

소프트웨어공학

Git 공동작업

Kwangwoon Univ.
Dept. of Computer Engineering
Ki-Hoon Lee



Requirements

■ Softcopy

- 보고서: **2020. 05. 22(금) 23:58**까지 Klas로 제출(개인별 제출)
 - Due Delay 없음
 - 본인이 구현한 소스코드가 포함된 개인별 보고서만 제출
 - 보고서 파일 명은 **SE_학번**으로 작성
 - e.g. SE_2017XXXXXX.pdf [보고서는 PDF로 변환하여 제출](#)
- 소스코드: **2020. 05. 22(금) 23:58**까지 gitHub상에 올라와 있는 코드로 확인
 - 마감 날짜 이후에 코드를 push하거나 Readme.md나 wiki등 수정한 내역이 있을 경우 감점
 - 소스코드는 주어진 스켈레톤 코드(main.c)를 수정해서 사용할 것
 - Repository 이름은 **Project2**로 작성하고, 공개범위는 반드시 **public**으로 할 것



Project 2

■ 파일 비교 프로그램 구현

- linux **ubuntu(18.04 LTS)**상에서 두 개의 파일을 비교하는 프로그램 구현
- 다음과 같은 기능을 구현하되, milestone과 issue를 이용해서 개발을 진행
 1. 파일 정보 받아 오기
 2. 파일 시간 받아 오기
 3. 파일 사이즈 비교
 4. 파일 블록 수 비교
 5. 파일 수정 날짜(월/일) 비교
 6. 파일 수정 시간(시/분) 비교
- 각각의 기능별로 milestone을 생성하고 해당 milestone내의 to-do를 issue로 관리
- 코드 취합은 **반드시 git을 사용**할 것
 - commit하지 않은 팀원이 존재할 경우 감점
 - 코드를 통째로 올렸을 경우 감점



Project 2

■ 파일 비교 프로그램 구현

- 스켈레톤 코드 내부의 함수를 작성하여 commit할 것

```

1  #include <stdio.h>
2  #include <sys/types.h>
3  #include <sys/stat.h>
4  #include <unistd.h>
5  #include <time.h>
6
7  struct stat stat1, stat2;
8  struct tm *time1, *time2;
9
10 void filestat1(void);
11 void filestat2(void);
12 void filetime1(void);
13 void filetime2(void);
14 void sizecmp(void);
15 void blockcmp(void);
16 void datecmp(void);
17 void timecmp(void);
18
19 int main(void)
20 {
21     filestat1();
22     filestat2();
23     filetime1();
24     filetime2();
25     sizecmp();
26     blockcmp();
27     datecmp();
28     timecmp();
29 }

```

31 //파일 1의 정보를 가져오는 함수 작성	51 //두 개의 파일 크기를 비교하는 함수 작성
32 void filestat1(void)	52 void sizecmp(void)
33 {	53 {
34 }	54 }
35	55
36 //파일 2의 정보를 가져오는 함수 작성	56 //두 개의 파일 블록 수를 비교하는 함수 작성
37 void filestat2(void)	57 void blockcmp(void)
38 {	58 {
39 }	59 }
40	60
41 //파일 1의 시간 정보를 가져오는 함수 작성	61 //두 개의 파일 수정 날짜를 비교하는 함수 작성
42 void filetime1(void)	62 void datecmp(void)
43 {	63 {
44 }	64 }
45	65
46 //파일 2의 시간 정보를 가져오는 함수 작성	66 //두 개의 파일 수정 시간을 비교하는 함수 작성
47 void filetime2(void)	67 void timecmp(void)
48 {	68 {
49 }	69 }
50	

각각의 함수를 팀원별로 알맞게 분배하여 구현할 것
e.g.) a는 1,2번 함수 구현, b는 3, 4번 함수 구현...



