5513-Assignment-2/pull/new/testbranch
remote:
To github.com:apra0095/ETC5513-Assignment-2.git
 * [new branch]      testbranch -> testbranch
```

Question 5: Modify example.qmd in Main to Prepare Creating Conflict

- Type `git checkout main` in the terminal to switch back to the `main` branch.
- Add some changes in exapmle.qmd on main
- Then push the changes to the `main`. This will create an error in Question 6 because what the `main` branch records right now is only Questions 1, 2, 5, and will continue to 6, whereas `testbranch` records Questions 3 and 4.

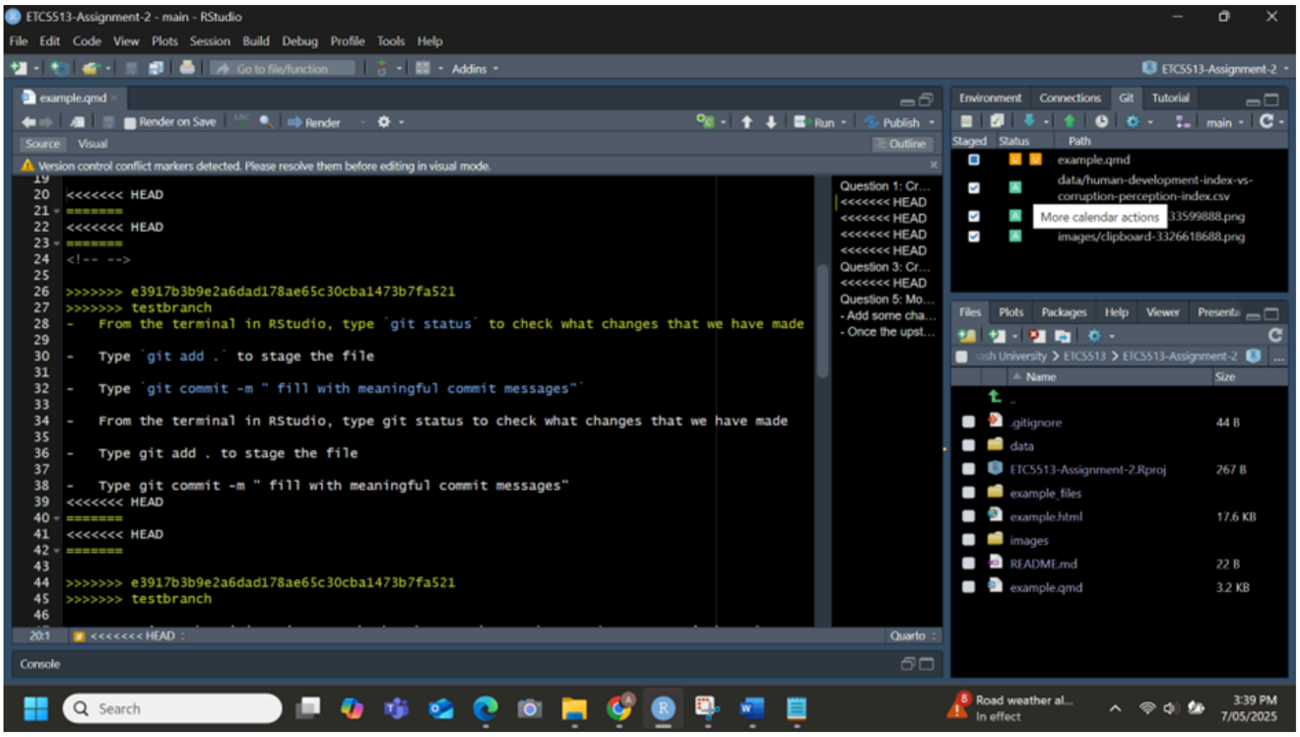
Question 6: Merge and Resolve Conflict

- Now, to simulate a conflict, type `git merge testbranch` to merge the changes from the branch.
- Git will show an error like this:



```
Aji Pratama@DESKTOP-BKCOU3P MINGW64 ~/OneDrive - Monash University/ETC5513/ETC5513-Assignment-2 (main)
$ git merge testbranch
Auto-merging example.qmd
CONFLICT (content): Merge conflict in example.qmd
Automatic merge failed; fix conflicts and then commit the result.
```

- Then, in the **Source** panel, this message will pop up showing `<<<<<<< HEAD` and `=======`. This means Git wants us to clarify which changes we want to keep.



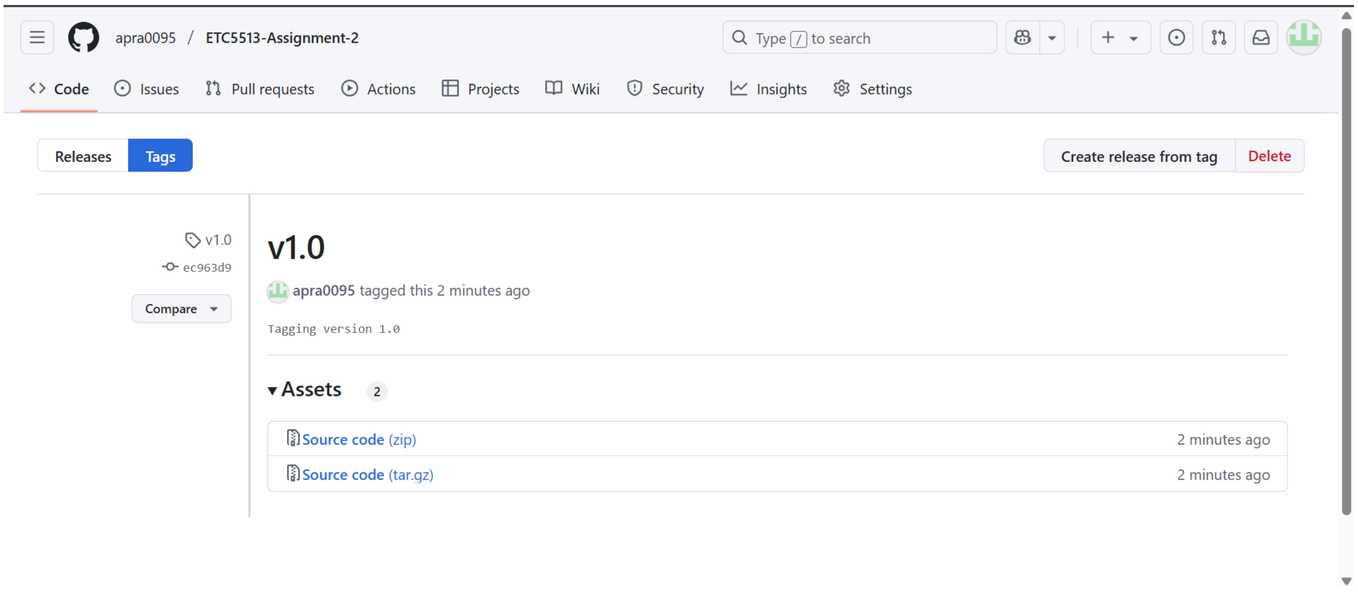
Version control conflict markers detected. Please resolve them before editing in visual mode.

```
<<<<<<< HEAD
=======
<<<<<<< HEAD
=======
<!-- -->

>>>>>>> e3917b3b9e2a6dad178ae65c30cba1473b7fa521
>>>>>>> testbranch
-   From the terminal in RStudio, type `git status` to check what changes that we have made

-   Type `git add .` to stage the file

-   Type `git commit -m " fill with meaningful commit messages"`

-   From the terminal in RStudio, type git status to check what changes that we have made

-   Type git add . to stage the file

-   Type git commit -m " fill with meaningful commit messages"
<<<<<<< HEAD
=======
<<<<<<< HEAD
=======

