



Demo #03: Git and GitHub Collaboration

Link to [Demo #3 Hand-in Template](#)

Link to [Git Cheat Sheet](#) (for reference)

Learning objectives

Last week, you learnt how to create a local repo, create an empty remote repo, and push your local repo into the remote repo (Method 1 below). This week, we will try the other methods. We will also try Visual Studio Code. BTW, whenever you see "Elmo", *use your own name* 😊

▼ Summary of some methods to start a repository:

Method	Details	Note	
1. Initialize on the local machine	start folder locally, initialize with <code>git init</code> , stage/commit files, then push to an "empty" remote	no history	last week!
2. Fork and clone	go to an existing project, then use "Fork" to create your copy of it, then <code>git clone URL</code> to bring it to your local computer	keep the commit history of the owner	today!
2. Create a new and clone	create a new repo on GitHub, then <code>git clone URL</code> to bring it to the local computer	no commit history	today!
4. Clone someone else's repo	<code>git clone URL</code> to bring it to the local computer	inherit commit history of the owner	today!
5. Use a template and clone	go to someone's template, then click "Use this template" create your copy of it, then <code>git clone URL</code> to bring it to your local computer	no commit history of the owner	in lab!

▼ Part 1: Fork and clone (Screen Capture #1)

<https://www.attosol.com/top-20-projects-on-github/>

Let's pick some famous open-source projects to "fork" together.

Now it's yours!

You are free to clone, create branches, and add your own work.

When done, you can issue a "Pull request" to ask the owner to pull your changes into the original repo!

We won't do it today.



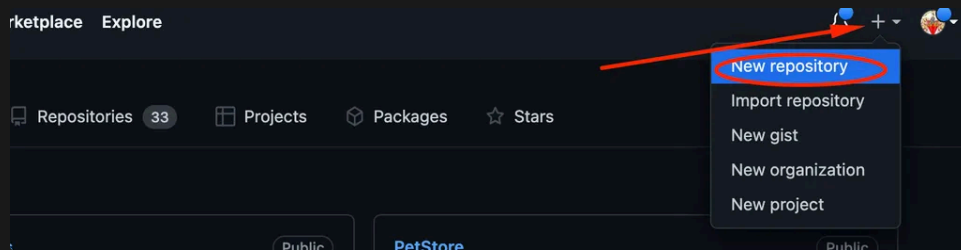
Time to get Screen Capture #1

To prove that you have done everything correctly, let's see YOUR GitHub URL of the forked project!

The screenshot shows the GitHub interface for a repository named 'free-programming-books'. The repository is public and was forked from 'EbookFoundation/free-programming-books'. The user 'COMP1800Hoda' is the owner of this fork. The repository has 1 branch (main) and 0 tags. The commit history shows a recent commit by 'Sumonta056 and eshellman' titled 'Added Bangla C programming course (E...' with commit hash 'ccd4f2c' 5 hours ago, and another commit by 'EbookFoundation' titled 'chore(deps): bump actions/checkout from 3 to 4 (EbookFoundati...' yesterday. The repository also has 8,394 commits in total. The file list shows folders for '.github', 'books', and 'casts'. The 'books' folder has a commit titled 'Added Bangla C programming course (EbookFoundation#9516)' 5 hours ago. The 'casts' folder has a commit titled 'Update free-podcasts-screencasts-ar.md (EbookFoundation#9357)' 4 months ago.

▼ Part 2: Create a new non-empty repo on GitHub (Screen Capture #2)

1. Go to your gitHub, and click on the "+" (top right corner), to Create a new repo.



2. Let's make a (non-empty) repo:

- use name " `demo03-yourname` "
- public (for comp1800 learning purposes)
- add a `README` file
- add `.gitignore` file (Node template)
- choose " `The Unlicense` " just for fun (not needed for 1800 project)

Create a new repository

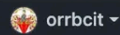
A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Repository template

Start your repository with a template repository's contents.

No template ▾

Owner *



orrbcit ▾

Repository name *

/ demo03-carly ✓

Great repository names are short and memorable. Need inspiration? How about [fictional-octo-enigma?](#)

Description (optional)



Public

Anyone on the internet can see this repository. You choose who can commit.



Private

You choose who can see and commit to this repository.

Initialize this repository with:

Skip this step if you're importing an existing repository.