Project 2

- 파일 비교 프로그램 구현
 - 파일 블록 수 비교

```
//두 개의 파일 블록 수를 비교하는 함수 작성  
void blockcmp(void)  
{  
}
```

- ※ block size?
 - 데이터 또는 프로그램의 한 블록 크기이며, 정보를 처리하는 논리적인 단위로 보통 레코드의 집합으로 구성됨



Project 2

■ 결과 화면

- e.g.) 파일 크기 비교 함수에서는 두 개의 파일을 비교하고 어느 파일이 더 큰지(or 작은지) 출력
 - 둘 중 하나가 크면 "OOO is bigger" 출력
 - 같으면 "sizes are equal" 출력
- e.g.) 파일 수정 시간 비교 함수에서는 두 개의 파일을 비교하고 어느 파일이 더 최근에 수정했는지 출력
 - 둘 중 하나가 빠르면 "OOO is early" 출력
 - 같으면 "same time" 출력
- 첫 번째 파일 이름은 text1로, 두 번째 파일 이름은 text2로 지정할 것



Project 2

■ 결과 화면

- 프로그램을 실행한 결과와 ls -l의 결과를 모두 캡처할 것

```
root@linkaden:~/2014722064_OSS# ./main
size compare
text2 is bigger

block compare
text2 is bigger

date compare
same date

time compare
same time
```

프로그램 실행 결과

```
root@linkaden:~/2014722064_OSS# ls -l
total 40
-rwxr-xr-x 1 root root 9176 2월 12 16:15 main
-rw-r--r-- 1 root root 2111 2월 12 16:15 main.c
-rwxr-xr-x 1 root root 8824 2월 12 16:48 test
-rw-r--r-- 1 root root 476 2월 12 16:47 test.c
-rw-r--r-- 1 root root 38 2월 12 16:13 text1
-rw-r--r-- 1 root root 1977 2월 12 16:13 text2
root@linkaden:~/2014722064_OSS#
```

ls -l의 실행 결과

파일 사이즈

수정 날짜와 시간



Project 2

■ gitHub 진행 방법

- 팀장은 gitHub public repository를 생성
- 팀원들은 해당 repository를 fork하고 checkout하여 작업을 진행
- **코드 취합은 반드시 git의 fork & pull request를 이용하여 진행**
- 코드를 한번에 통째로 올리지 말 것
 - e.g.) 각각의 함수를 작성한 뒤 복사 및 붙여넣기를 사용해 텍스트 파일로 수동으로 합쳐서 한 파일로 올리지 말 것
- Pull request를 통하지 않고 master branch에 바로 push할 경우 감점
- pull request를 보냈을 경우 나머지 팀원들이 확인 댓글을 단 뒤 merge를 수행



Project 2

■ gitHub 진행 방법

• Readme 작성

- Readme 파일은 repository 폴더 최상단에 Read.md로 존재함
- Repository 생성시 Readme.md를 생성하면서 초기화할 수도 있고, 임의로 생성할 수도 있음
- Markdown 문법 사용

- Readme에는 과제명, 팀장 및 팀원 이름과 gitHub id를 적을 것

Assignment2

팀장

2014722064 OOO(github id)

팀원

20147220XX OOO (github id)

20147220XX OOO (github id)

20147220XX OOO (github id)

