Pull Requests.

Many developers who don't follow a particular workflow say that it's a waste of time and that it takes away precious development time. There is a truth in that statement, because following the workflow means waiting for other people to review your code. Even so, you have to keep in mind that you don't have to wait around doing nothing. While you wait for a review, you can directly go on to solve another issue! That's why branches are so powerful in Version Control Systems; you can work on multiple issues at the same time.

With the workflow, you can begin to work on an issue, ask for some ideas or directive from your peers, and then work on another issue while you await responses. After you receive the necessary feedback, you can continue to work on the first issue. You may repeat this until the issue is resolved.

Pull Requests, as useful as they are, are a fairly easy-to-understand concept. Submitting a Pull Request, or PR, is just asking for permission to apply all the commits in a branch to another branch. All the same, we're moving too fast. Before learning about Pull Requests, we have to learn what a "pull" is.

Pull

In Git terminology, a pull is just the opposite of push (give yourself a high five if you guessed that!). Push takes your branch and copies all its commits to a remote branch. It also creates the branch if it doesn't exist on the server yet. Pull is just that, but backward; it looks at a remote branch and copies the commits on it to your local repository. It's just an exchange of commits; push if it's from local to remote, and pull if it's from remote to local.

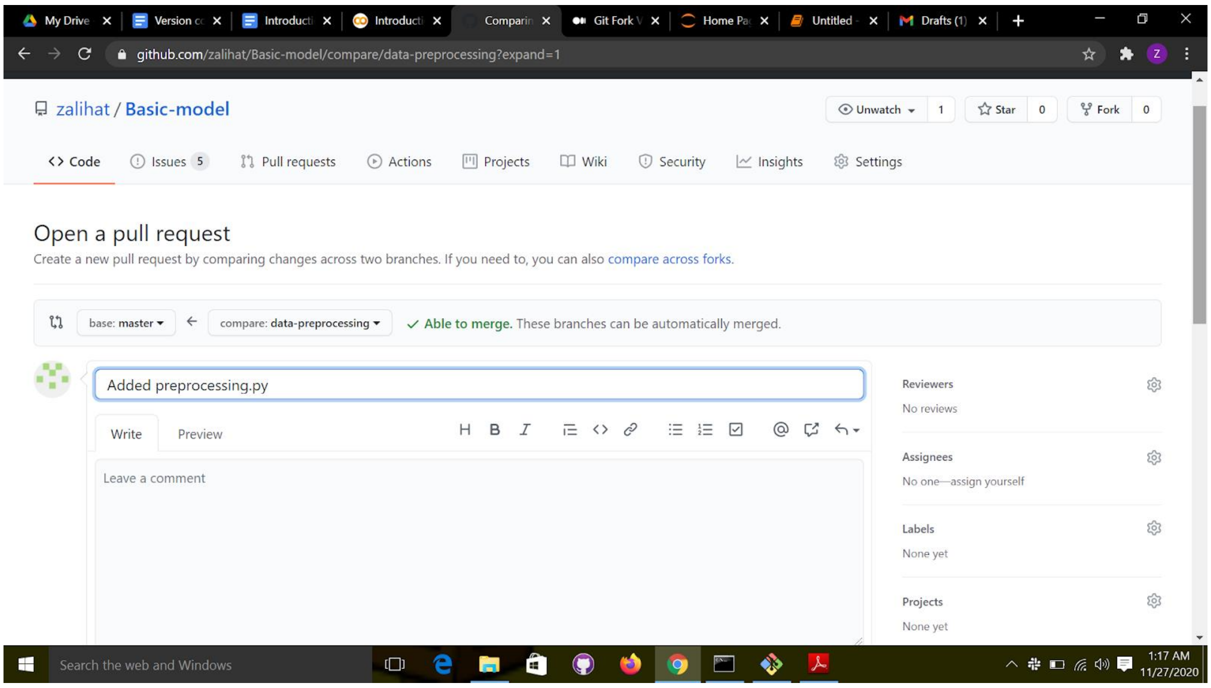
The syntax is very similar too:

```
git pull <remote_name> <branch_name>
```