>>>>>>> e3917b3b9e2a6dad178ae65c30cba1473b7fa521
>>>>>>> testbranch
```

- In this case, I choose to keep all the changes that I have made both in the branch and in `main`, so I just need to delete all of the `<<<<<<< HEAD` and `=======` signs.

Question 7: Create a Commit Using Annotated Tag

- After resolving the conflict in the previous question, we want to push the changes with an annotated tag.
- Type `git tag -a v1.0 -m "Tagging version 1.0"` to create a commit with an annotated tag.
- Push it using `git push origin v1.0`.



apra0095 / ETC5513-Assignment-2

v1.0

ec963d9

apra0095 tagged this 2 minutes ago

Tagging version 1.0

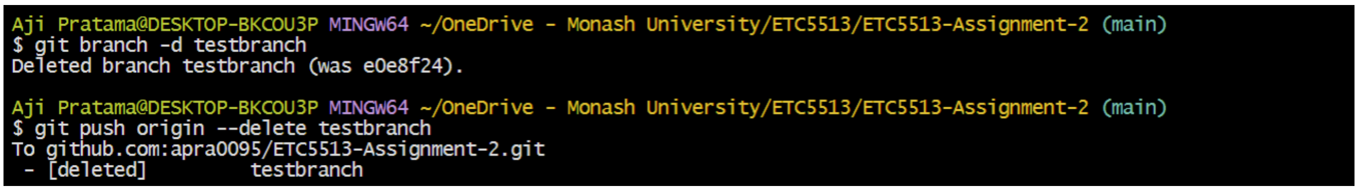
Assets 2

Source code (zip)	2 minutes ago
Source code (tar.gz)	2 minutes ago

Question 8: Delete Testbranch

- To delete the `testbranch` we created earlier, use the command `git branch -d testbranch` to delete the local branch.

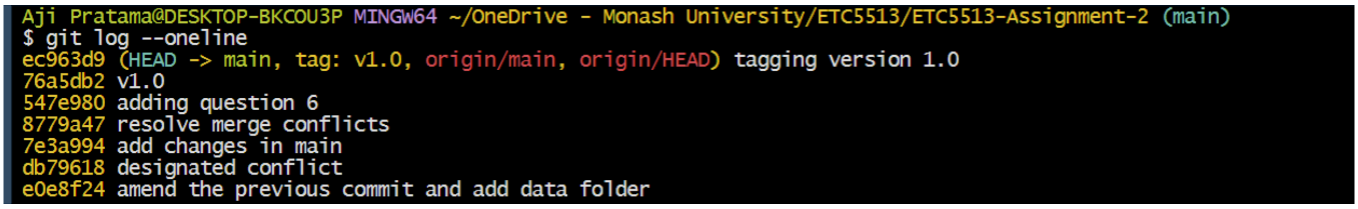
- Then, type `git push origin --delete testbranch` to push the deletion process to GitHub.



```
Aji Pratama@DESKTOP-BKCOU3P MINGW64 ~/OneDrive - Monash University/ETC5513/ETC5513-Assignment-2 (main)
$ git branch -d testbranch
Deleted branch testbranch (was e0e8f24).

Aji Pratama@DESKTOP-BKCOU3P MINGW64 ~/OneDrive - Monash University/ETC5513/ETC5513-Assignment-2 (main)
$ git push origin --delete testbranch
To github.com:apra0095/ETC5513-Assignment-2.git
 - [deleted]         testbranch
```

Question 9: Show Condensed Log

- As we've made many changes to our files, Git has a feature to show the commits we have made so we can track them.
- To do this, type `git log --oneline` in the terminal to see a condensed view of the commit history.