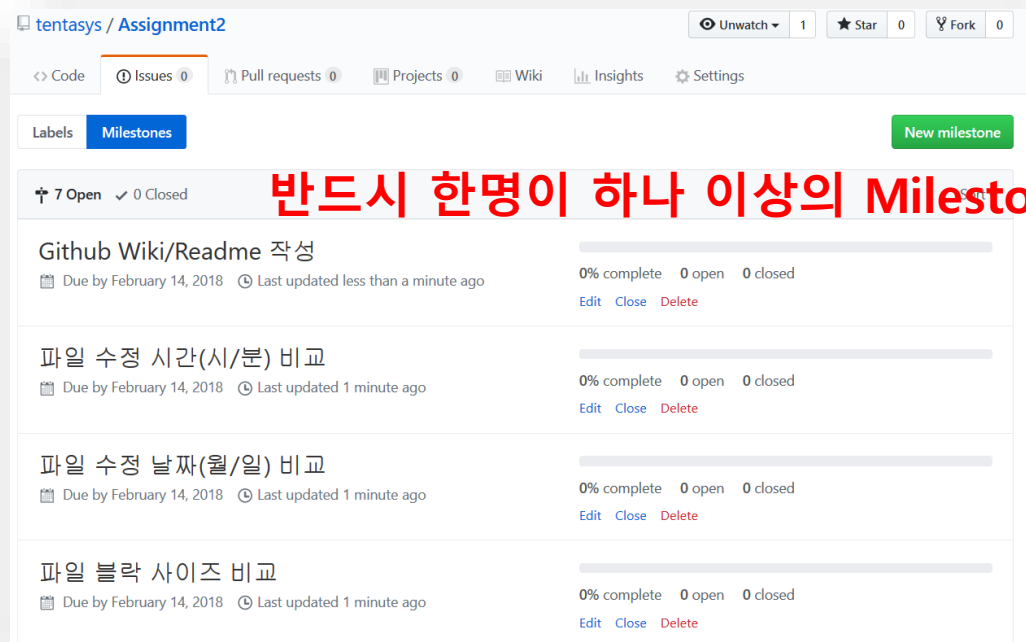


Project 2

■ gitHub 진행 방법

• Milestone 생성

- 각각의 모듈(기능 1~6)별 milestone과 gitHub wiki/readme 작성
milestone은 반드시 생성해야 함(총 milestone은 3개 이상)
- 그 이외에 필요하다고 생각하는 milestone을 작성해도 무방함





Project 2

■ gitHub 진행 방법

• Issue 작성

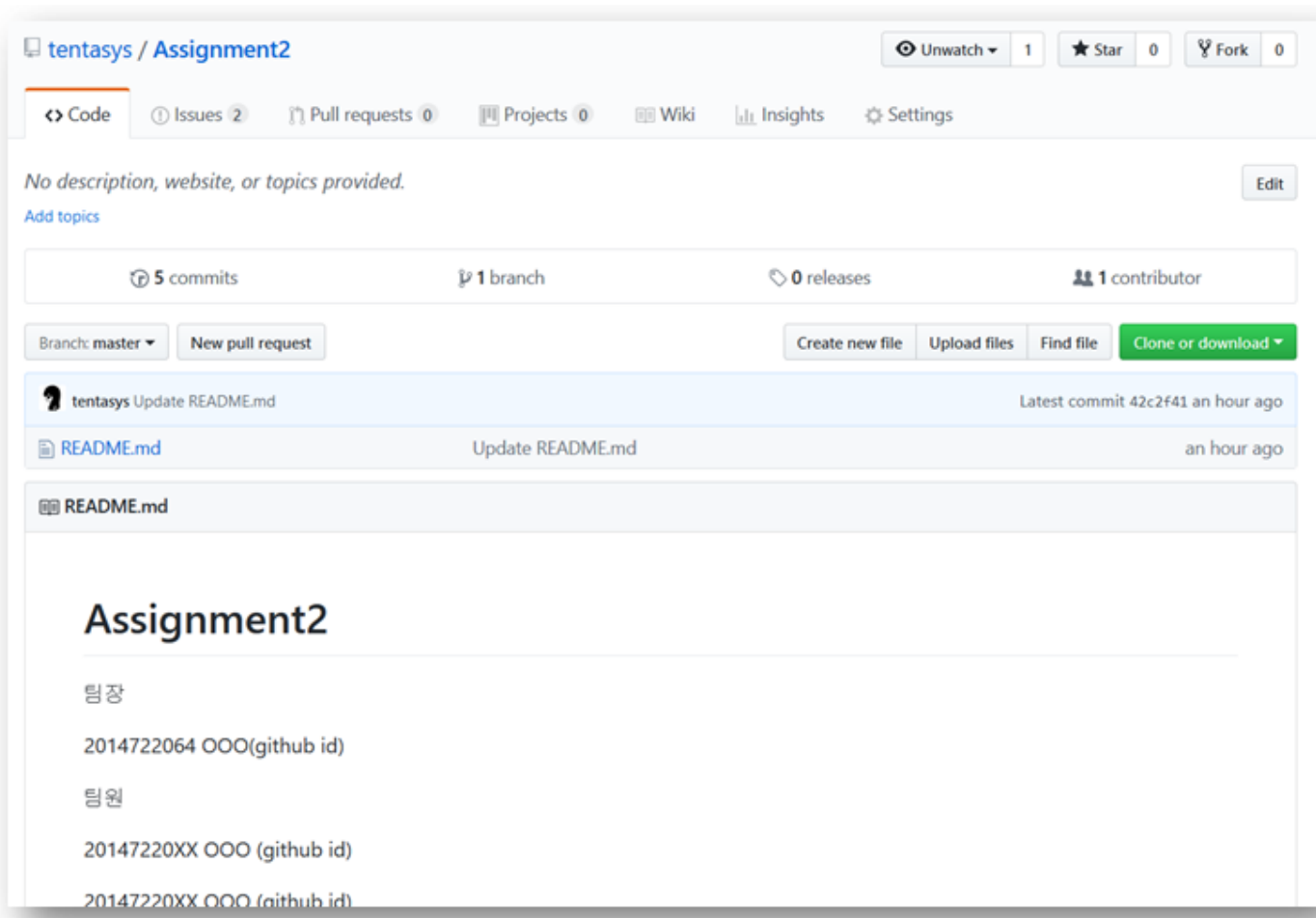
- Milestone을 달성하는데 필요하다고 생각하는 step을 issue로 생성하여 관리
- Milestone에 맞게 issue와 assignee가 잘 할당되어 있어야 함
- 과제 마감 시 closed상태가 아닌 issue가 있으면 감점
- Issue의 업데이트를 활발히 하여 현재 자신이 어느 작업을 하고 있는지 팀원이 알 수 있도록 해야 함
- 한 milestone에 최소한 하나 이상의 issue가 존재해야 함