For example, if you wanted to get the commits from the master branch on GitHub, you would have to execute the command while checking out the master branch:

```
 git pull origin master
```

In the last example, there is a new call to action on the page, right above the list of branches. It shows the name of the branch that you just created, and a big button for creating a PR. Click the button to continue; you should get to the Pull Request creation form, just like the Figure below;



github.com/zalihat/Basic-model/compare/data-preprocessing?expand=1

zalihat / Basic-model

Unwatch 1 | Star 0 | Fork 0

Code | Issues 5 | Pull requests | Actions | Projects | Wiki | Security | Insights | Settings

Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also compare across forks.

base: master ← compare: data-preprocessing ✓ Able to merge. These branches can be automatically merged.

Added preprocessing.py

Write | Preview

Leave a comment

Reviewers
No reviews

Assignees
No one—assign yourself

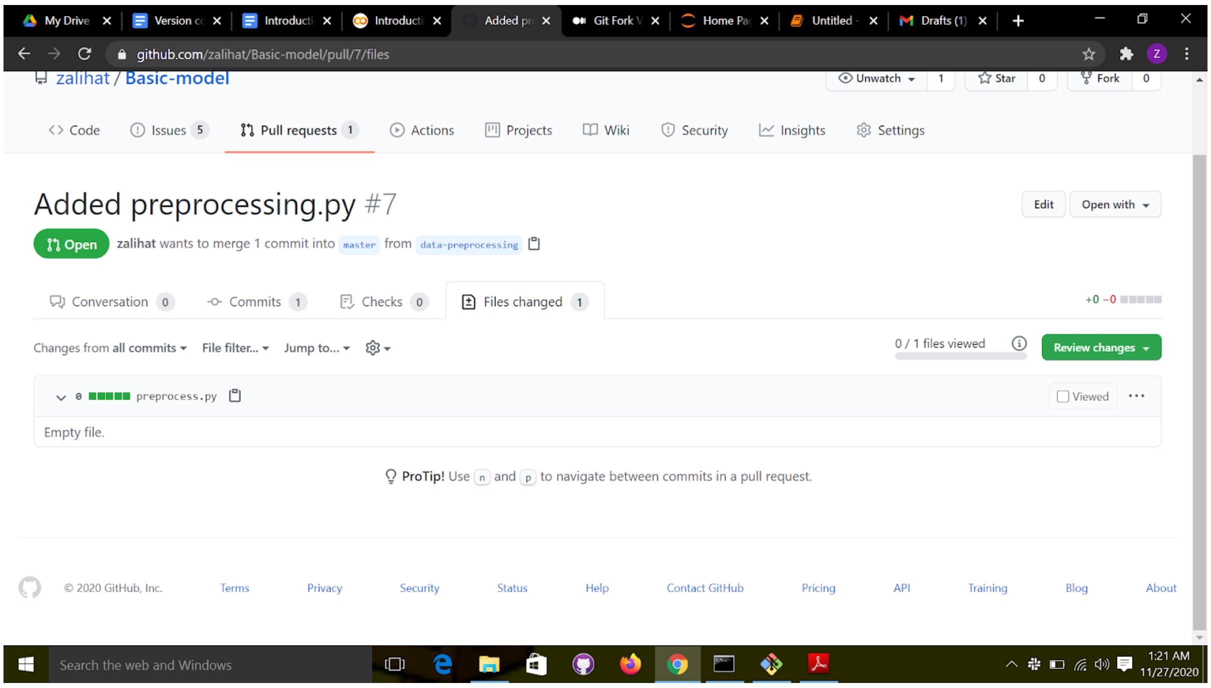
Labels
None yet

Projects
None yet

Figure 37: Pull request form
You can note that the PR creation form is very similar to the issue creation form.

