

오픈소스SW기초 (2024-2)

9. Git 명령어 II (프로젝트 협업)

국립금오공과대학교

컴퓨터공학과 / 인공지능공학과

김 경 수

목차

- ① GitHub 프로젝트 협업 방법
- ② GitHub 프로젝트 협업 방법 실습

학습 목표

- ① GitHub에서 다른 프로젝트에 기여하는 방법들에 대해 이해하고 차이점을 설명할 수 있다.
- ② GitHub를 활용하여 하나의 프로젝트에 대해 여러 사용자들이 협업하는 방법을 이해하고 이에 필요한 작업들을 수행할 수 있다.
- ③ 조별 활동을 통해 GitHub 프로젝트 협업에 필요한 기능들을 숙지할 수 있다.

GitHub 프로젝트 협업 방법

GitHub의 다른 프로젝트에 기여하기 위한 방법

① 원격 저장소에 대한 협력자(collaborator) 자격으로 기여하는 방법

- 해당 프로젝트에 협력자로 등록한 후 직접적으로 참여하는 방법
- 주로 프로젝트 개발의 핵심 인력일 때 활용하는 방법

② 기여자(contributor) 자격으로 프로젝트에 참여하는 방법

- 내가 해당 프로젝트에 등록된 협력자가 아니더라도 코드를 수정한 후 수정 반영을 프로젝트 관리자에게 요청할 수 있음.
- 주로 외부에 공개된 대규모 오픈소스 프로젝트에 활용되는 방법

GitHub 협업을 위한 사용자의 종류

- 관리자(Supervisor)

- 해당 프로젝트(원격 저장소)의 총 관리자이자 소유자
- 협력자를 초대하거나 프로젝트 기여 작업에 대한 승인/거부 등을 수행

- 협력자(Collaborators)

- 프로젝트의 공동 책임자로, 프로젝트의 관리자(소유자)가 직접 지정
- 해당 원격 저장소에 대한 commit, push, pull 등의 작업을 자유롭게 수행할 수 있음

GitHub 협업을 위한 사용자의 종류

- 기여자(Contributors)

- 현재 프로젝트에 commit하는 모든 사용자들을 통칭
- 협력자와는 달리 해당 프로젝트의 원격 저장소에 push 명령을 수행할 권한은 없음.
- 대신 pull request 기능을 사용하여 자신이 commit한 내용을 프로젝트에 최종적으로 반영할 것인가를 프로젝트 관리자와 지속적으로 토론한 후, 관리자가 pull request를 최종 승인하면 push가 완료됨.

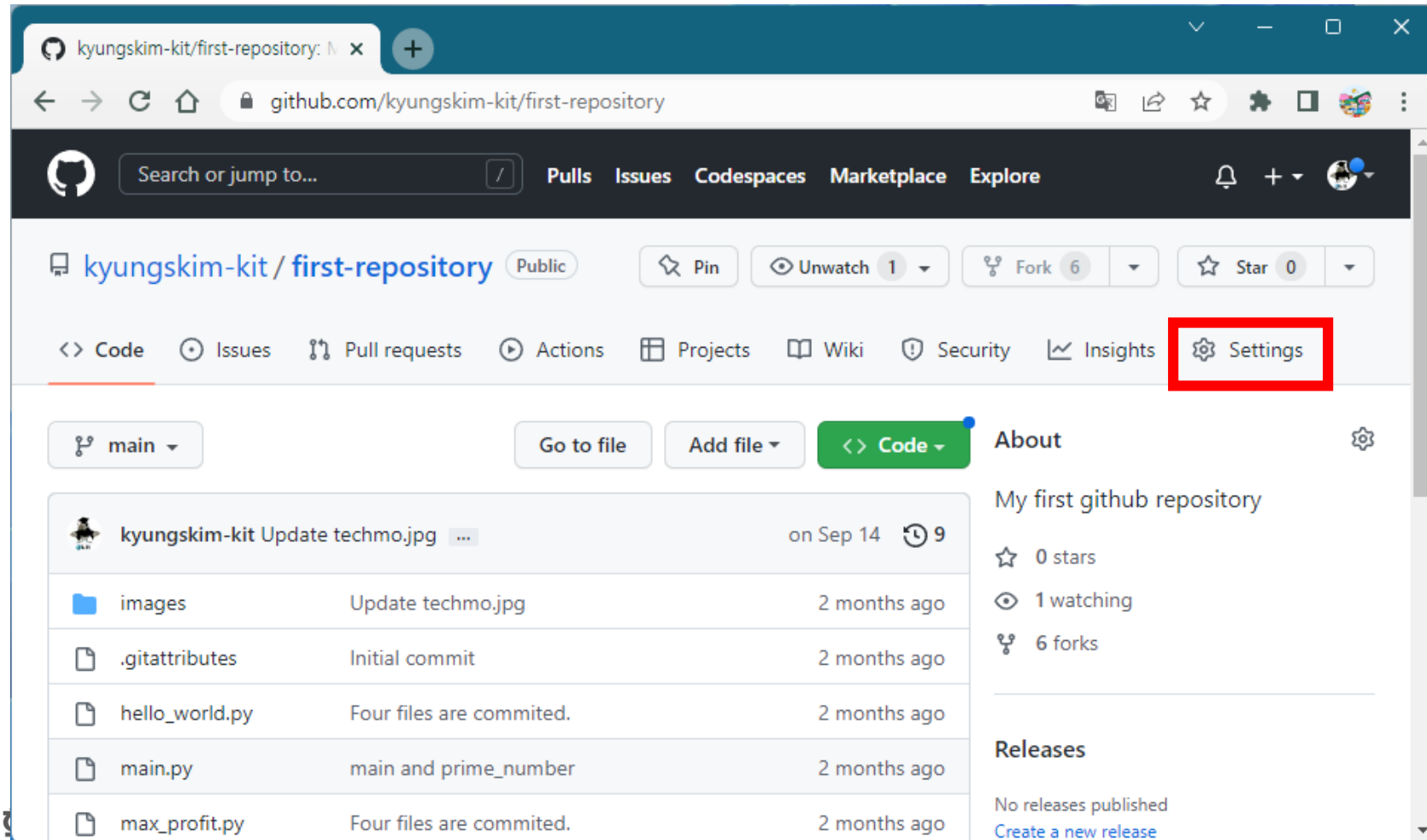
원격 저장소 협력자 추가

- 프로젝트의 원격 저장소에 협력자를 추가하는 방법

- ① 원격 저장소 화면 상단의 “Settings” 클릭
- ② 왼쪽 메뉴에서 “Collaborators” 선택
- ③ 사용자 이름(ID) 또는 이메일 주소를 입력한 후 “Add collaborator” 버튼 클릭
- ④ “Copy invite link” 창에 표시된 주소를 해당 협력자에게 전송
- ⑤ 초대받은 협력자는 “Accept invitation”을 버튼을 클릭하여 해당 프로젝트의 협력자로 등록