<input type="checkbox"/> ⓘ 2 Open ✓ 1 Closed		Author ▾	Labels ▾	Projects ▾	Milestones ▾	Assignee ▾
<input type="checkbox"/>	🟢	파일 크기 비교 코드 테스트				👤
#3 opened 24 seconds ago by tentasys		🔗 파일 사이즈 비교				
<input type="checkbox"/>	🟢	파일 크기 비교 코드 작성				👤
#2 opened 38 seconds ago by tentasys		🔗 파일 사이즈 비교				
<input type="checkbox"/>	🔴	stat 함수 조사				👤
#1 by tentasys was closed 12 seconds ago		🔗 파일 사이즈 비교				



Project 2

- gitHub 진행 방법
 - gitHub 메인 캡처는 다음과 같은 화면을 캡처할 것





Project 2

■ gitHub Scenario

1. A, B, C, D 4명으로 구성된 팀의 경우, A를 팀장으로 선정
2. A는 gitHub repository를 생성하고 그 주소를 팀원들에게 공유
3. B, C, D는 해당 repository를 fork하거나 A가 B, C, D를 collaborator로 등록시킨 다음에 branch를 checkout
4. A 또한 master branch가 아닌 새로운 branch를 생성하고 checkout
5. A는 함수 1, 2, 3을 구현, B는 함수 4, 5, 6을 구현, C는 함수 7, 8을 구현, D는 readme 작성 및 wiki page를 작성하도록 역할을 분배
6. A는 함수 1, 2, 3 구현이라는 1개 이상의 milestone을 생성(다른 팀원들도 동일하게 milestone을 생성)
7. A는 함수1 구현에 기반 기술 조사, 코드 작성, 코드 리뷰라는 3개의 issue를 생성(다른 팀원 및 다른 milestone들도 동일)
8. 작업 진행 중, issue가 완료되면 issue를 close



Project 2

■ gitHub Scenario

9. A가 함수1 구현을 완료한 경우, git push를 진행하고 pull request를 보냄
10. B, C, D는 pull request를 확인하고 확인했다는 내용의 댓글을 작성
11. B, C, D의 확인 댓글이 달리면 pull request를 merge하고 pull request를 close함
12. 충돌이 발생했을 경우 충돌이 발생했다는 issue를 생성하거나 pull request에 충돌이 발생한 사람과 댓글을 주고받으면서 충돌을 해결



평가 기준

- 다음과 같은 요구사항을 충족해야 함

소스코드
파일 사이즈 비교가 제대로 되는가?
파일 블록 사이즈 비교가 제대로 되는가?
파일 수정 날짜 비교가 제대로 되는가?
파일 수정 시간 비교가 제대로 되는가?
결과화면을 제대로 출력하는가?
Runtime error가 일어나지 않고 제대로 동작하는가?
정해진 코드 양식에 따라 작성했는가?
코드 취합 과정에서 git을 사용하였는가?