On the right, you can find the same information about assignees and labels; they work exactly the same. On the bottom of the page, you can see the commits to be applied by the Pull Request; and if you scroll down, you'll find the differences between the versions.

Check Figure below for an example of this.



github.com/zalihat/Basic-model/pull/7/files

zalihat / Basic-model

Code | Issues 5 | Pull requests 1 | Actions | Projects | Wiki | Security | Insights | Settings

Added preprocessing.py #7

Open — zalihat wants to merge 1 commit into `master` from `data-preprocessing`

Conversation 0 | Commits 1 | Checks 0 | Files changed 1 — +0 −0

Changes from all commits ▾ File filter... ▾ Jump to... ▾ — 0 / 1 files viewed — Review changes ▾

0 preprocess.py — Viewed

Empty file.

ProTip! Use n and p to navigate between commits in a pull request.

© 2020 GitHub, Inc. Terms Privacy Security Status Help Contact GitHub Pricing API Training Blog About

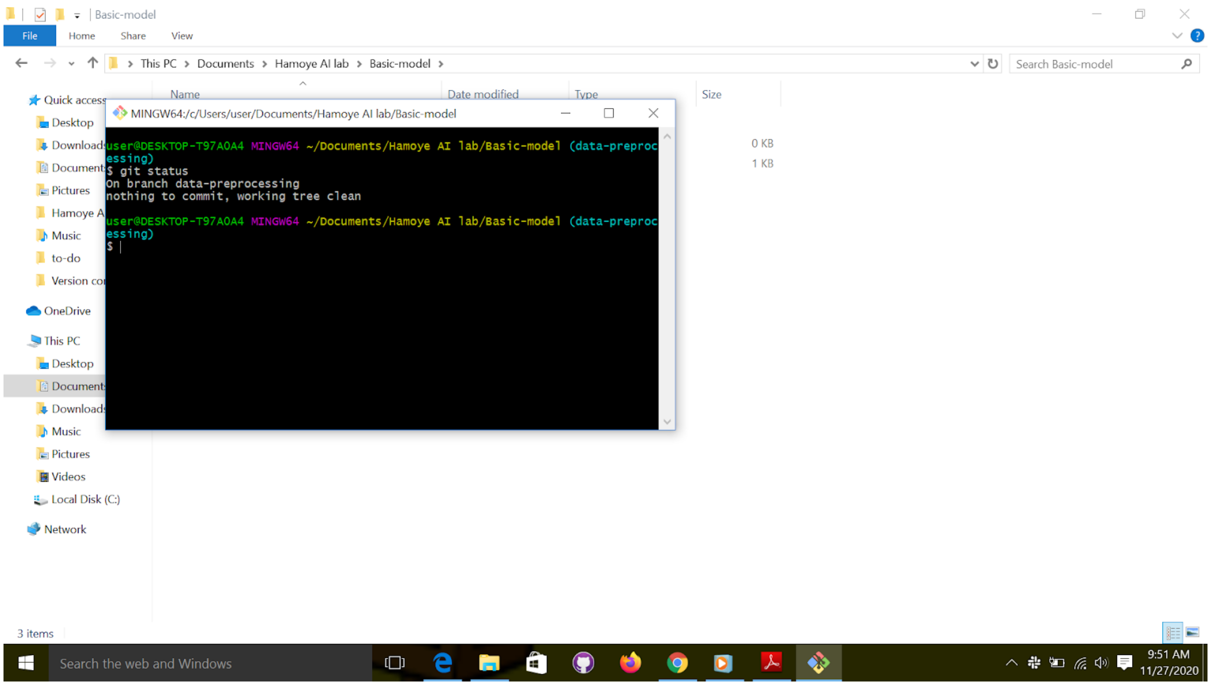
Figure 38: sample of a Pull request
After you create a Pull request, the reviewer can then review the changes. After the review, if the changes are great, the branch will be merged with the master branch.