```
Aji Pratama@DESKTOP-BKCOU3P MINGW64 ~/OneDrive - Monash University/ETC5513/ETC5513-Assignment-2 (main)
$ git log --oneline
ec963d9 (HEAD -> main, tag: v1.0, origin/main, origin/HEAD) tagging version 1.0
76a5db2 v1.0
547e980 adding question 6
8779a47 resolve merge conflicts
7e3a994 add changes in main
db79618 designated conflict
e0e8f24 amend the previous commit and add data folder
```

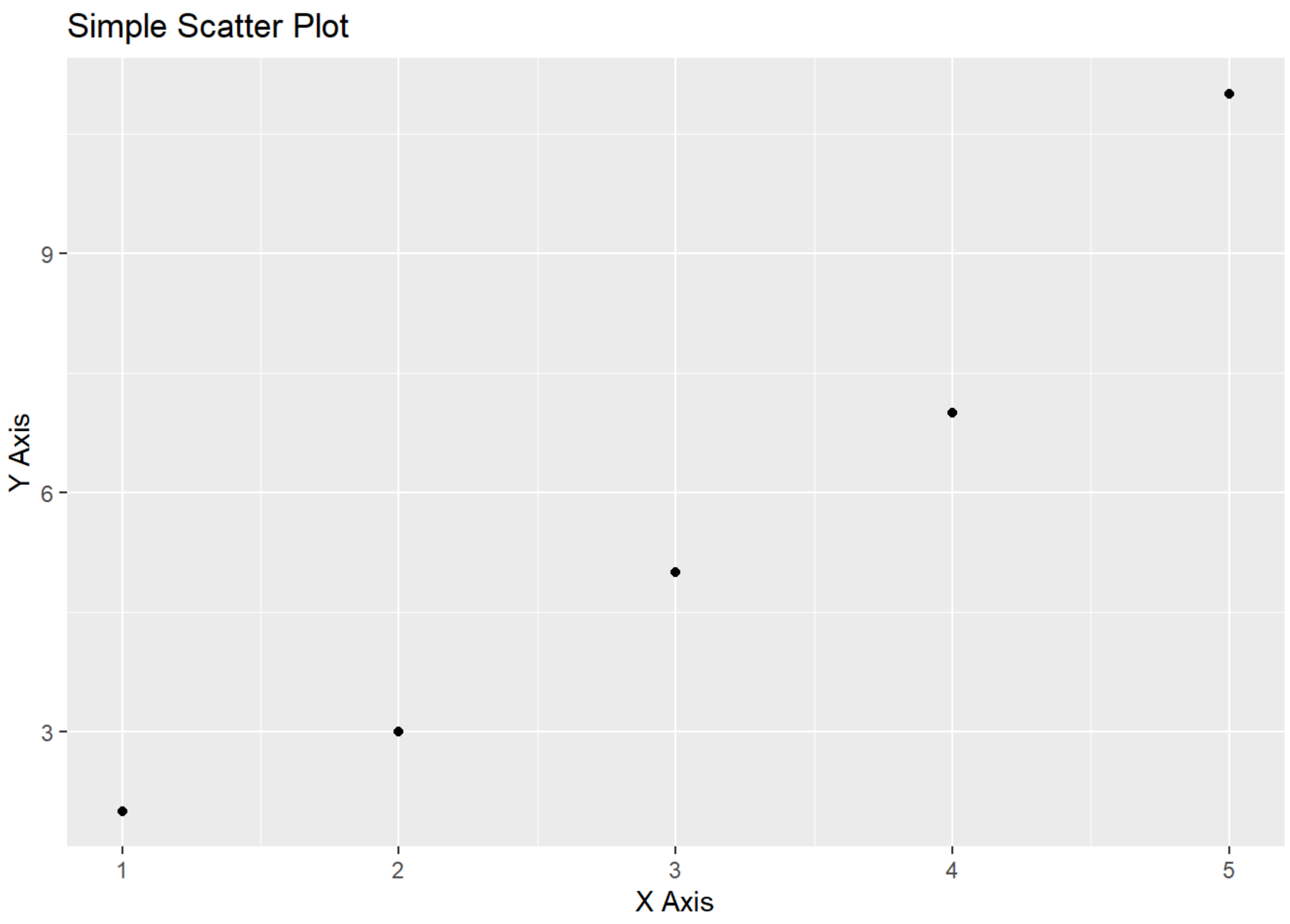
Question 10: Undo Commit Without Losing Changes

- Git also has a feature to undo a commit while preserving the changes you made. It moves the commit pointer to a previous state but keeps the changes from that commit in the staging area (index). This allows you to edit or add new changes before committing again.

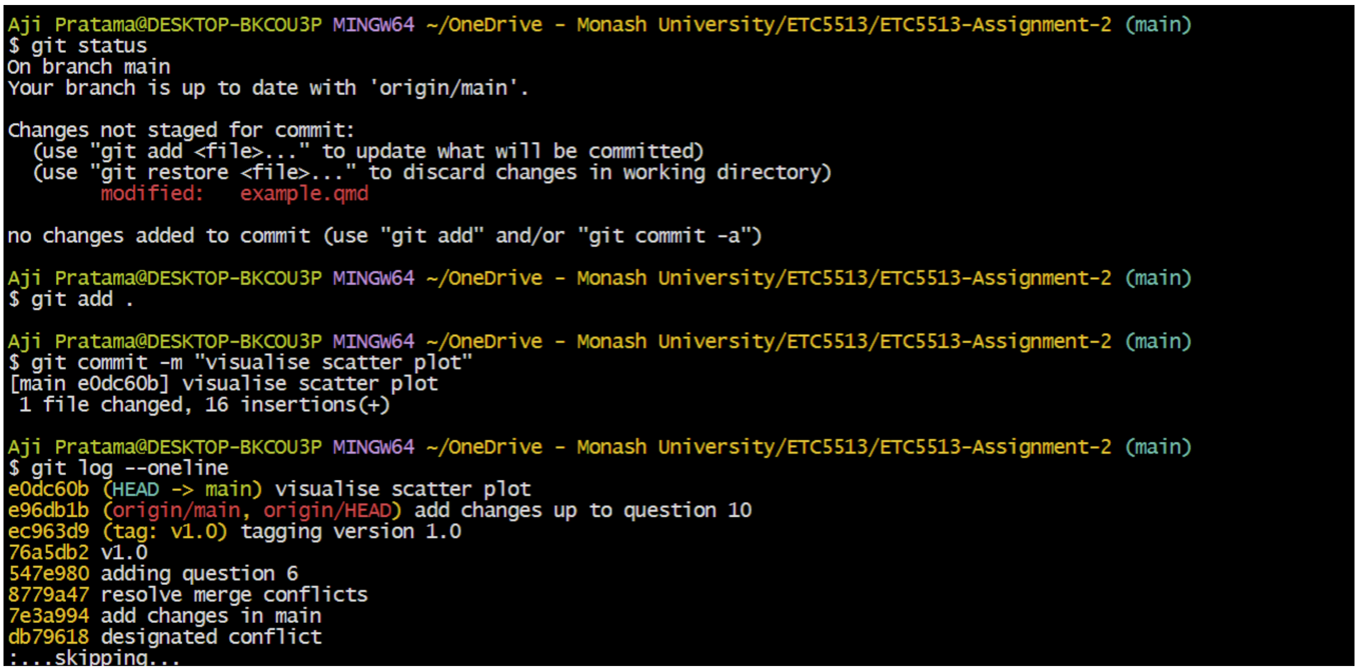
- To do this, type `git reset --soft HEAD~1` in the terminal.

- In this example, we will practice using the function by committing a simple plot using `ggplot2`.