평가 기준

- 다음과 같은 요구사항을 충족해야 함

gitHub
Milestone이 task별로 잘 나뉘어져 있는가?
Issue가 Milestone에 제대로 할당되어 있는가?
Issue에 Assignee가 할당이 되어 있는가?
Task를 팀원에게 균등하게 배분하였는가?
Issue와 comment를 이용하여 팀원 간의 소통이 원활이 이루어졌는가?
과제 마감 시 closed되지 않은 issue가 존재하는가?
Readme.md를 제대로 작성하였는가? (가독성보다는 내용 중시)
Wiki page를 제대로 작성하였는가? (가독성보다는 내용 중시)
팀원 전부가 commit에 참여하였는가?
마감 기한에 맞춰 commit이 이루어졌는가?



평가 기준

- 다음과 같은 요구사항을 충족해야 함

Report
gitHub repository 주소가 제대로 적혀 있는가?
결과 화면 캡처가 존재하는가?
gitHub repository 메인 화면 캡처가 존재하는가?
고찰을 잘 작성하였는가?

- 이 외에도 본 문서에 나타난 요구사항을 지키지 않았을 시 감점이 될 수 있음



Appendix

- stat() : 파일의 정보를 받아오는 함수

```
#include <sys/types.h>
#include <sys/stat.h>
#include <unistd.h>

int stat(const char* path, struct stat* buf);
```

- const char* path : 파일의 경로
- struct stat* buf : 파일의 정보가 들어가는 구조체 포인터

<pre>struct stat { dev_t st_dev; /* ID of device containing file */ ino_t st_ino; /* inode number */ mode_t st_mode; /* protection */ nlink_t st_nlink; /* number of hard links */ uid_t st_uid; /* user ID of owner */ gid_t st_gid; /* group ID of owner */ dev_t st_rdev; /* device ID (if special file)</pre>	<pre>off_t st_size; /* total size, in bytes */ blksize_t st_blksize; /* blocksize for file system I/O */ blkcnt_t st_blocks; /* number of 512B blocks allocated */ time_t st_atime; /* time of last access */ time_t st_mtime; /* time of last modification */ time_t st_ctime; /* time of last status change */ };</pre>
---	---



Appendix

- localtime(): time_t값을 활용할 수 있게 가공시켜주는 함수

```
#include <time.h>
```

```
struct tm *localtime(const time_t* timep);
```

- const time_t* timep : time_t형 변수를 가리키는 포인터
- struct tm* localtime : 시간 정보가 들어가는 구조체 포인터

```
struct tm {  
    int tm_sec; /* seconds */  
    int tm_min; /* minutes */  
    int tm_hour; /* hours */  
    int tm_mday; /* day of the month */  
    int tm_mon; /* month */  
    int tm_year; /* year */
```

```
    int tm_wday; /* day of the week */  
    int tm_yday; /* day in the year */  
    int tm_isdst; /* daylight saving time */  
};
```

- month의 경우 0부터 시작함(1월이 0으로 나타남)



Appendix

- stat과 localtime을 이용하여 파일의 정보를 가져오는 예제

```

1  #include <stdio.h>
2  #include <sys/types.h>
3  #include <sys/stat.h>
4  #include <unistd.h>
5  #include <time.h>
6
7  int main(void)
8  {
9      struct stat buf;
10     struct tm* time;
11
12     stat("text1", &buf);
13     time = localtime(&buf.st_mtime);
14     printf("size : %d \n", (int)buf.st_size);
15     printf("blocks : %d \n", (int)buf.st_blocks);
16     printf("month : %d \n", time->tm_mon+1);
17     printf("date : %d \n", time->tm_mday);
18     printf("hour : %d \n", time->tm_hour);
19     printf("min : %d \n", time->tm_min);
20
21     return 0;
22 }

