**Git/Github 학습 보고서**

**4조 (김미주, 신현지, 엄다연, 이수경)**

1. 학습목표

1) git으로 소스를 다루는 skill

2) github로 협업하는 skill

이 두가지 목표를 가지고 Git/Github 학습을 진행하였다.

2. Git을 사용하는 이유

코드 변경이 일어난 지점을 기록할 수 있으며 변경된 코드뿐만 아니라 메시지도 기록할 수 있다.

또한, 변경 전으로 쉽게 돌아갈 수 있다.

3. Git의 역사관리 기능의 이해

git download를 통해 먼저 git을 설치를 받아야 한다. 이를 설치한 후에는 git 명령이 가능하다.

commit이 정상적으로 되지 않을 경우를 대비하여 user.name과 user.email config 설정을 해줘야 한다.

* mkdir git-test : 테스트 폴더 만들기
* git init: git 초기화 (새로운 프로젝트 시작할 때 default로 생김)
  + 현재 폴더 안에 있는 파일을 보면 .git이 자동으로 생성되며, 이것은 git을 사용하는데 관련된 파일이다.
* 임의의 소스파일 생성 후 그 소스코드에 대해 역사분을 만들 준비를 한다.
* add와 commit을 통해 역사를 만든다.
* git add : 변화분 만들기를 준비하는 것으로, gid add 명령어를 사용하여 git이 해당 파일을 추적하도록 만든다. Git status를 입력하면 new file이 생긴 것을 확인 할 수 있다.
* git commit: 변화분을 저장하는 것으로, 해당 명령어를 사용하여 파일 변경 내용에 대한 메시지를 입력한다.
* Git log: 지난 commit에 대한 로그를 확인할 수 있다.
* Shortlog: commit 정보를 확인할 수 있다.
* git diff: 수정된 부분 차이점을 확인 가능하다. > 커밋할 변화분

4. Git으로 소스 다루기: 오픈소스 uftrace 프로젝트 살펴보기

* git clone > 프로젝트를 내려받을 수 있다.

소스코드뿐만 아니라, git 히스토리 까지 같이 받는다.

* blame> 소스 라인 단위로 히스토리 확인 가능하다.

5. Git으로 소스 다루기: 소스코드 변화내용 다루기

* git reset> add작업에 대한 취소 가능
* git stash> 임시저장 가능
* git stash pop> 다시 가지고와서 현재 branch에 적용
* git checkout -- 파일명 > 가장 최신 히스토리 기반으로 복구

6. git으로 소스 다루기: branch 이해와 실습

branch란? 기존 branch에서 독립하여 가지치는 기능이다. 이는 경로는 같지만 다른 공간으로 볼 때 사용하며 각각의 branch는 다른 branch의 영향을 받지 않는다.

* Git checkout branch name: 해당 branch로 이동하는 기능이다.
* Git branch branch name : branch를 만드는 명령이다.
* Git branch -d branch name : branch를 제거하는 명령이다.

7. Git으로 소스 다루기: 타임머신기능 이해와 실습

타임머신 기능을 통해 과거로 돌아갈 수도 있다.

8. Git으로 소스 다루기: git-blame 실습

이 함수는 어떻게 만들어졌을까, 이 클래스는 어떻게 만들어졌을까 등을 추적할 때 사용한다.

9. Github로 협업 : Pull request(내가 만든 commit을 제출하기)

소스의 수정부분을 제출하는 것으로, official한 node.js에서 fork를 뜨는 것부터 시작이다.

즉, 예를 들어 오픈소스로 운영되고 있는 Node.js가 있다면 fork를 떠서 내 계정 안에 Node.js가 생기면 clone을 받아서 그 내용에 대해 작업을 하고 브랜치를 잡아서 push를 한 후, 깃허브의 기능인 pull request를 통해서 내가 만든 commit을 official한 프로젝트에 제출하는 방식이다.

10. Git/Github 협업: Rebase 필요한 상황 이해

많은 사람들이 개발할 경우에는 Base의 갱신이 반드시 필요하다.

따라서, 팀프로젝트에서 새롭게 추가된 내용을 가져오기 위해 fetch 명령어를 사용한다.

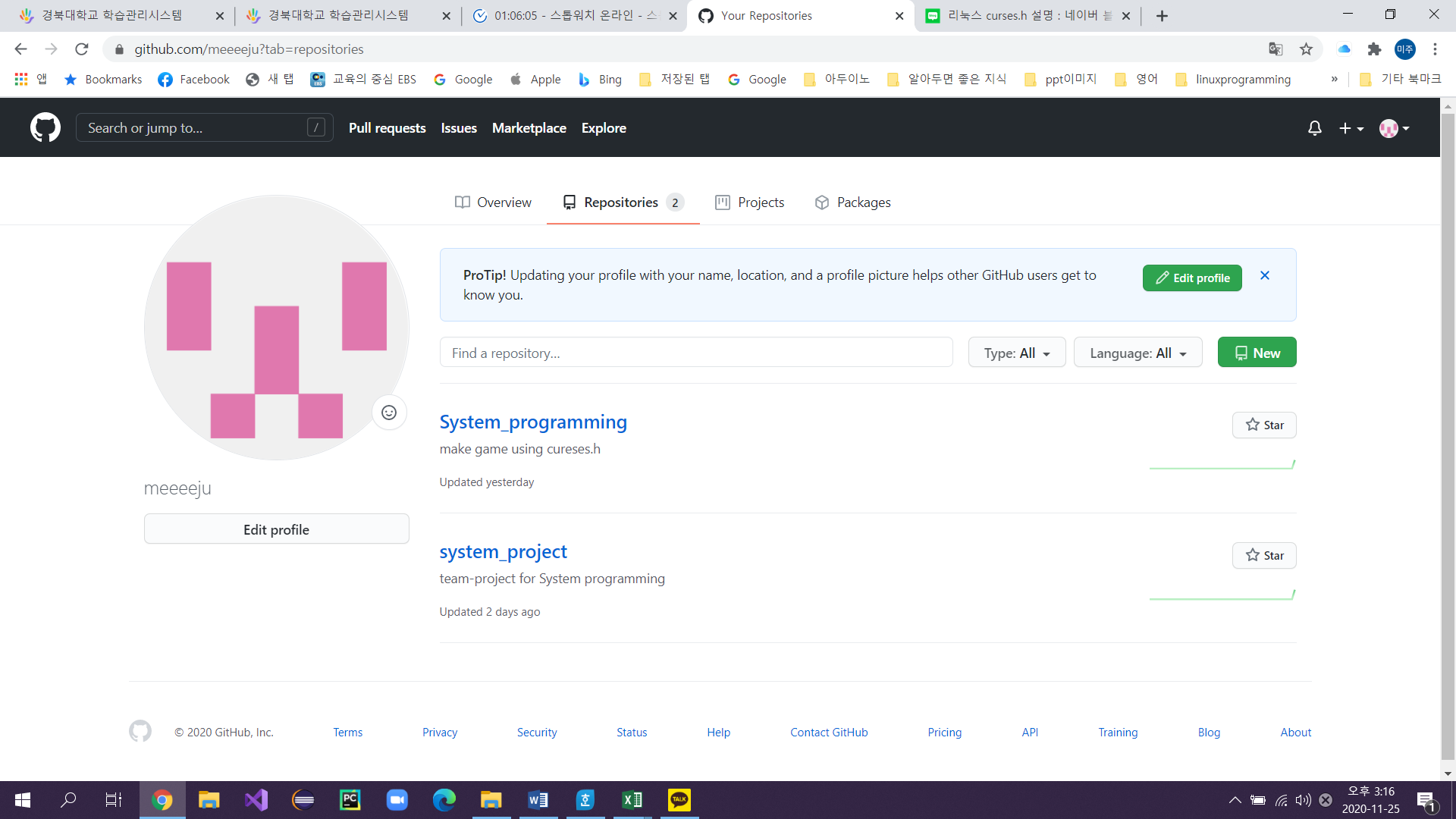
(pull로도 가져올 수 있으나 pull은 현재 브랜치에 반드시 merge를 동반하므로, fetch 명령어를 사용한다.)

fetch로 가져오면 바로 merge하지 않고 .git에 저장을 해둔다. 그 상태에서 rebase 명령을 하게 되면 내가 만든 commit에 대해 일단 rewind(되감기)를 한다.

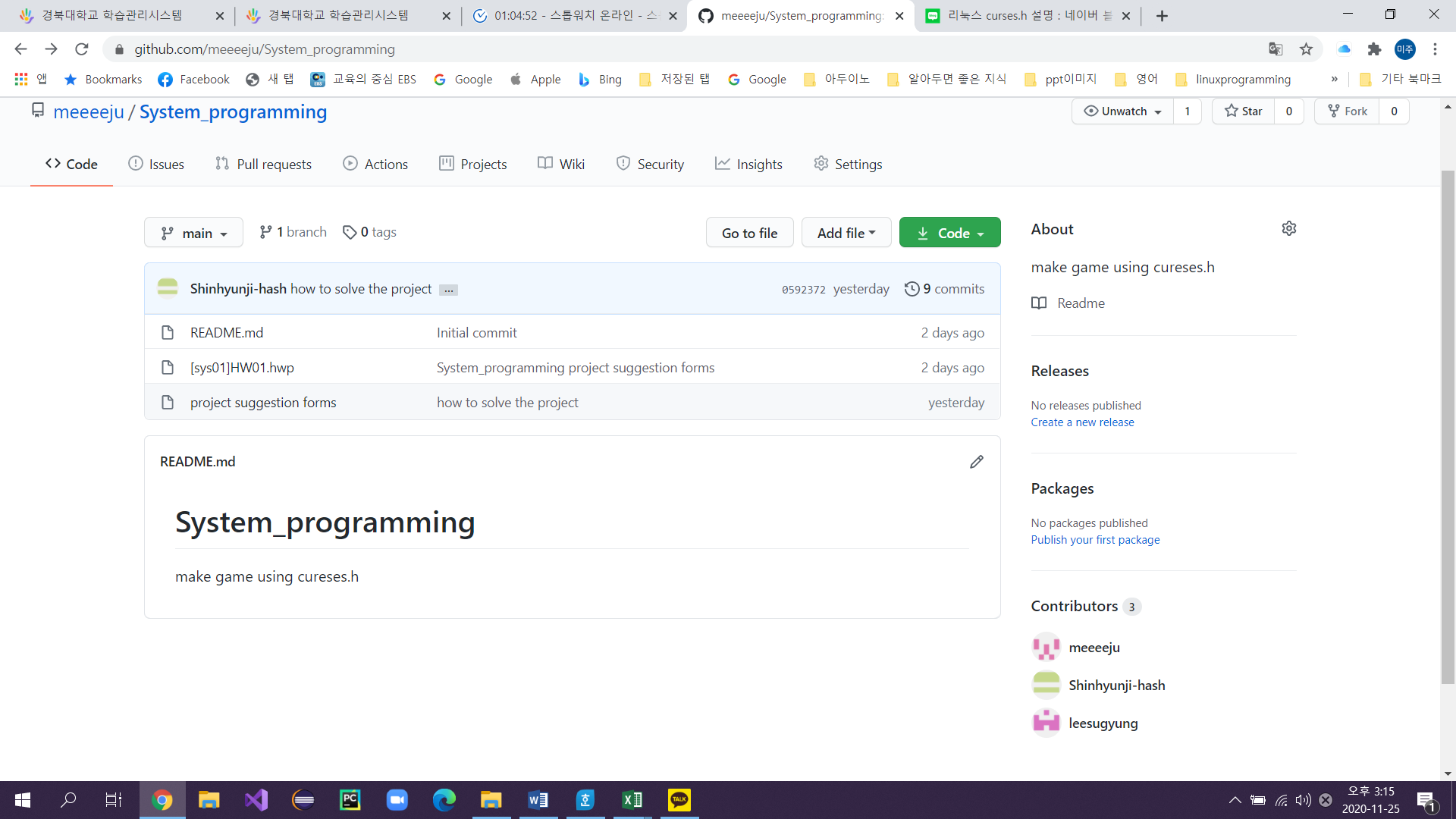
그리고 base를 교체한 이후에 올리는 방식으로 진행된다.

12. 실습 해보기

-계정 생성하여 repository 만들어보기



- 공동 작업 환경 개설하기



11. 결론

평소에 git/github을 사용하는 이유나, 사용법에 대해 정확히 알지 못하였는데 이번 git/github 학습을 바탕으로 git으로 협업하는 방법에 관한 정보를 습득하였다. 따라서, 이를 바탕으로 앞으로 진행할 프로젝트를 github를 이용한 협업을 통하여 효율적으로 진행해 볼 예정이다.