```
── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
✔ dplyr     1.1.4     ✔ readr     2.1.5
✔ forcats   1.0.0     ✔ stringr   1.5.1
✔ ggplot2   3.5.1     ✔ tibble    3.2.1
✔ lubridate 1.9.4     ✔ tidyr     1.3.1
✔ purrr     1.0.4     
── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
✖ dplyr::filter() masks stats::filter()
✖ dplyr::lag()    masks stats::lag()
ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

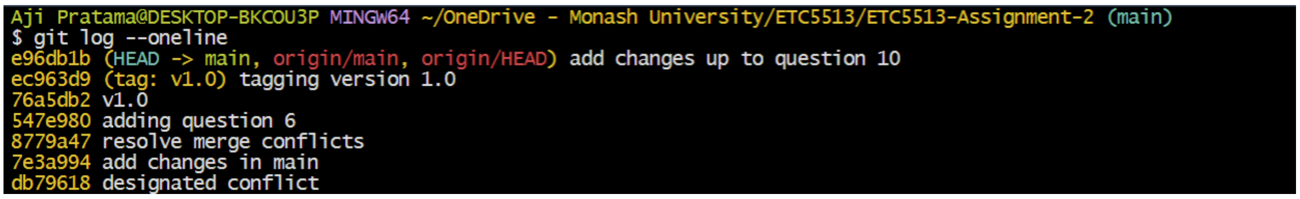


- First, we commit our plot to the repository, but after committing, we realize we want to make some changes to the plot. For example, we want to change the title to "Simple Scatter Plot" instead of "Scatter Plot".