Git Merge

The changes made to the data-preprocessing will only be on the branch until we merge the branch with the master branch. To merge the two branches, we use the git merge command,

To do that, run the command below;

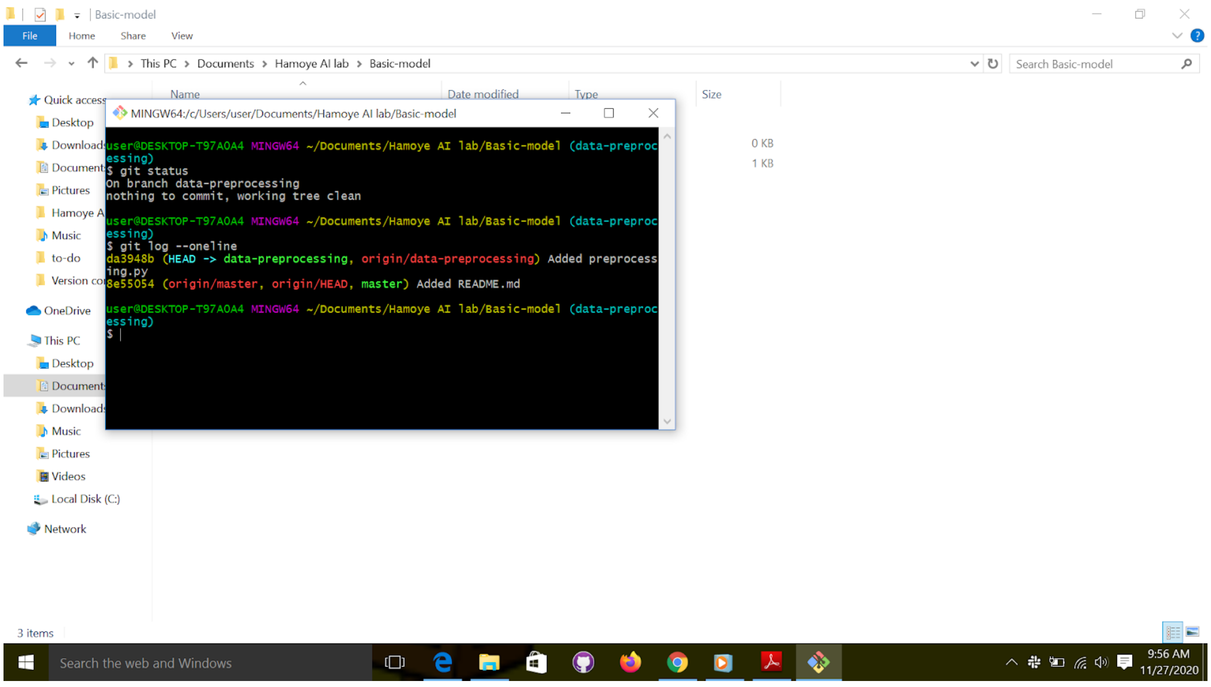
1. Git status to check the status of the repository. As shown in the figure below, we are still in the data-preprocessing branch



```
user@DESKTOP-T97A0A4 MINGW64 ~/Documents/Hamoye AI lab/Basic-model (data-preproc
essing)
$ git status
On branch data-preprocessing
nothing to commit, working tree clean

user@DESKTOP-T97A0A4 MINGW64 ~/Documents/Hamoye AI lab/Basic-model (data-preproc
essing)
$ |
```

Figure 39: checking the status of the repository.

2. Run git log to check the project history.



```
user@DESKTOP-T97A0A4 MINGW64 ~/Documents/Hamoye AI lab/Basic-model (data-preproc
essing)
$ git status
On branch data-preprocessing
nothing to commit, working tree clean

user@DESKTOP-T97A0A4 MINGW64 ~/Documents/Hamoye AI lab/Basic-model (data-preproc
essing)
$ git log --oneline
da3948b (HEAD -> data-preprocessing, origin/data-preprocessing) Added preprocess
ing.py
8e55054 (origin/master, origin/HEAD, master) Added README.md

user@DESKTOP-T97A0A4 MINGW64 ~/Documents/Hamoye AI lab/Basic-model (data-preproc
essing)
$ |
```

Figure 40: checking the project's history

As shown in the figure below, we are on branch data-preprocessing and the branch has one commit.