☒ Add a README file

This is where you can write a long description for your project. [Learn more.](#)

☒ Add .gitignore

Choose which files not to track from a list of templates. [Learn more.](#)

.gitignore template: Node ▾

☒ Choose a license

A license tells others what they can and can't do with your code. [Learn more.](#)

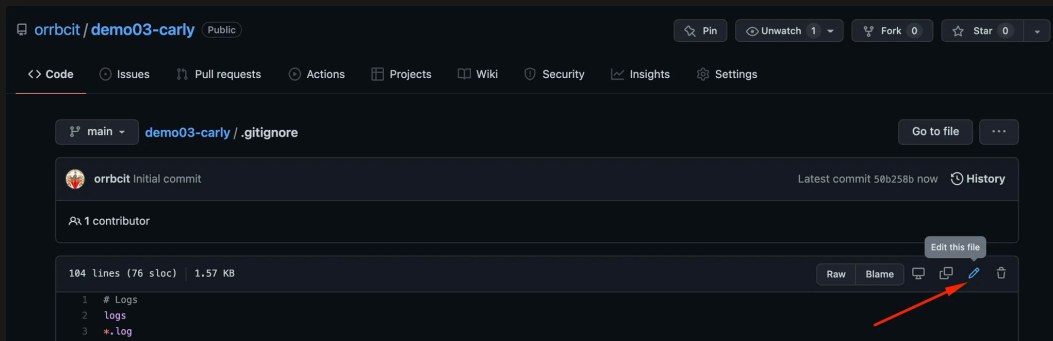
License: The Unlicense ▾

This will set `main` as the default branch. Change the default name in your [settings](#).

Create repository

3. Edit the `.gitignore` file in GitHub.

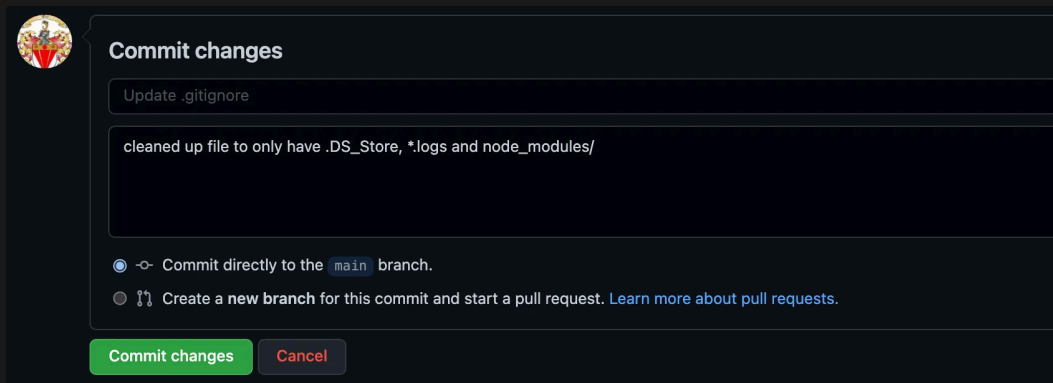
Note: GitHub is not an IDE. But, minor edits are allowed. Saving changes means doing a "commit".



Delete everything in the file, since we don't need them in our project. The only files that affect us in comp1800 might be:

```
.DS_Store *.log node_modules/ .vscode/
```

After editing, then "save" the file, we actually perform a "**Commit**". Add a comment, and click "Commit changes".