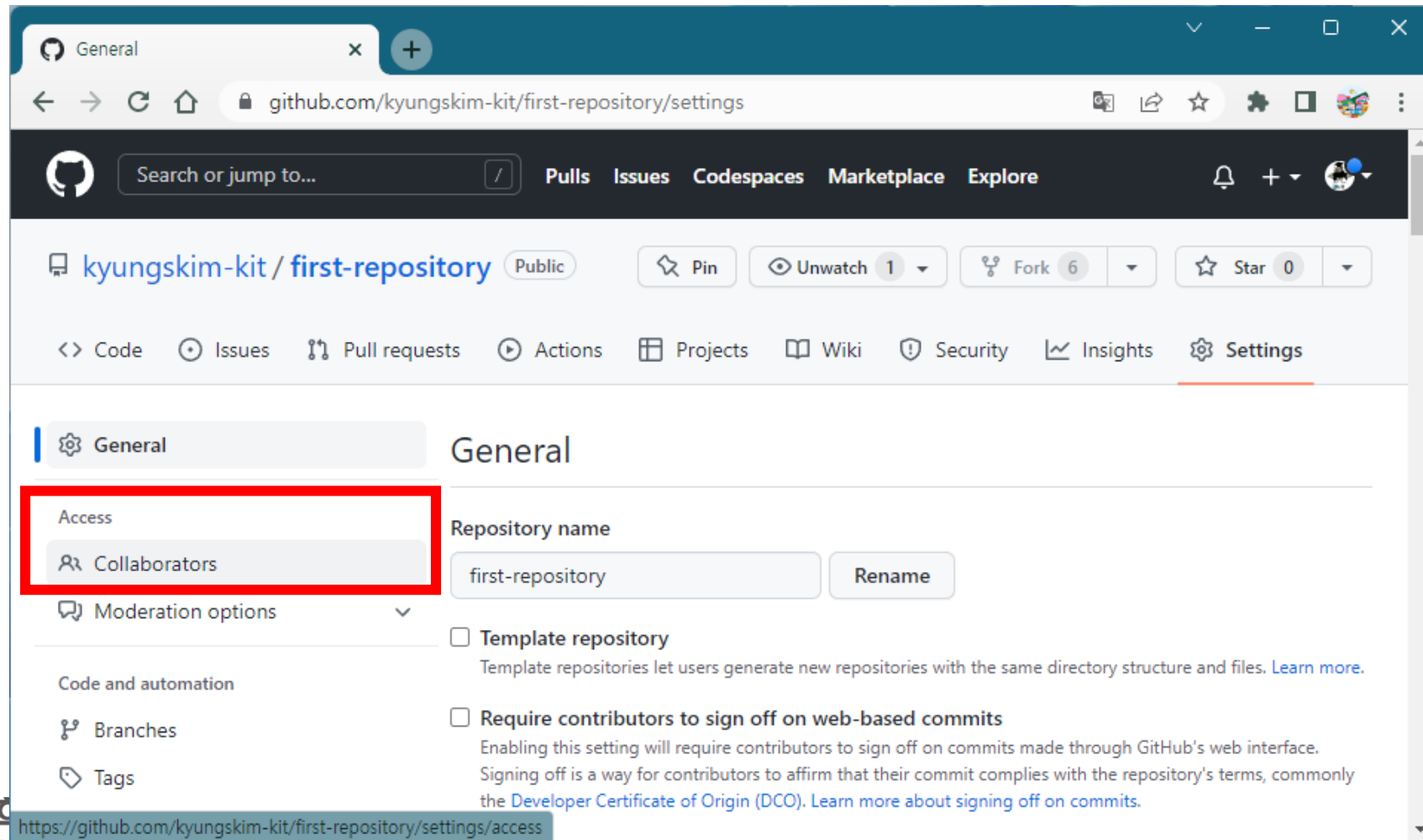
원격 저장소 협력자 추가 방법

- 원격 저장소 화면 상단의 “Settings” 클릭



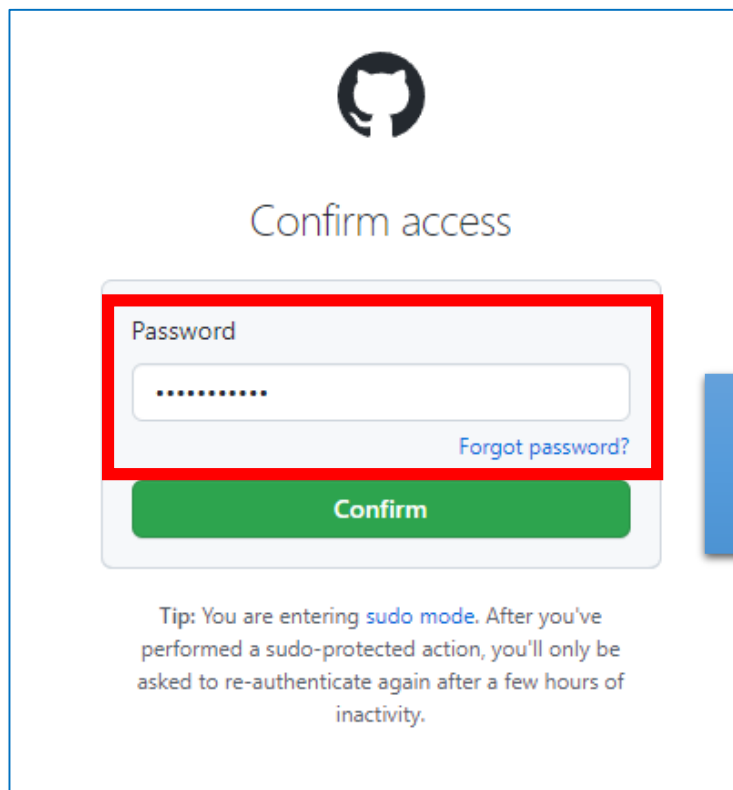
원격 저장소 협력자 추가 방법

- 왼쪽 메뉴에서 “Collaborators” 선택



원격 저장소 협력자 추가 방법

- “Confirm access” 화면이 나오면 패스워드를 다시 한 번 입력
- “Manage access”에서 “Add people” 버튼을 클릭



Confirm access

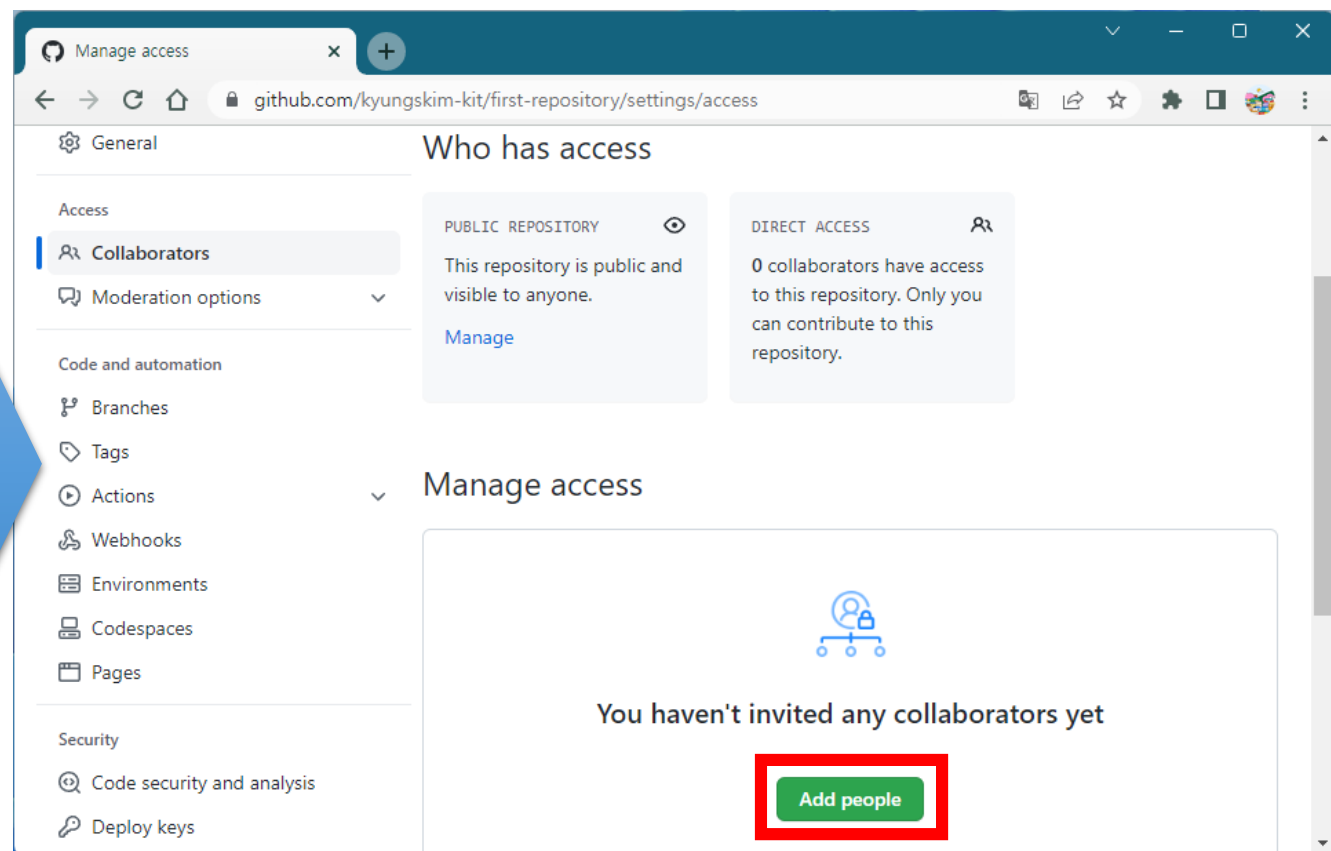
Password

.....

[Forgot password?](#)

Confirm

Tip: You are entering **sudo mode**. After you've performed a sudo-protected action, you'll only be asked to re-authenticate again after a few hours of inactivity.

Manage access

github.com/kyungskim-kit/first-repository/settings/access

General

Access

Collaborators

Moderation options

Code and automation

Branches

Tags

Actions

Webhooks

Environments

Codespaces

Pages

Security

Code security and analysis

Deploy keys

Who has access

PUBLIC REPOSITORY

This repository is public and visible to anyone.

[Manage](#)

DIRECT ACCESS

0 collaborators have access to this repository. Only you can contribute to this repository.

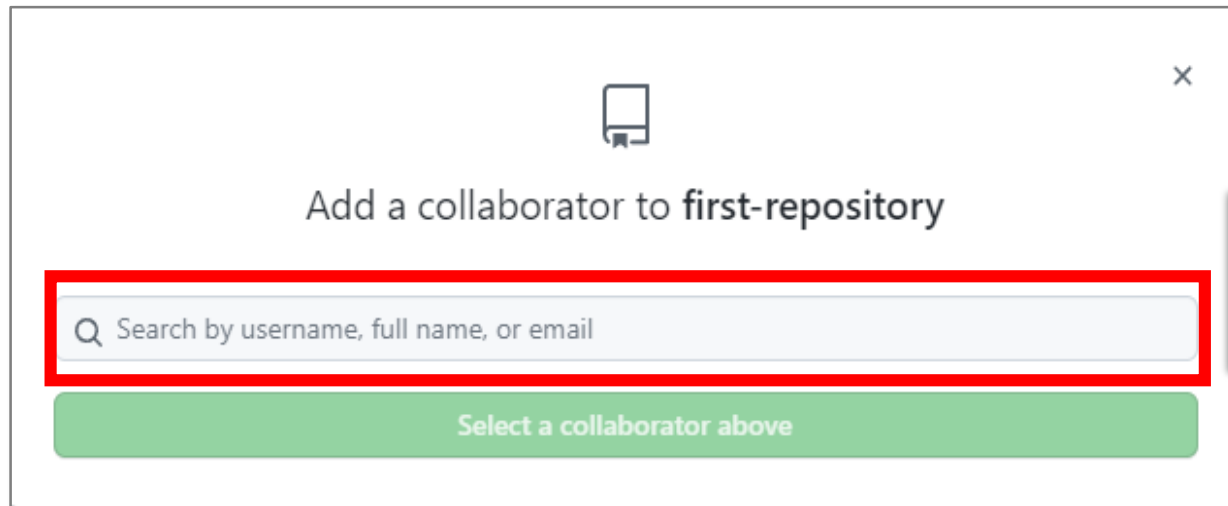
Manage access

You haven't invited any collaborators yet


Add people

원격 저장소 협력자 추가 방법

- 협력자의 GitHub 계정 ID 또는 이메일 주소를 입력한 후 하단의 “Select a collaborator above” 버튼을 클릭



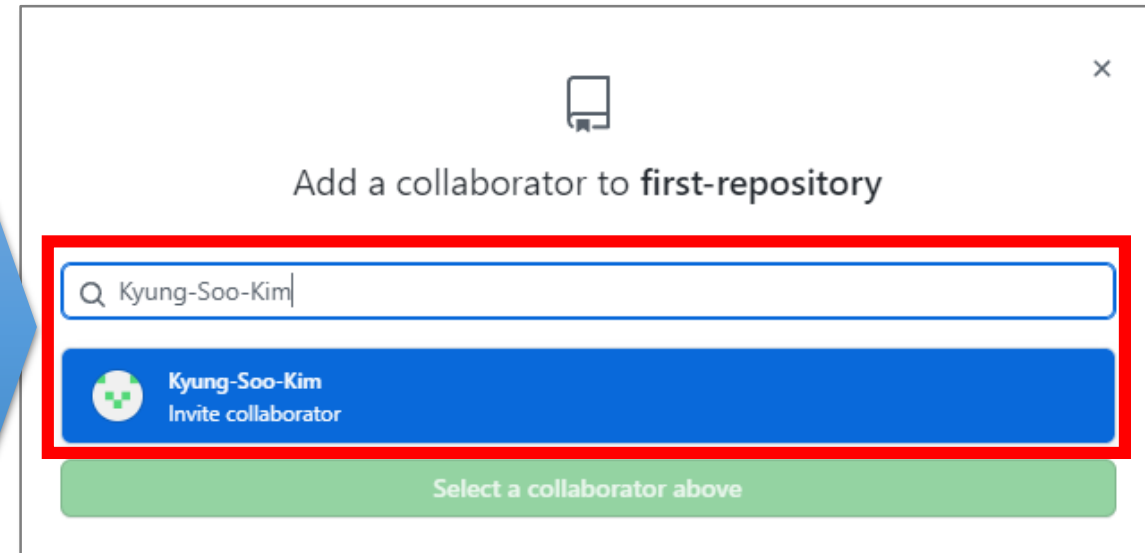
✕




Add a collaborator to first-repository

Q Search by username, full name, or email

Select a collaborator above




✕




Add a collaborator to first-repository

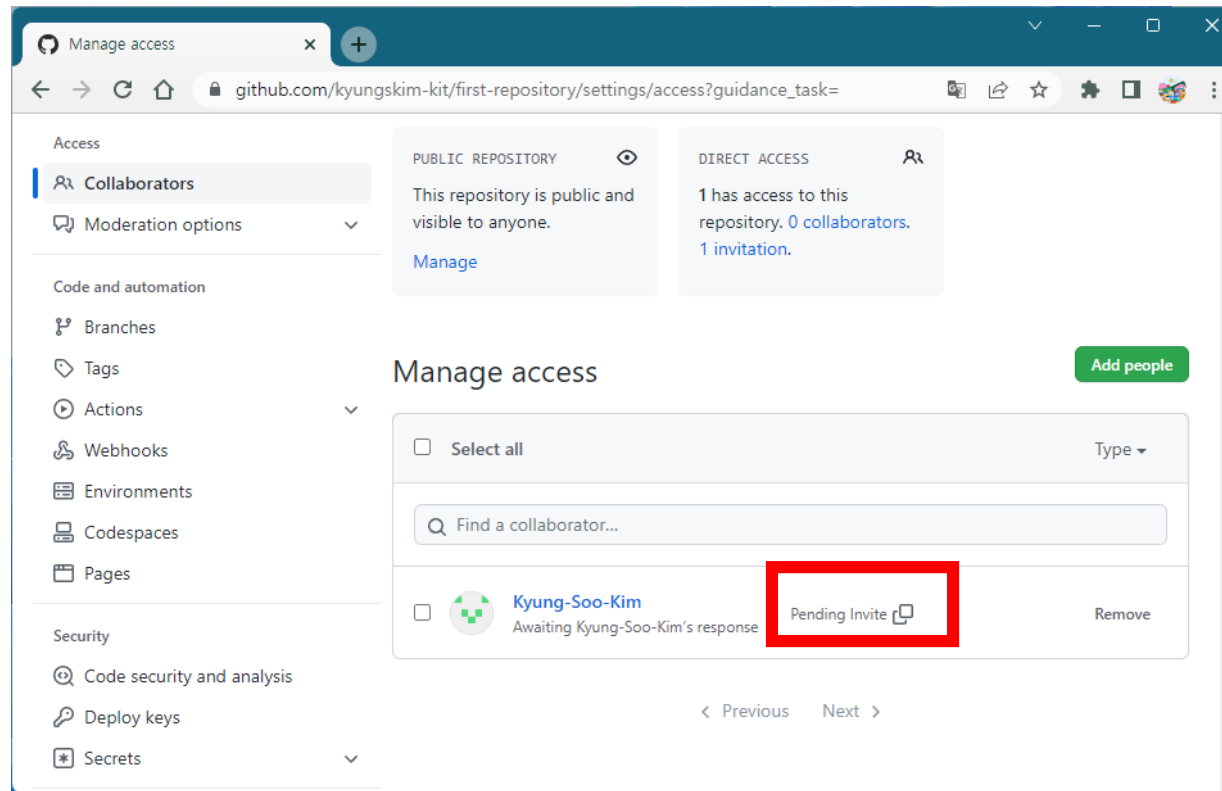
Q Kyung-Soo-Kim

 Kyung-Soo-Kim
Invite collaborator

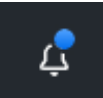
Select a collaborator above

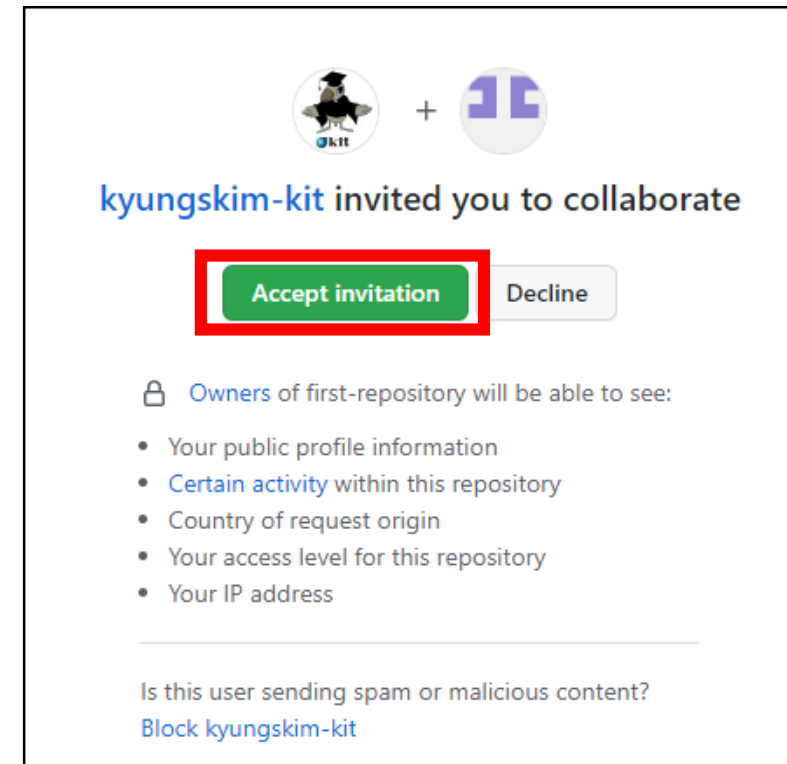
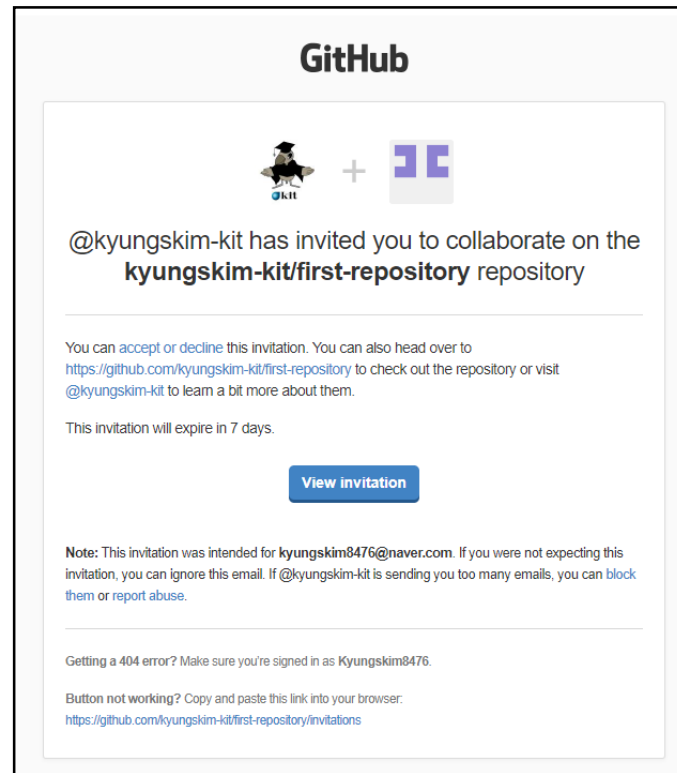
원격 저장소 협력자 추가 방법

- “Manage access”에 추가된 협력자가 표시됨.
- 해당 협력자를 초대하기 위해 “Pending Invite” 옆의  버튼을 클릭하면 협력자 초대를 위한 링크가 복사됨.



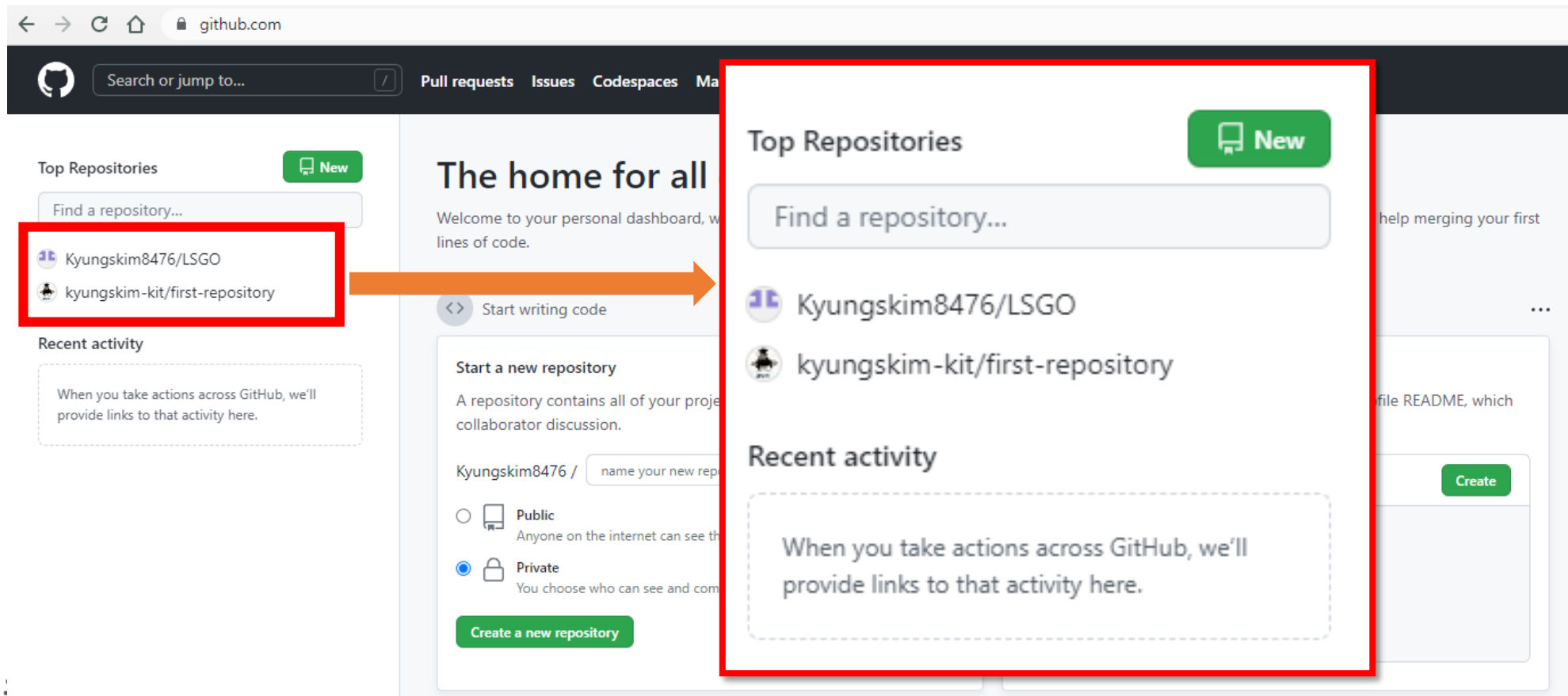
원격 저장소 협력자 추가 방법

- 협력자 초대 메일이 초대된 사용자에게 전송되며, 앞서 복사한 링크를 초대된 사용자에게 개별적으로 전송해도 무방함. (※ 이메일로 전송되지 않으면 본인의 GitHub 계정의 오른쪽 상단의 버튼  을 클릭하면 초대 메일을 확인할 수 있음.)
- 초대받은 협력자는 “Accept invitation”을 버튼을 클릭하여 해당 프로젝트에 협력자로 활동할 수 있음.