Use the option “--oneline” when using git log to get a prettier result.

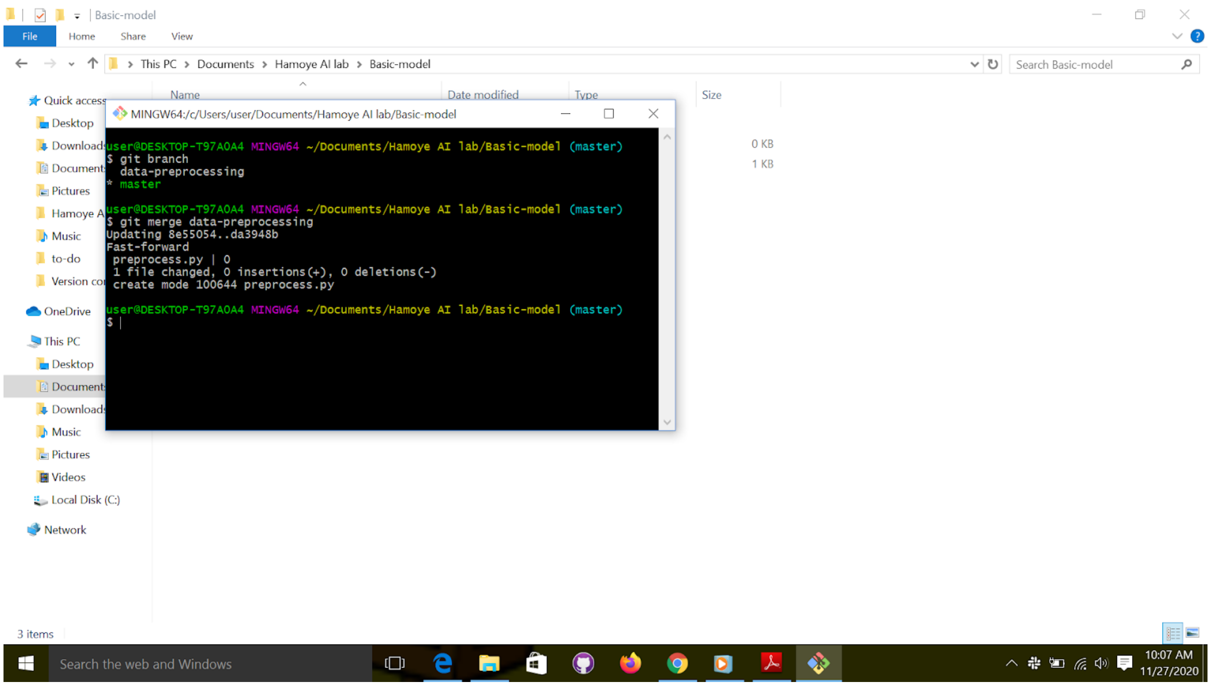
3. As you can see in the figure above, HEAD now points to the last commit of our new branch; which means that every commit we create will have that as a parent. You will also notice that the master didn’t change; that’s because we only worked on our newly created branch. Now that we are satisfied with our fix, let’s merge the branch to the master branch so we can test it. To merge our branch into master, we have to first check it out. Hence, switch to the master branch by running the git checkout command.