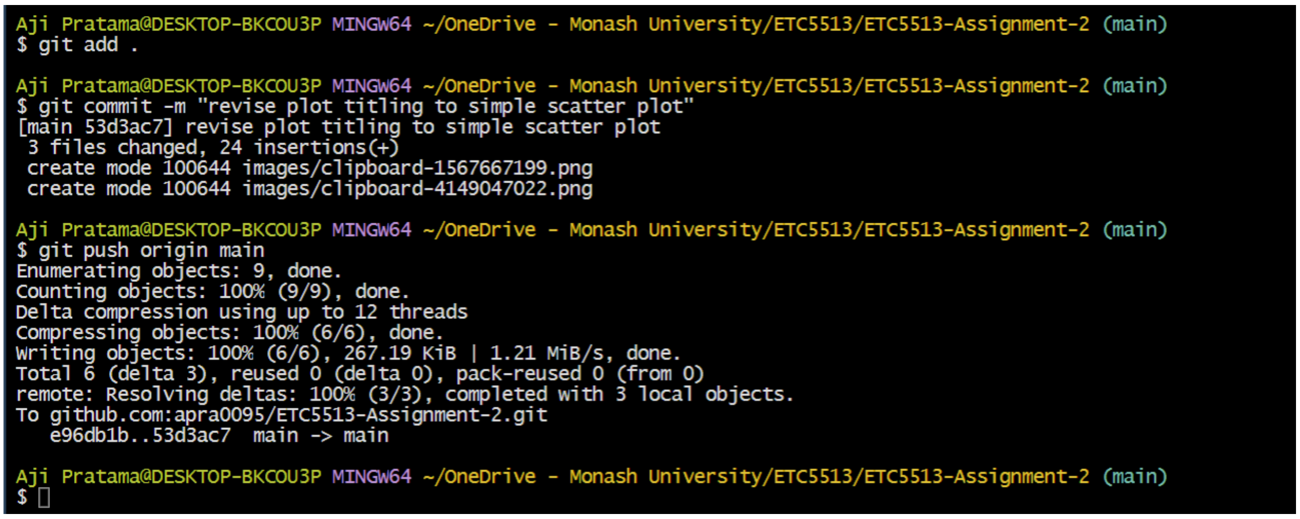
```
Aji Pratama@DESKTOP-BKCOU3P MINGW64 ~/OneDrive - Monash University/ETC5513/ETC5513-Assignment-2 (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   example.qmd

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

Aji Pratama@DESKTOP-BKCOU3P MINGW64 ~/OneDrive - Monash University/ETC5513/ETC5513-Assignment-2 (main)
$ git add .

Aji Pratama@DESKTOP-BKCOU3P MINGW64 ~/OneDrive - Monash University/ETC5513/ETC5513-Assignment-2 (main)
$ git commit -m "visualise scatter plot"
[main e0dc60b] visualise scatter plot
 1 file changed, 16 insertions(+)

Aji Pratama@DESKTOP-BKCOU3P MINGW64 ~/OneDrive - Monash University/ETC5513/ETC5513-Assignment-2 (main)
$ git log --oneline
e0dc60b (HEAD -> main) visualise scatter plot
e96db1b (origin/main, origin/HEAD) add changes up to question 10
ec963d9 (tag: v1.0) tagging version 1.0
76a5db2 v1.0
547e980 adding question 6
8779a47 resolve merge conflicts
7e3a994 add changes in main
db79618 designated conflict
:...skipping...
```

- After running the command `git reset --soft HEAD~1` and checking our commit history, we will see that the commit has moved one step back but the work remains in the directory and is ready to be committed again.



```
Aji Pratama@DESKTOP-BKCOU3P MINGW64 ~/OneDrive - Monash University/ETC5513/ETC5513-Assignment-2 (main)
$ git log --oneline
e96db1b (HEAD -> main, origin/main, origin/HEAD) add changes up to question 10
ec963d9 (tag: v1.0) tagging version 1.0
76a5db2 v1.0
547e980 adding question 6
8779a47 resolve merge conflicts
7e3a994 add changes in main
db79618 designated conflict
```

- After we finish the revision, the file is ready to be committed again.



```
Aji Pratama@DESKTOP-BKCOU3P MINGW64 ~/OneDrive - Monash University/ETC5513/ETC5513-Assignment-2 (main)
$ git add .

Aji Pratama@DESKTOP-BKCOU3P MINGW64 ~/OneDrive - Monash University/ETC5513/ETC5513-Assignment-2 (main)
$ git commit -m "revise plot titling to simple scatter plot"
[main 53d3ac7] revise plot titling to simple scatter plot
 3 files changed, 24 insertions(+)
 create mode 100644 images/clipboard-1567667199.png
 create mode 100644 images/clipboard-4149047022.png

Aji Pratama@DESKTOP-BKCOU3P MINGW64 ~/OneDrive - Monash University/ETC5513/ETC5513-Assignment-2 (main)
$ git push origin main
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 12 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (6/6), 267.19 KiB | 1.21 MiB/s, done.
Total 6 (delta 3), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.
To github.com:apra0095/ETC5513-Assignment-2.git
   e96db1b..53d3ac7  main -> main

Aji Pratama@DESKTOP-BKCOU3P MINGW64 ~/OneDrive - Monash University/ETC5513/ETC5513-Assignment-2 (main)
$ 
```