원격 저장소 협력자 추가 방법

- 해당 원격 저장소에 대한 협력자로 최종 지정되면 아래 화면과 같이 나의 저장소 목록에 해당 저장소의 이름이 나타남.



협력자로서 프로젝트 기여 방법

- 협력자 권한을 부여 받은 프로젝트의 원격 저장소에 대해서는 관리자와 마찬가지로 **commit**과 **push**, **pull** 명령을 통해 원격 저장소에 파일들을 추가, 수정 및 삭제할 수 있음.
- 협력자로서 프로젝트 기여 방법
 - 참여하는 프로젝트의 원격 저장소를 내 컴퓨터에 **clone**
 - Clone한 workspace에서 **commit**과 **push** 명령 수행
 - 원격 저장소의 내용이 변경된 경우 **pull** 명령을 통해 원격 저장소와 로컬 저장소를 동기화

협력자로서 프로젝트 기여 방법 예제 (1)

- 예제 설명

- 프로젝트(원격저장소) 명: GitStudy
- 관리자(supervisor): `main_user`
- 협력자(collaborator): `sub_user`

- 예제 상황 1

- 협력자(`sub_user`)가 관리자(`main_user`)의 GitHub 프로젝트 “GitStudy”에 새로운 파일 “`hello_world.txt`”를 업로드하려 한다.

협력자로서 프로젝트 기여 방법 예제 (1)

- 프로젝트 “GitStudy”의 원격 저장소를 내 컴퓨터에 **clone**
- Clone의 결과로 생성된 디렉토리 “GitStudy”로 이동하면 해당 원격 저장소와 연동된 workspace가 생성되었음을 확인할 수 있음.

```
KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/linuxstudy
$ git clone https://github.com/main_user/GitStudy.git
Cloning into 'GitStudy'...
warning: You appear to have cloned an empty repository.

KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/linuxstudy
$ cd GitStudy

KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/linuxstudy/GitStudy (main)
$ .....
```

협력자로서 프로젝트 기여 방법 예제 (1)

- 해당 workspace (폴더 “GitStudy”)에서 새로운 파일 “hello_world.txt”를 생성하여 내용을 작성한 후 **add**와 **commit** 명령을 수행

※ 어떤 협력자가 커밋한 것인가를 명확히 표시하기 위해 commit message 작성 시 협력자의 ID(본 예제에서는 sub_user)를 함께 기재하는 것을 추천.

```
KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/linuxstudy/GitStudy (main)
$ echo "Collaborator and Contribution in GitHub" > hello_world.txt

KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/linuxstudy/GitStudy (main)
$ git add hello_world.txt
warning: in the working copy of 'hello_world.txt', LF will be replaced by CRLF the next time Git touches it

KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/linuxstudy/GitStudy (main)
$ git commit -m "first commit by sub_user"
[main (root-commit) 44d41df] first commit by sub_user
1 file changed, 1 insertion(+)
create mode 100644 hello_world.txt

KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/linuxstudy/GitStudy (main)
$ git status
On branch main
Your branch is based on 'origin/main', but the upstream is gone.
(use "git branch --unset-upstream" to fixup)

nothing to commit, working tree clean
```

협력자로서 프로젝트 기여 방법 예제 (1)

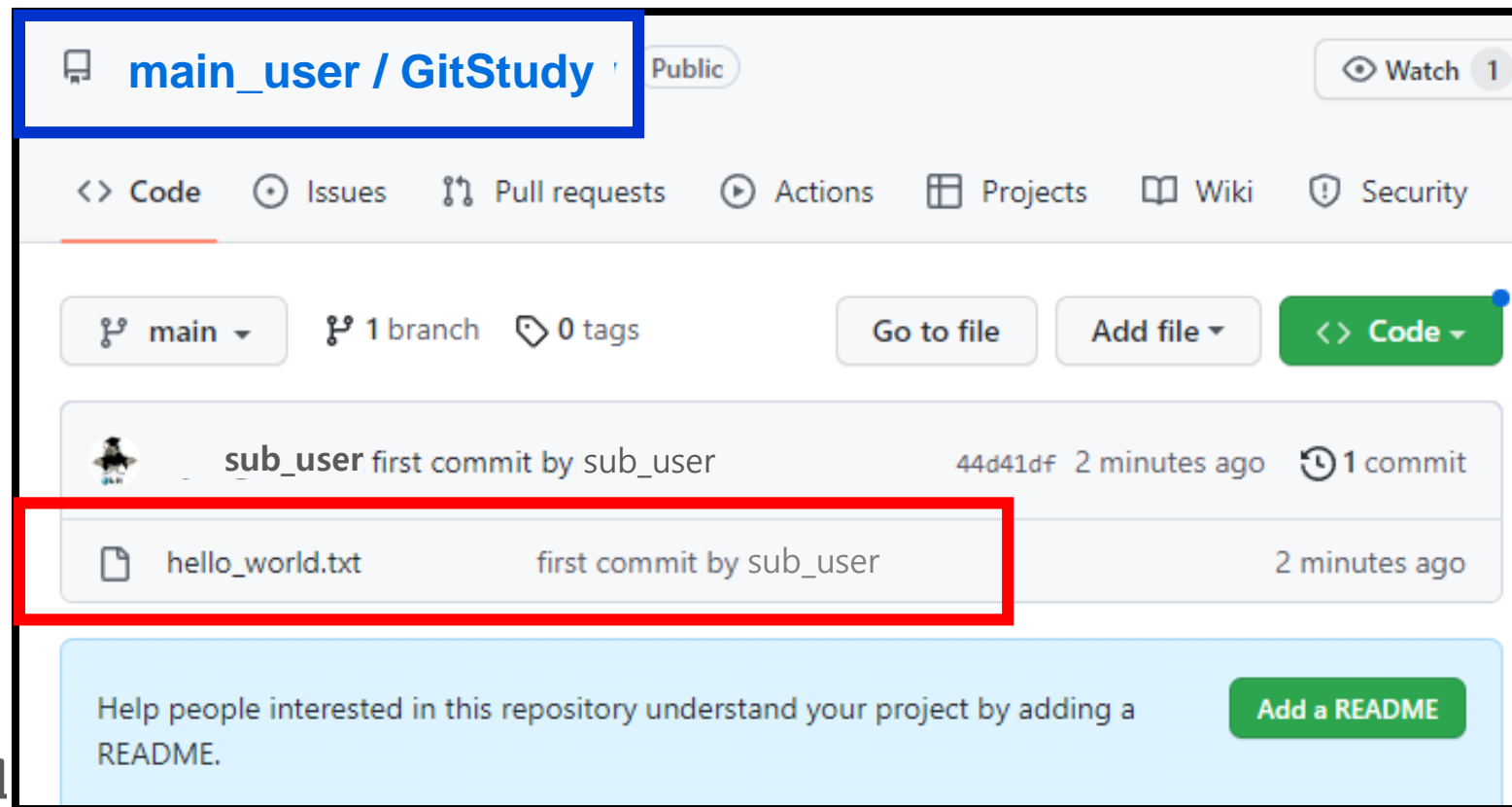
- “**git status**” 명령을 통해 **add**와 **commit** 명령이 올바르게 수행되었는지 확인
- Commit이 완료된 파일을 연동된 원격 저장소로 **push**

```
KyungSookKim@DESKTOP-M68HRBF MINGW64 /c/linuxstudy/GitStudy (main)
$ git push
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 264 bytes | 264.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/main_user/GitStudy.git
 * [new branch]      main -> main

KyungSookKim@DESKTOP-M68HRBF MINGW64 /c/linuxstudy/GitStudy (main)
$ |
```

협력자로서 프로젝트 기여 방법 예제 (1)

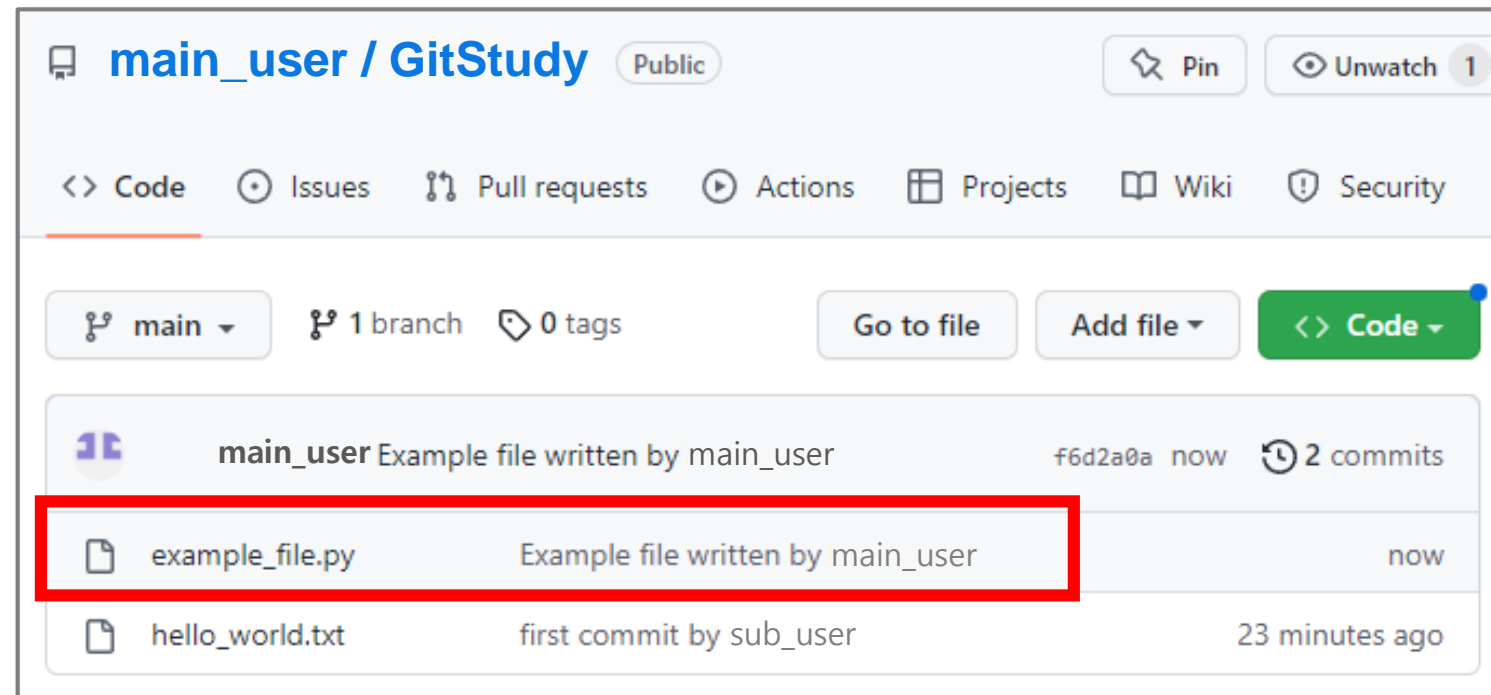
- 관리자 “main_user”의 원격 저장소 “GitStudy”에 접속하여 확인하면, 아래와 같이 협력자 “sub_user”가 push한 파일이 성공적으로 업로드 되었음을 확인할 수 있음.



협력자로서 프로젝트 기여 방법 예제 (2)

• 예제 상황 2

- 관리자 “main_user”가 자신의 원격 저장소에 새로운 파일 “example_file.py”를 업로드하였다.
- 이 때, 협력자 “sub_user”가 해당 원격 저장소의 변경된 사항을 **pull** 연산을 통해서 자신의 로컬 저장소와 workspace에 반영할 수 있는가?



협력자로서 프로젝트 기여 방법 예제 (2)

- 협력자 “sub_user”의 git bash에서 “**git pull**” 명령을 수행하면 아래와 같이 원격 저장소에 새롭게 추가된 파일 “example_file.py”가 협력자의 workspace에 반영됨을 확인할 수 있음.

```
KyungSookKim@DESKTOP-M68HRBF MINGW64 /c/linuxstudy/GitStudy (main)
$ git pull
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3). 821 bytes | 273.00 KiB/s, done.
From https://github.com/main_user/GitStudy
   44d41df..f6d2a0a  main      -> origin/main
Updating 44d41df..f6d2a0a
Fast-forward
 example_file.py | 13 +++++
 1 file changed, 13 insertions(+)
 create mode 100644 example_file.py
```


협력자로서 프로젝트 기여 방법 예제 (3)

- 예제 상황 3

- 협력자 “kyunskim-kit”가 파일 “example_file.py”를 자신의 workspace에서 삭제하고 이를 원격 저장소에 반영하려 함.

```
KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/linuxstudy/GitStudy (main)
$ rm example_file.py

KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/linuxstudy/GitStudy (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        deleted:    example_file.py

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


협력자로서 프로젝트 기여 방법 예제 (3)

- 협력자 “sub_user”가 파일 “example_file.py”을 삭제한 작업에 대한 **add**와 **commit**을 수행함.

```
KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/linuxstudy/GitStudy (main)
$ git add example_file.py

KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/linuxstudy/GitStudy (main)
$ git commit -m "Delete example_file.py by sub_user"
[main 12e2de3] Delete example_file.py by sub_user
1 file changed, 13 deletions(-)
delete mode 100644 example_file.py
```

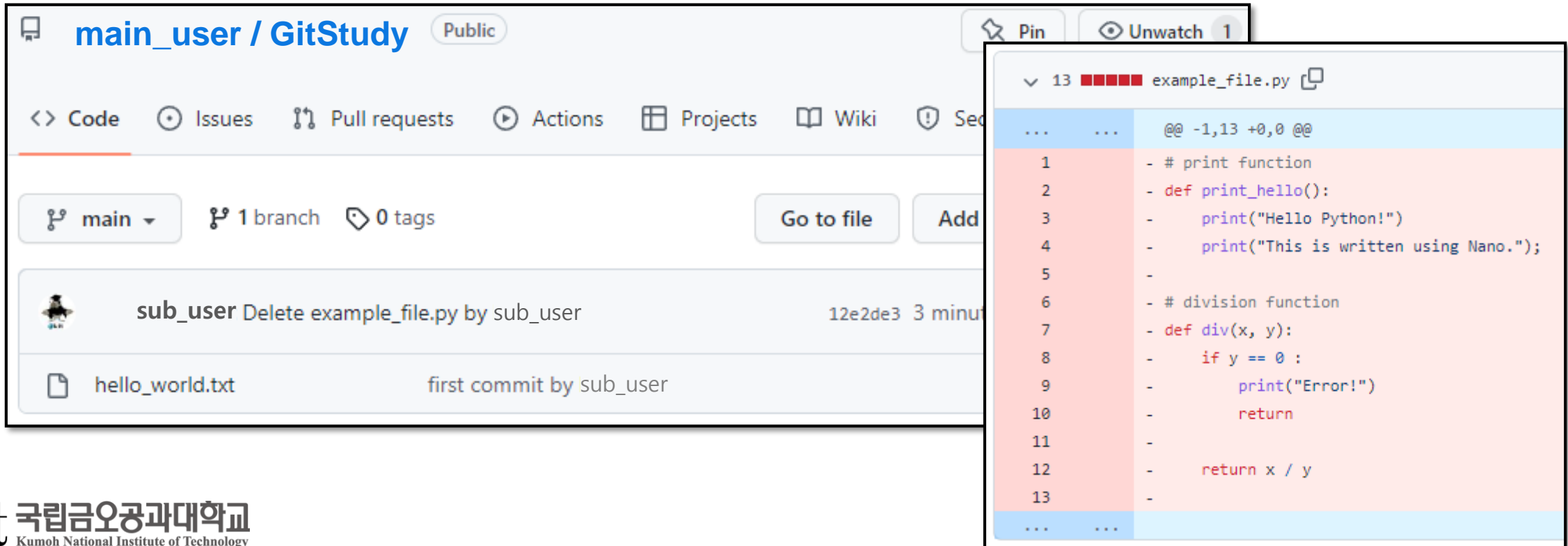
협력자로서 프로젝트 기여 방법 예제 (3)

- 협력자 “sub_user”가 “example_file.py” 파일의 삭제 작업에 대한 commit 결과를 관리자(main_user)의 원격 저장소 “GitStudy”에 반영하기 위해 **push**를 수행함.

```
KyungSooKim@DESKTOP-M68HRBF MINGW64 /c/linuxstudy/GitStudy (main)
$ git push
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 20 threads
Compressing objects: 100% (1/1), done.
Writing objects: 100% (2/2), 254 bytes | 254.00 KiB/s, done.
Total 2 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/main_user/GitStudy.git
   f6d2a0a..12e2de3  main -> main
```

협력자로서 프로젝트 기여 방법 예제 (3)

- 관리자(main_user)의 원격 저장소 “GitStudy”에 접속하여 확인하면 아래와 같이 협력자 “sub_user”가 “example_file.py” 파일을 삭제한 결과가 원격 저장소에도 반영되었음을 확인할 수 있음.



The screenshot displays the GitHub interface for the repository 'main_user / GitStudy'. The 'Code' tab is selected. A commit by 'sub_user' is highlighted, titled 'Delete example_file.py by sub_user'. The commit message indicates the deletion of the file. The file 'example_file.py' is shown in the commit details, with its content displayed. The content includes a print function and a division function.

```
... @@ -1,13 +0,0 @@  
1 - # print function  
2 - def print_hello():  
3 -     print("Hello Python!")  
4 -     print("This is written using Nano.");  
5 -  
6 - # division function  
7 - def div(x, y):  
8 -     if y == 0 :  
9 -         print("Error!")  
10 -     return  
11 -  
12 -     return x / y  
13 -
```

GitHub 프로젝트 협업 실습

GitHub 프로젝트 협업 실습 (조별 활동)

- ① 본 실습을 위해 모든 조원들은 각자의 계정에 새로운 원격 저장소를 생성한다. 이 때 실습에 사용되는 원격 저장소 이름은 조원들끼리 서로 달라야 한다.
- ② 조원 1명을 관리자(supervisor)로 지정한다. 나머지 조원들은 협력자(collaborator)가 된다.
- ③ 관리자는 다른 조원들을 자신의 실습용 원격 저장소에 대한 협력자로 지정한다.
- ④ 협력자 조원들은 관리자의 원격 저장소를 clone한 후 해당 원격 저장소에 프로그래밍 수업시간에 작성한 코드들을 add, commit, push하는 작업을 수행한다.
- ⑤ 관리자는 원격 저장소에 접속하여 직접 새로운 파일을 추가한다.
- ⑥ 협력자 조원들은 pull 명령을 통해 관리자가 새롭게 추가한 파일이 자신의 workspace에 반영되는지 확인한다.
- ⑦ 상기 ②~⑥의 실습을 돌아가면서 진행한다. 즉, 모든 조원이 관리자 역할을 한 번씩 수행해야 한다.

GitHub 프로젝트 협업 실습 (조별 활동)

- 실습 시 유의사항

- ① 실습을 위한 GitHub 계정은 본 수업용 계정을 사용할 것.
- ② 누가 수행한 commit인지 손쉽게 알아볼 수 있도록 commit message에는 자신의 영어 이름 포함할 것.

(예) `git commit -m “committed by kyungsoo-kim”`

- ③ 실습에 사용할 파일은 프로그래밍 시간에 작성한 코드를 사용할 것.

Assignment #2

- “GitHub 프로젝트 협업 실습 (조별활동)”의 실습 과정과 결과를 실습 보고서로 상세히 작성하여 LMS에 제출한다.
- 자신이 “관리자”로서 수행한 작업들과 “협력자”로서 작업한 화면들을 캡처하여 보고서에 반드시 첨부해야 한다.
 - 단, 자신이 “협력자”로서 실습한 결과는 “협력자 1”과 “협력자 2”로 수행한 작업의 결과에 대해서만 첨부할 것.

Q & A