4. Now let’s try to merge the branch into the master branch. Merging just means reproducing all the commits on one branch on another. To do so, we will use the “git merge” command, followed by the name of the branch being merged.

git merge <name>

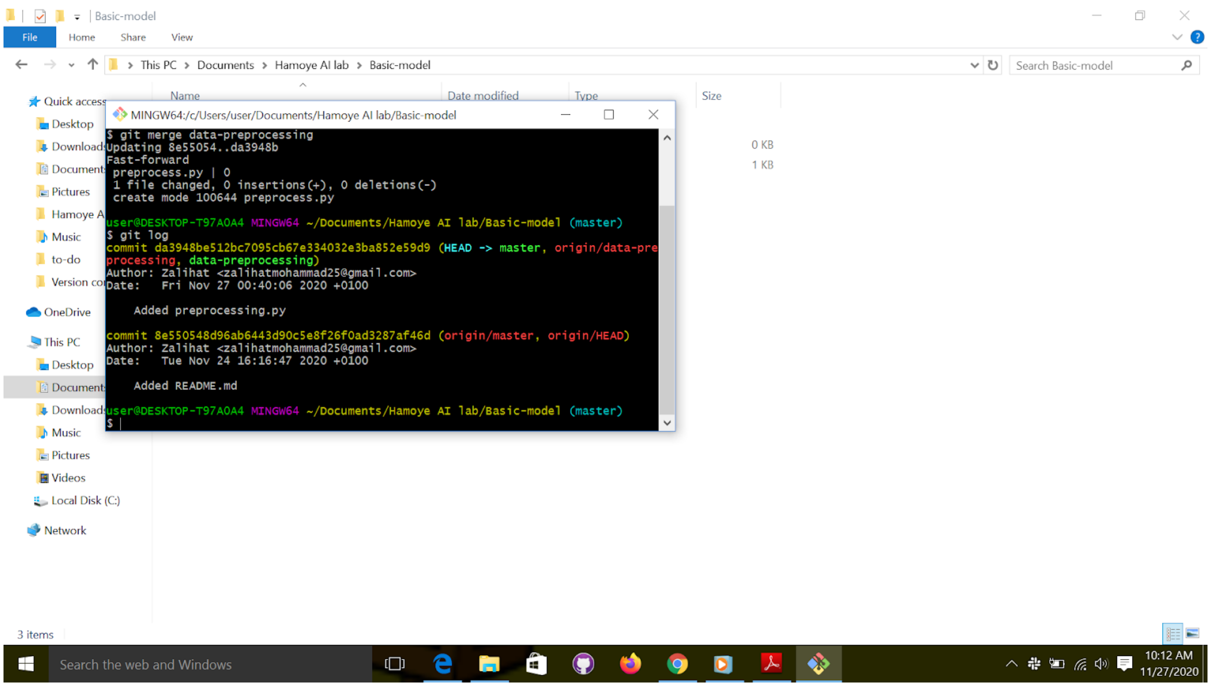
Since we are looking to merge “data-preprocessing” into “master,” ourcommand to execute on the master branch is;

git merge data-preprocessing



```
user@DESKTOP-T97A0A4 MINGW64 ~/Documents/Hamoye AI lab/Basic-model (master)
$ git branch
  data-preprocessing
* master

user@DESKTOP-T97A0A4 MINGW64 ~/Documents/Hamoye AI lab/Basic-model (master)
$ git merge data-preprocessing
Updating 8e55054..da3948b
Fast-forward
 preprocess.py | 0
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 preprocess.py

user@DESKTOP-T97A0A4 MINGW64 ~/Documents/Hamoye AI lab/Basic-model (master)
$ |
```

Figure 41: Git merge command
Let's check the history of the repository by running the “git log” command;