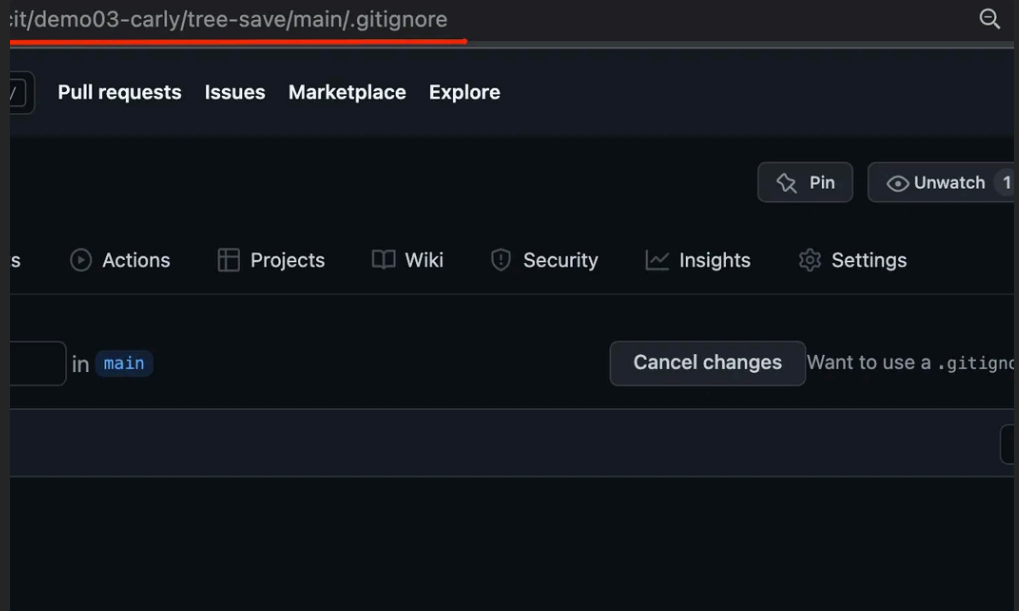


Optional: Edit the `README.md` file to provide a title and author, and try doing a small edit in GitHub. The file extension "`.md`" means GitHub Markdown Language file. It is easy to learn. To save, you must "commit" this change.



Time to get Screen Capture #2

To prove that you have done everything correctly, let's see your GitHub URL, and your `.gitignore` file contents.



▼ Part 3: Clone someone's repo (Screen Capture #3)

1. Let's go to your computer and create a working location.

Let's clone SOMEONE'S project.

le, Today, please clone your instructor's project. **THE URL WILL BE GIVEN TO YOU DURING DEMO TIME.**

After cloning, "`cd`" (change directory) into the repo. Type "`dir`" (or `ls`) to look around.

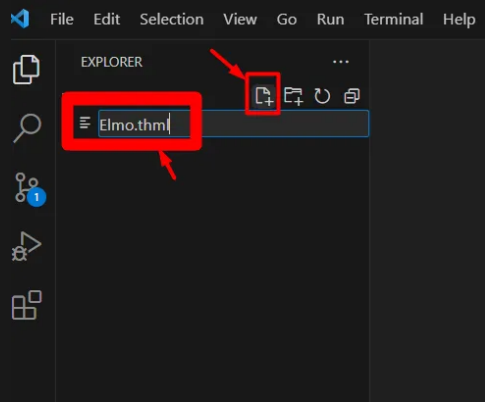
```
cd Documents/comp1800 mkdir demo03 cd demo03 git clone URL (URL
to be cloned will be given to you during Demo) dir cd repo-name
dir
```

2. Now, open this project using a code editor, such as VSCode.

I will use VSCode in this folder and launch it by typing "`code .`" at the command prompt.

You can also launch VSCode first, then open the files in the repo folder, using the VSCode File Menu.

3. Let's code!



Create an HTML file named after you, and put a simple `<h1>` message there. Customize with your name.

Eg. `elmo.html`, `hesam.html`, `carly.html`

```
<html> <body> <h1>This is Elmo's page!</h1> </body> </html>
```

Create an `index.html` with the following code. Let's add our first interaction element! Customize with your name.

```
<html> <body> <h1>This is the demo03 landing page.</h1> <button  
type="button" onclick="window.location.href='elmo.html'"> Go to  
Elmo's Page!</button> </body> </html>
```

That's it! Select the index HTML file in VSCode, right-click and choose "Open with Live Server" to render it. Or, you can simply open `index.html` from the file folder, and double-click.

Your web page shows up! Click on the "Go to Elmo's Page" button.

Does everything look good?

Now that you are finished your coding. It's time to stage/commit. In VSCode, open " `Terminal` → `New Terminal` " from the menu. (You can also stage/commit from the command-line window, ie. the place where you first cloned.)

```
git add . git commit -m "adding index.html and yourname.html" git status
```



Time to get Screen Capture #3

To prove that you have done everything correctly, let's see your VS Code window with (at least) 3 items: `index.html`, `yourname.html`, and a terminal window showing clean "`git status`". Your project name should reflect the project that was cloned.

Hint: Go to View, Editor Layout, and split up/down to show two files like so.

Example image:

