

5주차 결과보고서

전공 : 경영학과

학년 : 4학년

학번 : 20190808

이름 : 방지혁

1. 실습 제출물 설명

```
cse3013 — -bash — 80x24

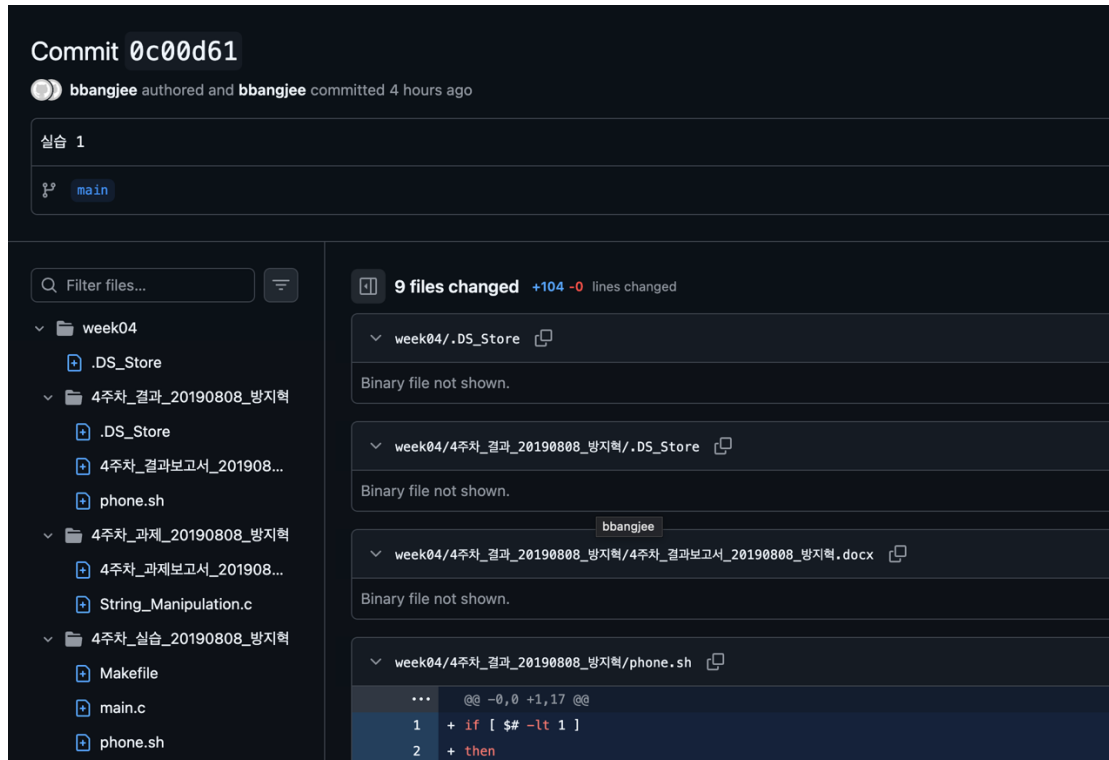
[(base) bbangjee-MacBook-Pro:cse3013 bangjee$ pwd
/Users/bangjee/Documents/cse3013
[(base) bbangjee-MacBook-Pro:cse3013 bangjee$ ls -al
total 24
drwxr-xr-x  5 bangjee  staff   160  9월 30 15:23 .
drwx-----@ 25 bangjee  staff   800  9월 30 16:51 ..
-rw-r--r--@  1 bangjee  staff 10244  9월 30 15:50 .DS_Store
drwxr-xr-x 12 bangjee  staff   384  9월 30 15:26 .git
drwxr-xr-x  6 bangjee  staff   192  9월 30 15:21 week04
[(base) bbangjee-MacBook-Pro:cse3013 bangjee$
```

보다시피 ls -al을 입력했을 때 .git이 생성되어 있는 것을 확인할 수 있습니다.

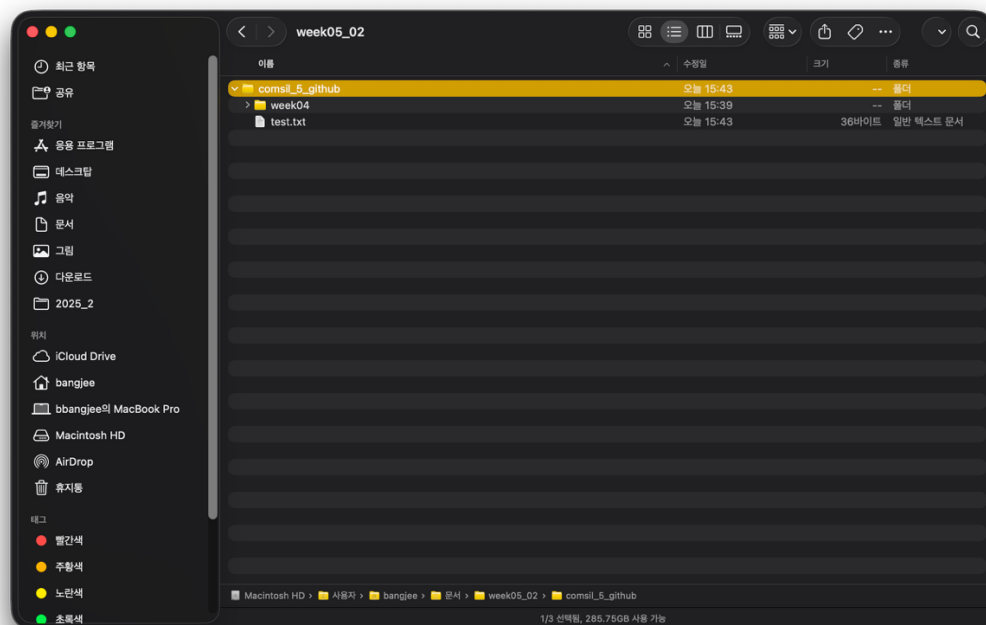
```
cse3013 — -bash — 80x24

[Password for 'https://github.com':
remote: No anonymous write access.
fatal: Authentication failed for 'https://github.com/bbangjee/comsil_5_github.git/'
[(base) bbangjee-MacBook-Pro:cse3013 bangjee$ git push origin --all
Username for 'https://github.com': 
[Password for 'https://%E3%85%A0@github.com':
remote: Invalid username or token. Password authentication is not supported for
Git operations.
fatal: Authentication failed for 'https://github.com/bbangjee/comsil_5_github.git/'
[(base) bbangjee-MacBook-Pro:cse3013 bangjee$ git push origin --all
Username for 'https://github.com': bbangjee
Password for 'https://bbangjee@github.com':
오브젝트 나열하는 중: 15, 완료.
오브젝트 개수 세는 중: 100% (15/15), 완료.
Delta compression using up to 8 threads
오브젝트 압축하는 중: 100% (14/14), 완료.
오브젝트 쓰는 중: 100% (15/15), 216.51 KiB | 27.06 MiB/s, 완료.
Total 15 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), done.
To https://github.com/bbangjee/comsil_5_github.git
 * [new branch]      main -> main
[(base) bbangjee-MacBook-Pro:cse3013 bangjee$
```

실습 1에서 push를 한 이후의 터미널입니다.



Commit 내역을 확인해보면 실습 1 commit 때 해당 파일들이 업로드 된 것을 확인할 수 있습니다.



이것은 실습 2에서 원격 저장소를 저장한 폴더입니다.

```
comsil_5_github -- -bash -- 80x24

음악
잡다한문서
취준
[(base) bbangjee-MacBook-Pro:Documents bangjee$ cd week05_02 ]
[(base) bbangjee-MacBook-Pro:week05_02 bangjee$ ls ]
comsil_5_github
[(base) bbangjee-MacBook-Pro:week05_02 bangjee$ ls -al ]
total 24
drwxr-xr-x  4 bangjee  staff   128  9월  30 15:40 .
drwx-----@ 25 bangjee  staff   800  9월  30 16:51 ..
-rw-r--r--@  1 bangjee  staff  10244  9월  30 15:53 .DS_Store
drwxr-xr-x  6 bangjee  staff   192  9월  30 15:43 comsil_5_github
[(base) bbangjee-MacBook-Pro:week05_02 bangjee$ cd comsil_5_github/ ]
[(base) bbangjee-MacBook-Pro:comsil_5_github bangjee$ ls ]
test.txt      week04
[(base) bbangjee-MacBook-Pro:comsil_5_github bangjee$ ls -al ]
total 24
drwxr-xr-x  6 bangjee  staff   192  9월  30 15:43 .
drwxr-xr-x  4 bangjee  staff   128  9월  30 15:40 ..
-rw-r--r--@  1 bangjee  staff  6148  9월  30 15:40 .DS_Store
drwxr-xr-x 13 bangjee  staff   416  9월  30 15:43 .git
-rw-r--r--@  1 bangjee  staff    36  9월  30 15:43 test.txt
drwxr-xr-x  6 bangjee  staff   192  9월  30 15:39 week04
(base) bbangjee-MacBook-Pro:comsil_5_github bangjee$
```

보다시피 .git이 생성되어 있는 것을 확인할 수 있습니다. test.txt라는 파일을 하나 더 추가했습니다.

```
comsil_5_github -- -bash -- 80x24

자동으로 설정했습니다. 이 정보가 맞는지 확인하십시오. 이 메시지를
보지 않으려면 정보를 명시적으로 설정하십시오:

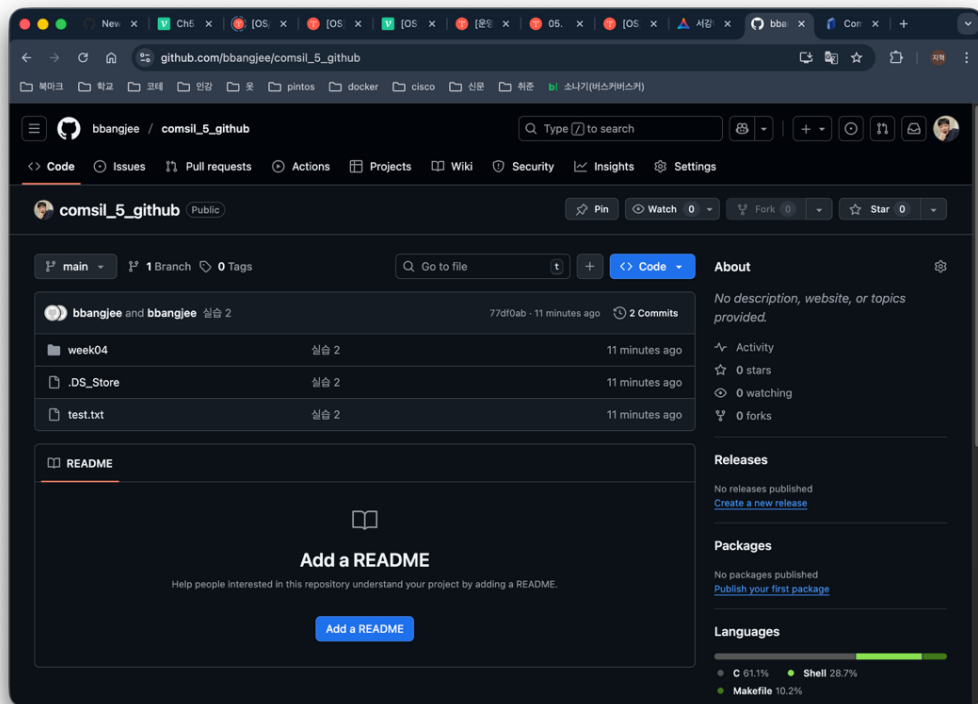
git config --global user.name "내 이름"
git config --global user.email you@example.com

이렇게 한 다음, 이 커밋에 사용한 신원 정보를 다음과 같이 해서 바꿀 수 있습니다:

git commit --amend --reset-author

4 files changed, 2 insertions(+)
create mode 100644 .DS_Store
create mode 100644 test.txt
[(base) bbangjee-MacBook-Pro:comsil_5_github bangjee$ git push ]
오브젝트 나열하는 중: 13, 완료.
오브젝트 개수 세는 중: 100% (13/13), 완료.
Delta compression using up to 8 threads
오브젝트 압축하는 중: 100% (7/7), 완료.
오브젝트 쓰는 중: 100% (8/8), 1.13 KiB | 1.13 MiB/s, 완료.
Total 8 (delta 4), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (4/4), completed with 3 local objects.
To https://github.com/bbangjee/comsil_5_github
   0c00d61..77df0ab  main -> main
(base) bbangjee-MacBook-Pro:comsil_5_github bangjee$
```

이를 다시 push해줍니다.



최종 결과인 원격 저장소는 이렇습니다.

2. 협업을 할 때 Github를 사용하는 것이 어떤 장점이 있는지 기술하여라

우선 우리가 잘 알다시피 버전관리 및 변경 이력 추적이 쉽습니다. 수정2, 수정3, 찌찌최종 이렇게 일일이 파일을 여러 개 만들 필요가 없습니다. 누가, 언제, 어떤 코드를 수정했는지 모든 기록이 남기에 책임 소재를 찾기 쉽고 누군가 큰 실수를 저질러도 이전 버전으로 쉽게 되돌릴 수 있습니다. 또한, 각 commit마다 왜 바꿨는지 설명을 남길 수 있기에 가독성도 좋습니다.

다음으로는 동시 작업이 가능합니다. 각자 다른 branch에서 여러 사람이 동시에 다른 기능을 개발할 수도 있기 때문입니다. 나중에 merge가 가능하기 때문입니다. 또한 여러 사람이 같은 파일을 수정할 때 git의 merge가 자동으로 합쳐주는데 코드의 어떤 줄에서 충돌이 일어나는지 알려주기 때문에 해결하기 쉽습니다.

또한 코드 리뷰가 가능합니다. Pull request 기능이 있기 때문에 합치기 전에 팀원 혹은 상급자들이 피드백을 하며 검토가 가능합니다.

무엇보다도 오픈소스 활용이 용이해집니다. 다른 사람의 코드를 fork해서 개선할 수도 있으며 내가 직접 다른 오픈소스를 내 저장소에 받기도 용이합니다.