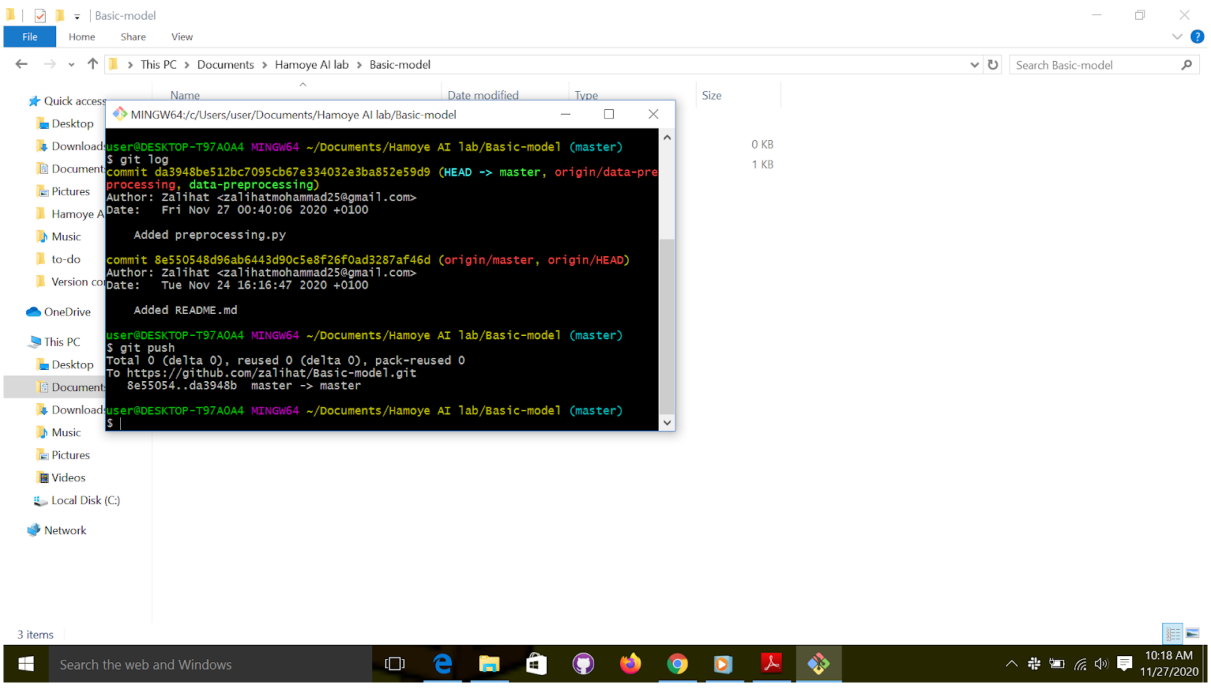
```
$ git merge data-preprocessing
Updating 8e55054..da3948b
Fast-forward
 preprocess.py | 0
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 preprocess.py

user@DESKTOP-T97A0A4 MINGW64 ~/Documents/Hamoye AI lab/Basic-model (master)
$ git log
commit da3948be512bc7095cb67e334032e3ba852e59d9 (HEAD -> master, origin/data-pre
processing, data-preprocessing)
Author: Zalihat <zalihatmohammad25@gmail.com>
Date:   Fri Nov 27 00:40:06 2020 +0100

    Added preprocessing.py

commit 8e550548d96ab6443d90c5e8f26f0ad3287af46d (origin/master, origin/HEAD)
Author: Zalihat <zalihatmohammad25@gmail.com>
Date:   Tue Nov 24 16:16:47 2020 +0100

    Added README.md

user@DESKTOP-T97A0A4 MINGW64 ~/Documents/Hamoye AI lab/Basic-model (master)
$ 
```

Figure 42: checking the project’s history

As you can see, “HEAD“ now points to “develop” because it’s the checked-out branch. You can also notice that master and data-preprocessing now point to the same commit; that’s because of the merge.

The changes made now are only on the local repository. Therefore, to push the changes to the remote repository, we use the git push command.

