The Rookies: Git Assignment

1. Question 1:
    - Go to GitHub and create a new private repository named “basic_git_flow”.
    - Result:

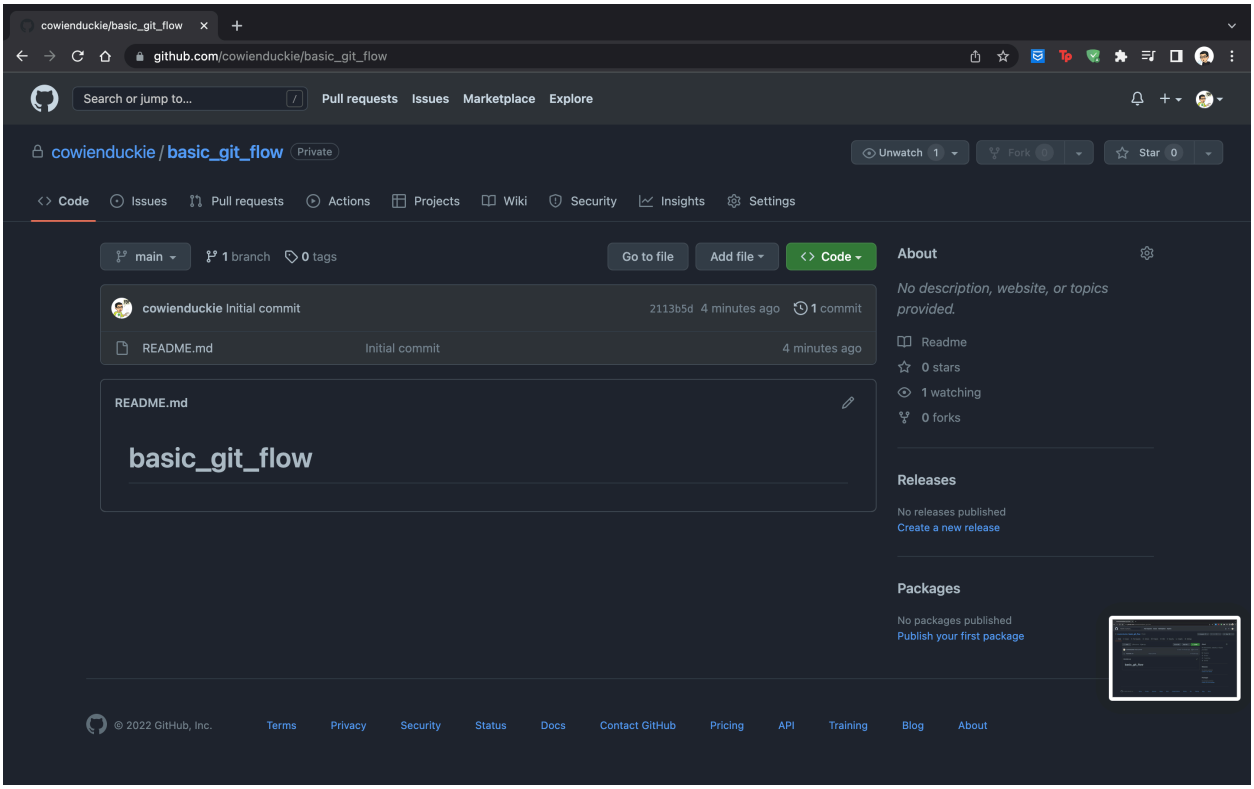

Figure 1. Created private repository on GitHub

2. Question 2:
    - Terminal commands:

```
<TL> git clone https://github.com/cowienduckie/basic_git_flow.git
<TL> git branch develop
<TL> git checkout develop
<TL> <Add some files>
<TL> git add .
<TL> git commit -m “A develop commit”
<TL> git push --set-upstream origin develop
```

- Result:

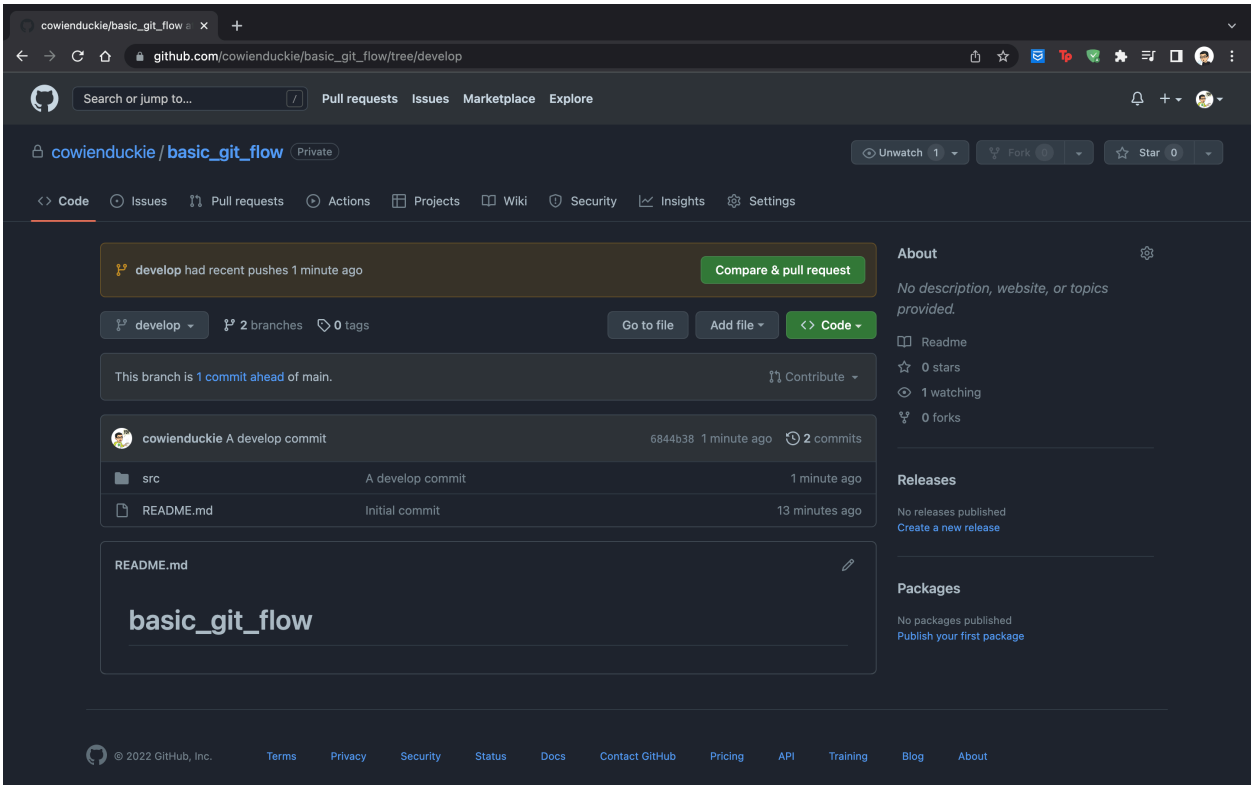

Figure 2. Created develop branch

3. Question 3:
    - Terminal commands:

```
<DEVA> git checkout develop
<DEVA> git branch feature/feat1
<DEVA> git checkout feature/feat1
<DEVA> git add .
<DEVA> git commit -m “A feature/feat1 commit”
<DEVA> git push --set-upstream origin feature/feat1
<DEVA> git request-pull develop origin feature/feat1
```

    - The “request-pull” command seems not to work in this situation.
    - I have tried to use “push” command from “feature/feat1” to “origin/develop”. Only the first time, the terminal create a pull request link for me. Maybe the reason is that I push and create new branch at the same time.
    - After 1 hour trying, I go to GitHub and create a pull request manually right on website.

- Result:

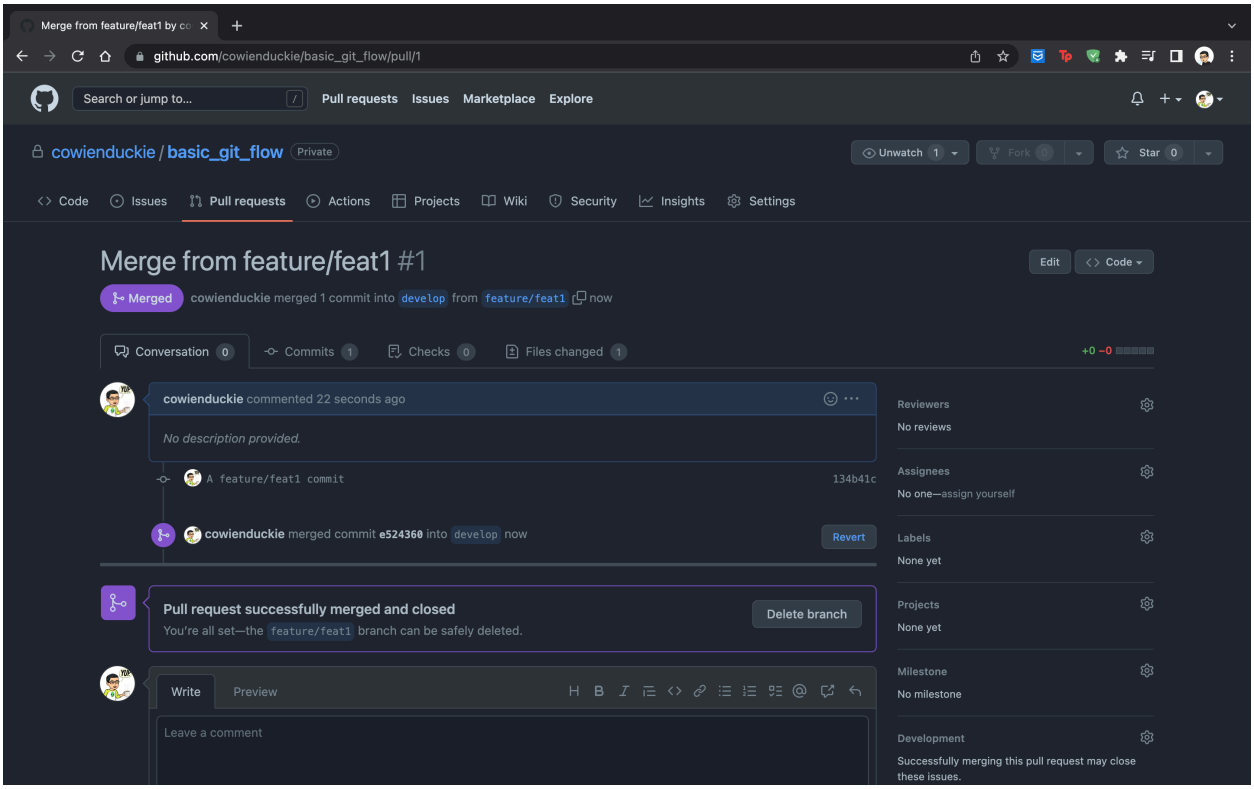

Figure 3. Merged pull request from feature/feat1 to develop

4. Question 4:
    - Terminal commands:

```
<DEVC> git checkout develop
<DEVC> git branch feature/feat2
<DEVC> git checkout feature/feat2
<DEVC> git add .
<DEVD> git commit -m “A feature/feat2 commit”
<DEVC> git push --set-upstream origin feature/feat2
<DEVD do some changes after DEVC commit>
<DEVD> git fetch
<DEVD> git pull
<DEVD> git config merge.tool <merge tool name>
<DEVD> git mergetool
<DEVD> git add .
```

```
<DEVD> git commit -m “Another feature/feat2 commit”
<DEVD> git push
<DEVD> git request-pull develop origin feature/feat2
```

- To decrease conflicts when commit changes, each dev has to fetch and pull the latest commit of the branch. If any conflict occurs, use merge tool to combine both incoming/our code.
- There are a lot of merge tools to visualize differences. Recommend using one of them to reduce mistakes when resolving conflicts.
- At last, communication is the most important if 2 devs both make changes in a file, a function, or a piece of code.
- Result:

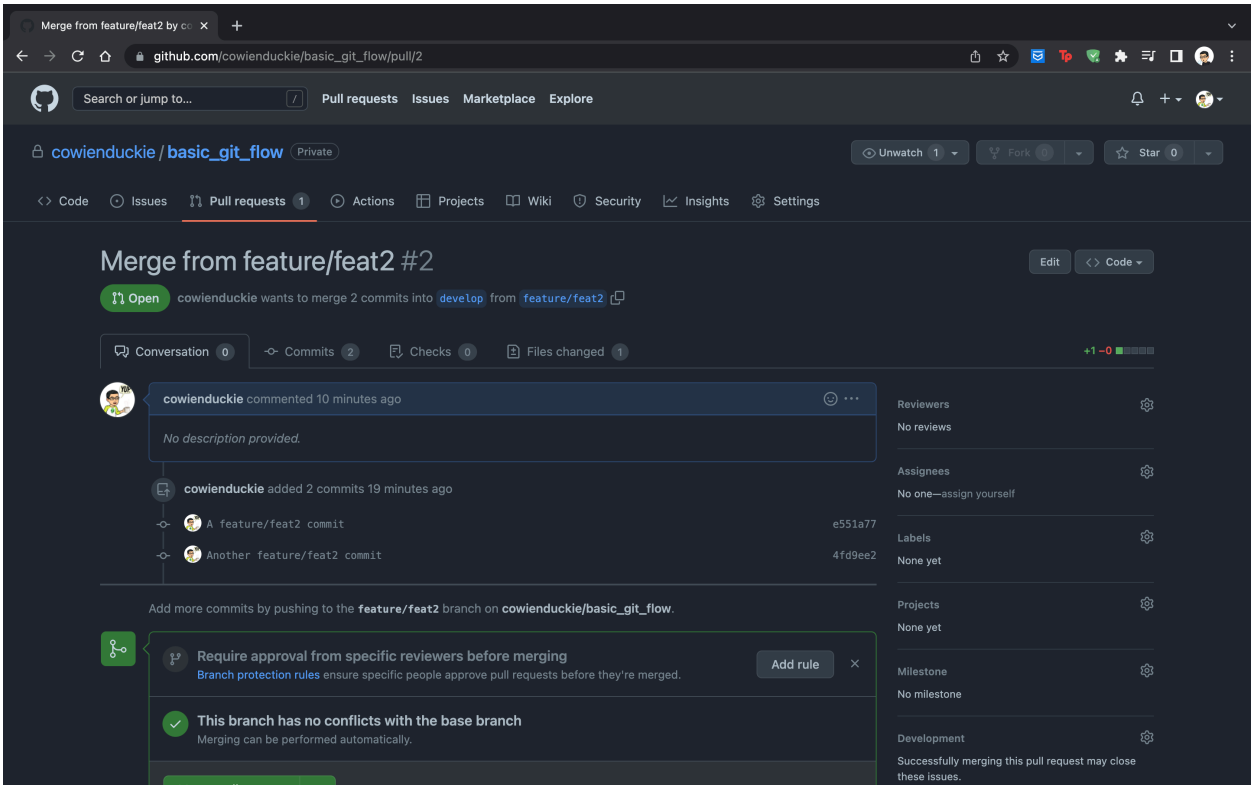

Figure 4. Created pull request from feature/feat2 to develop

5. Question 5:
    - Terminal commands:

```
<TL make some changes on develop, commit and push it to remote>
<DEVD create a pull request from feature/feat2 to develop>
<TL> git checkout feature/feat2
<TL> git fetch
<TL> git pull
<TL> git rebase origin/develop
<TL> git push --force origin feature/feat2
```

    - After that, we could see the changes in develop are put before ones in feature/feat2.
    - I think the reason for this is that we can safely merge a ton of code that is commited from a temporary branch, as feature/feat2, to a long-live branch like develop.
    - For more safety, we should create a backup branch for feature/feat2. If something went wrong, just use “reset --hard" to use the backup.
    - Result:

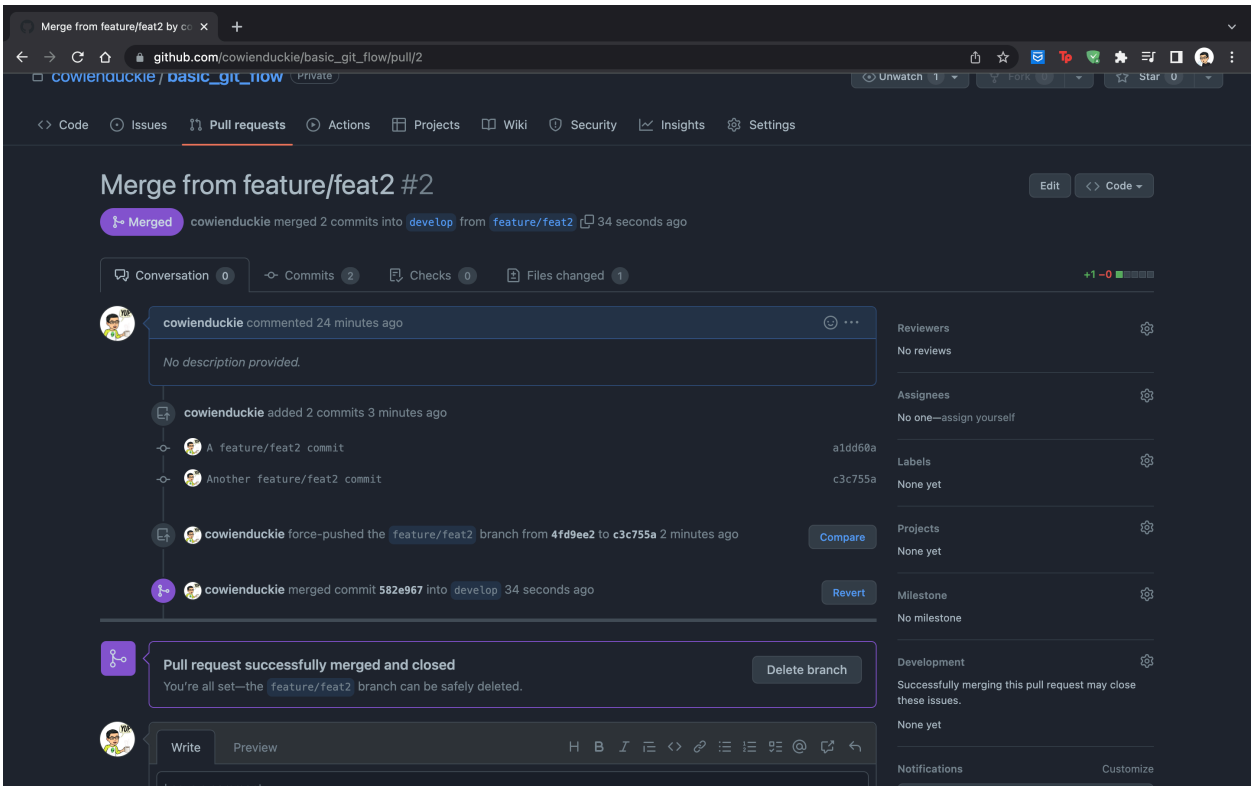

Figure 5. Merged pull request after rebase feature/feat2

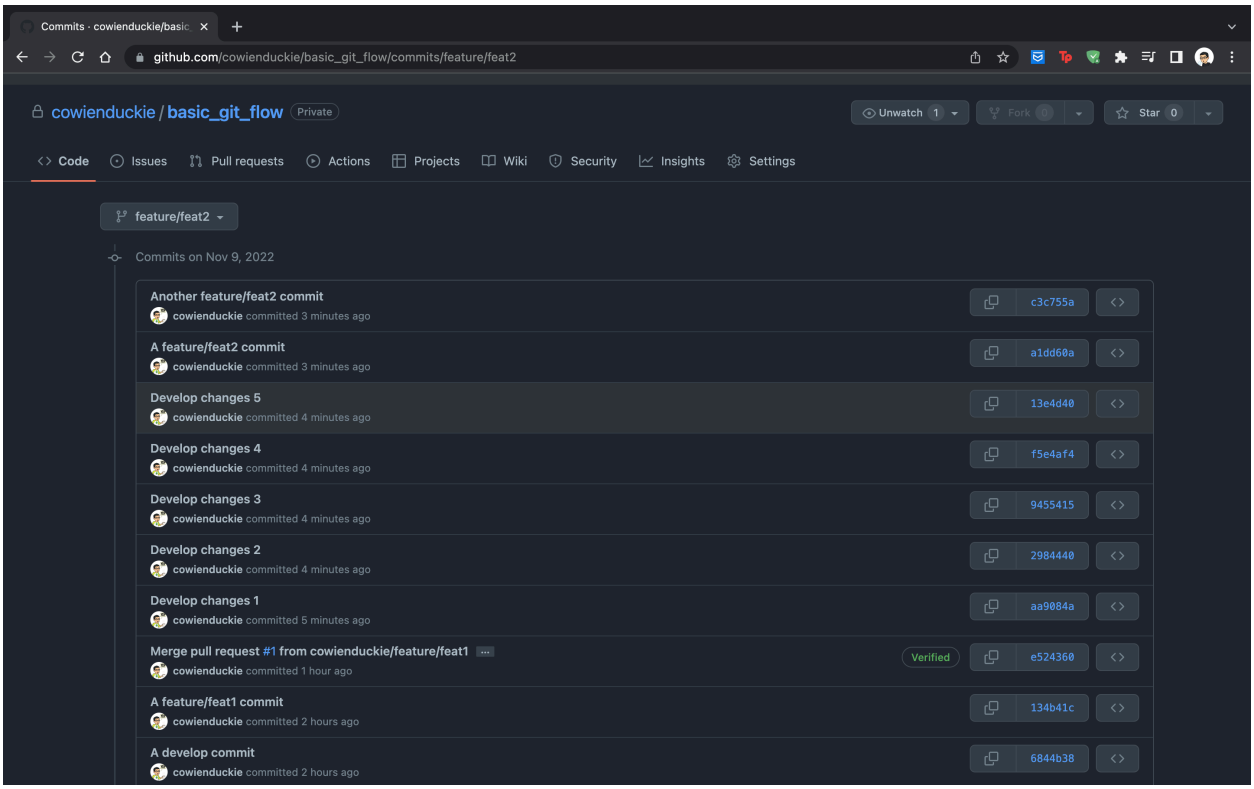

Figure 6. All commits of feature/feat2 after rebase from develop

6. Question 6:

   - Terminal commands for “git rebase” solution:

```
<DEVB add extraBehavior, commit and push to feature/ex_behavior>
<DEVA> git rebase origin/feature/ex_behavior
<DEVA> git push --force origin feature/feat3
<DEVA reuse DEVB’s code and fix feat3>
```

   - Terminal commands for “git merge” solution:

```
<DEVB add extraBehavior, commit and push to feature/ex_behavior>
<DEVA> git merge origin feature/ex_behavior
<DEVA> git commit -m “Merge branch 'feature/ex_behavior' into
feature/feat3”
<DEVA reuse DEVB’s code and fix feat3>
<DEVA> git commit -m “Fix feat3”
<DEVA> git push
<DEVA create a pull request to develop from feature/feat3>
```

- Result:

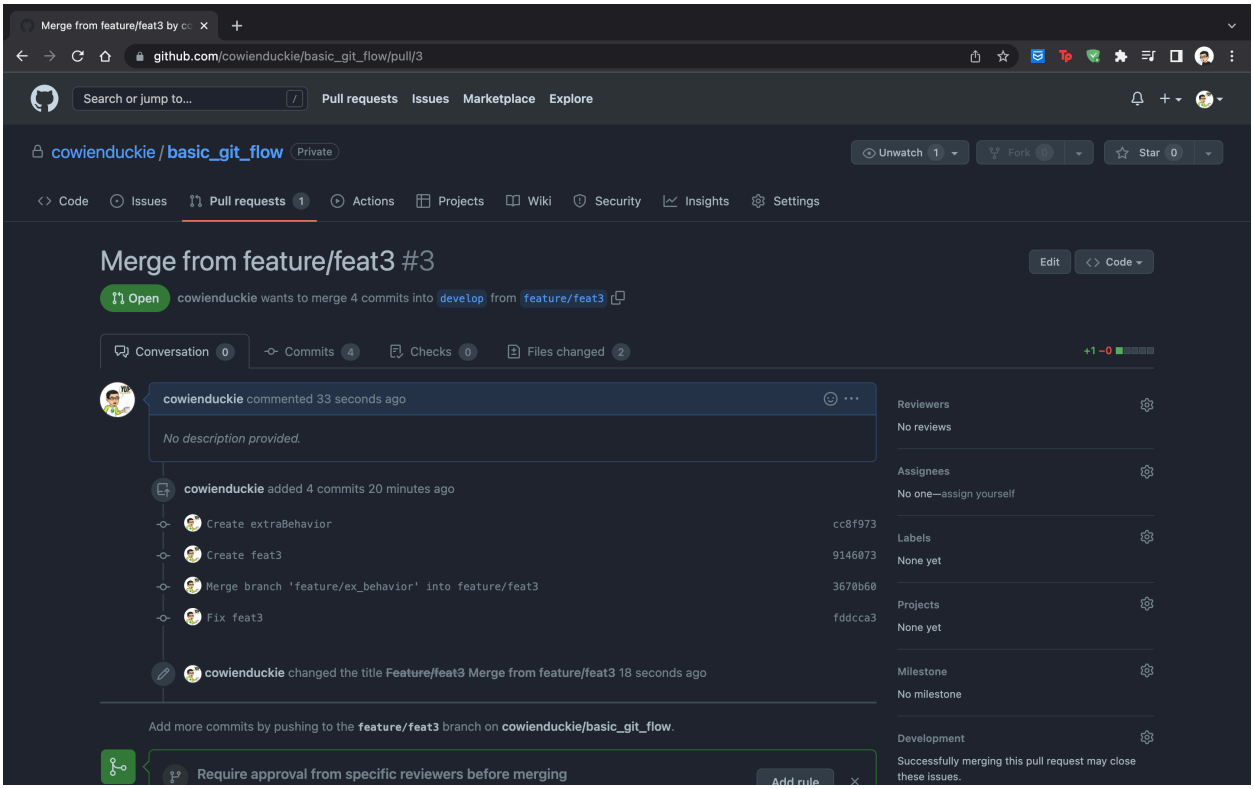

Figure 7. Created pull request from feature/feat3 to develop

7. Question 7:
    - Terminal commands:

```
<TL make some changes on release/r1, commit and push>
<TL> git checkout main
<TL> git merge origin release/r1
<TL> git push
<TL> git checkout develop
<TL> git merge origin release/r1
<TL> git push
```

- In my opinion, “main/master” branch is the most stable code, we should use a pull request to merge code into it.
- With my knowledge, I don’t know if is there any way to sync from a release branch automatically or just a smarter way. Please let me know if there are any.

- Result:

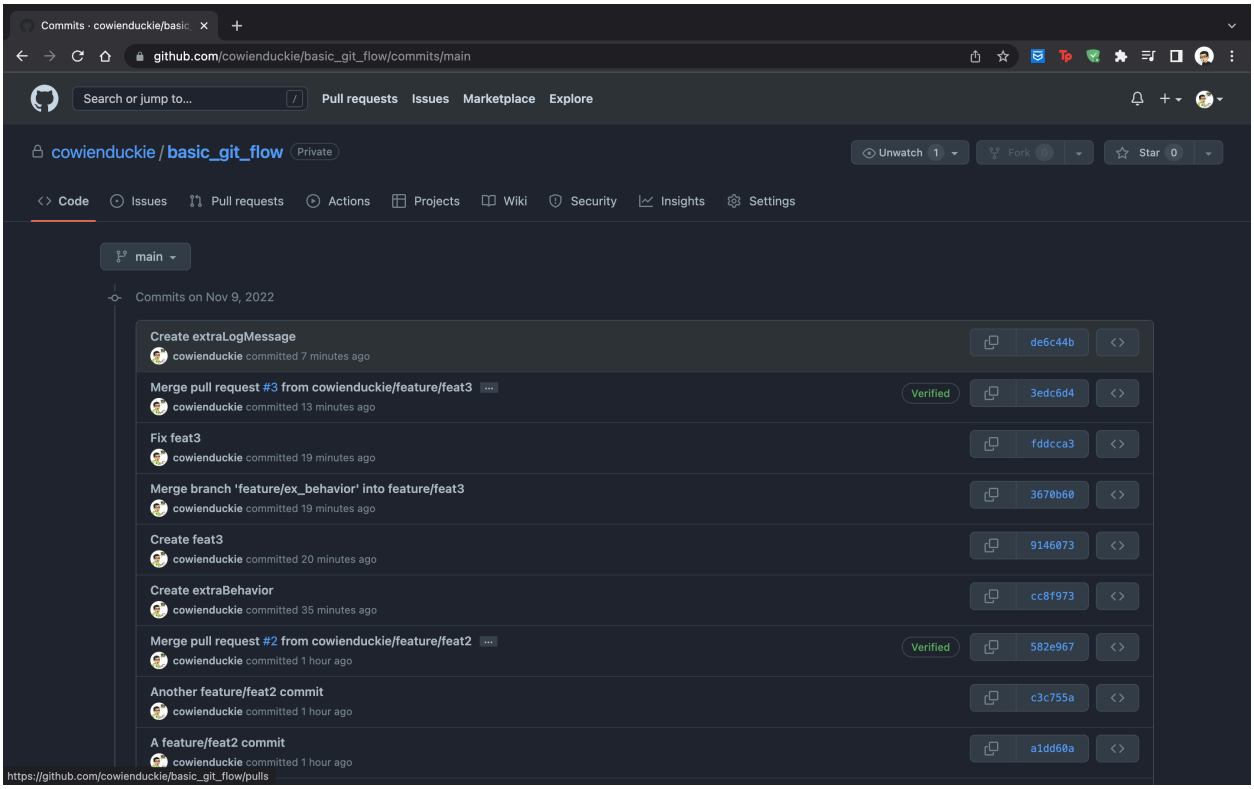

Figure 8. All commits of main after sync from release/r1

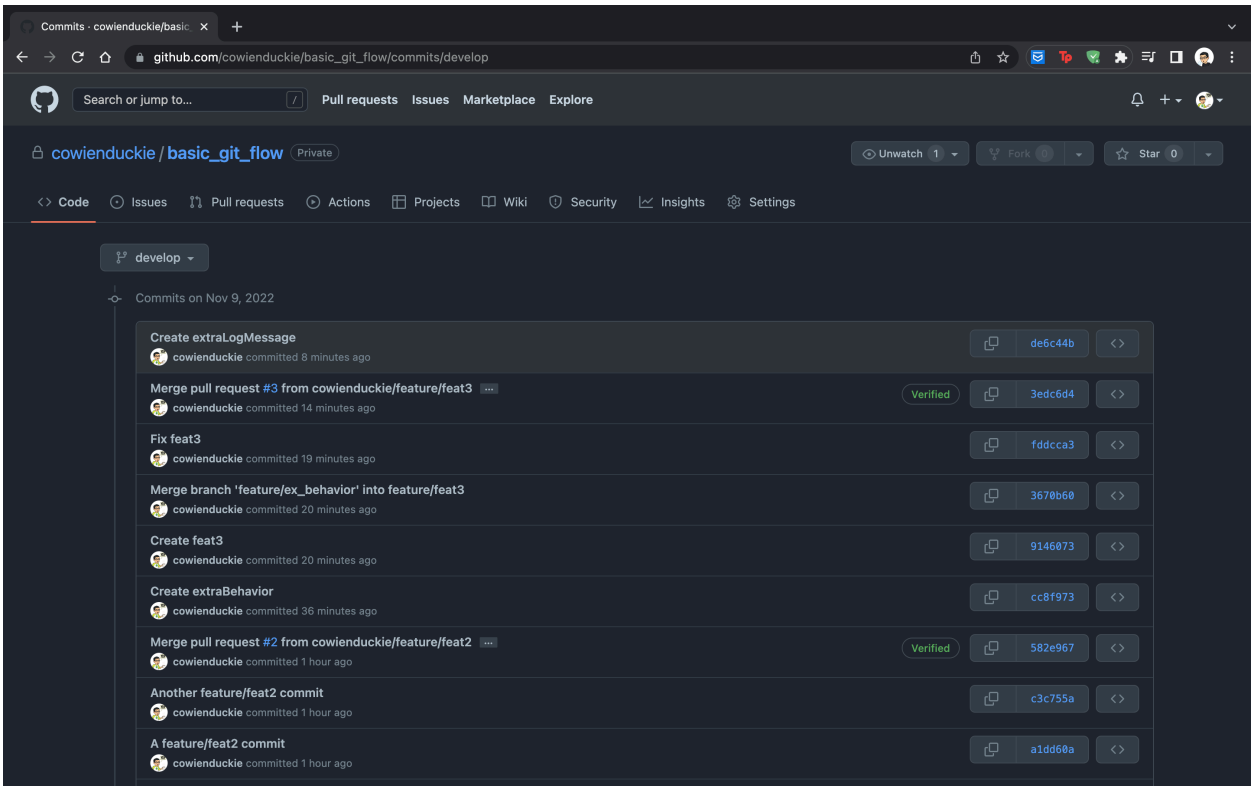

Figure 9. All commits of develop after sync from release/r1