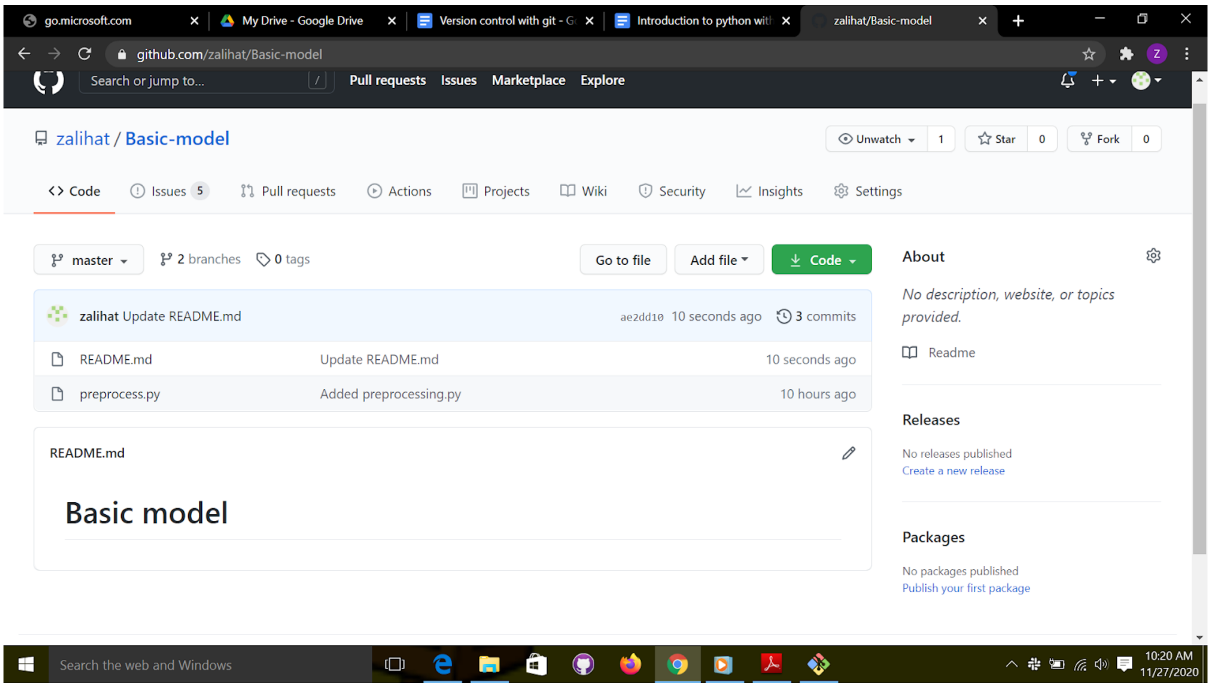
git push



```
user@DESKTOP-T97A0A4 MINGW64 ~/Documents/Hamoye AI lab/Basic-model (master)
$ git log
commit da3948be512bc7095cb67e334032e3ba852e59d9 (HEAD -> master, origin/data-pre
processing, data-preprocessing)
Author: Zalihat <zalihatmohammad25@gmail.com>
Date:   Fri Nov 27 00:40:06 2020 +0100

    Added preprocessing.py

commit 8e550548d96ab6443d90c5e8f26f0ad3287af46d (origin/master, origin/HEAD)
Author: Zalihat <zalihatmohammad25@gmail.com>
Date:   Tue Nov 24 16:16:47 2020 +0100

    Added README.md

user@DESKTOP-T97A0A4 MINGW64 ~/Documents/Hamoye AI lab/Basic-model (master)
$ git push
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/zalihat/Basic-model.git
   8e55054..da3948b  master -> master

user@DESKTOP-T97A0A4 MINGW64 ~/Documents/Hamoye AI lab/Basic-model (master)
$ 
```

Figure 43: Push changes to the remote repository.

Now, check the remote repository to see if the changes reflected on it. It should look like this;


zalihat / Basic-model

master · 2 branches · 0 tags

zalihat Update README.md · ae2dd10 10 seconds ago · 3 commits

README.md	Update README.md	10 seconds ago
preprocess.py	Added preprocessing.py	10 hours ago

README.md

Basic model

About

No description, website, or topics provided.

Releases

No releases published

Packages

No packages published

Figure 44: checking remote repository for changes made.