```

```

root@linkaden:~/2014722064_OSS# ls -al
total 48
drwxr-xr-x  2 root root 4096 2월 12 16:48 .
drwx----- 29 root root 4096 3월 27 16:41 ..
-rwxr-xr-x  1 root root 9176 2월 12 16:15 main
-rw-r--r--  1 root root 2111 2월 12 16:15 main.c
-rwxr-xr-x  1 root root 8824 2월 12 16:48 test
-rw-r--r--  1 root root 476 2월 12 16:47 test.c
-rw-r--r--  1 root root 38 2월 12 16:13 text1
-rw-r--r--  1 root root 1977 2월 12 16:13 text2
root@linkaden:~/2014722064_OSS# ./test
size : 38
blocks : 8
month : 2
date : 12
hour : 16
min : 13

```

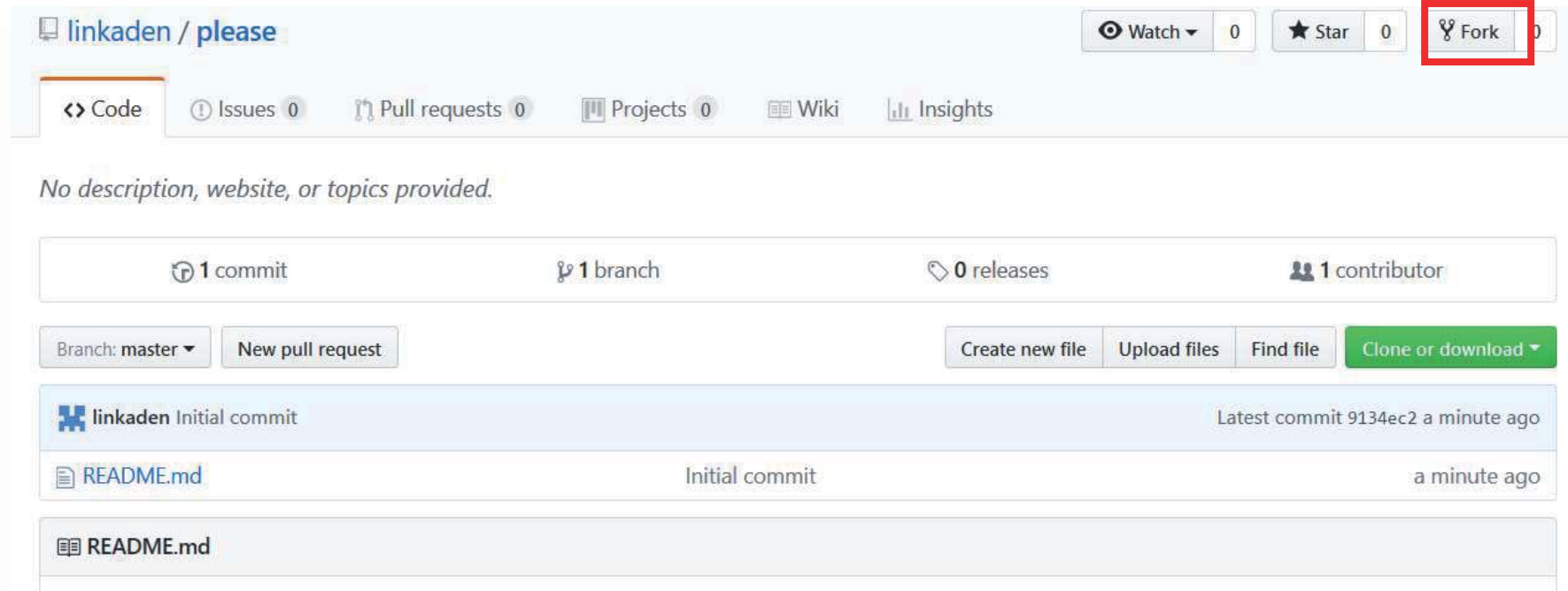


Appendix

- gitHub 예시 Repository
<https://gitHub.com/tentasys/Assignment2>

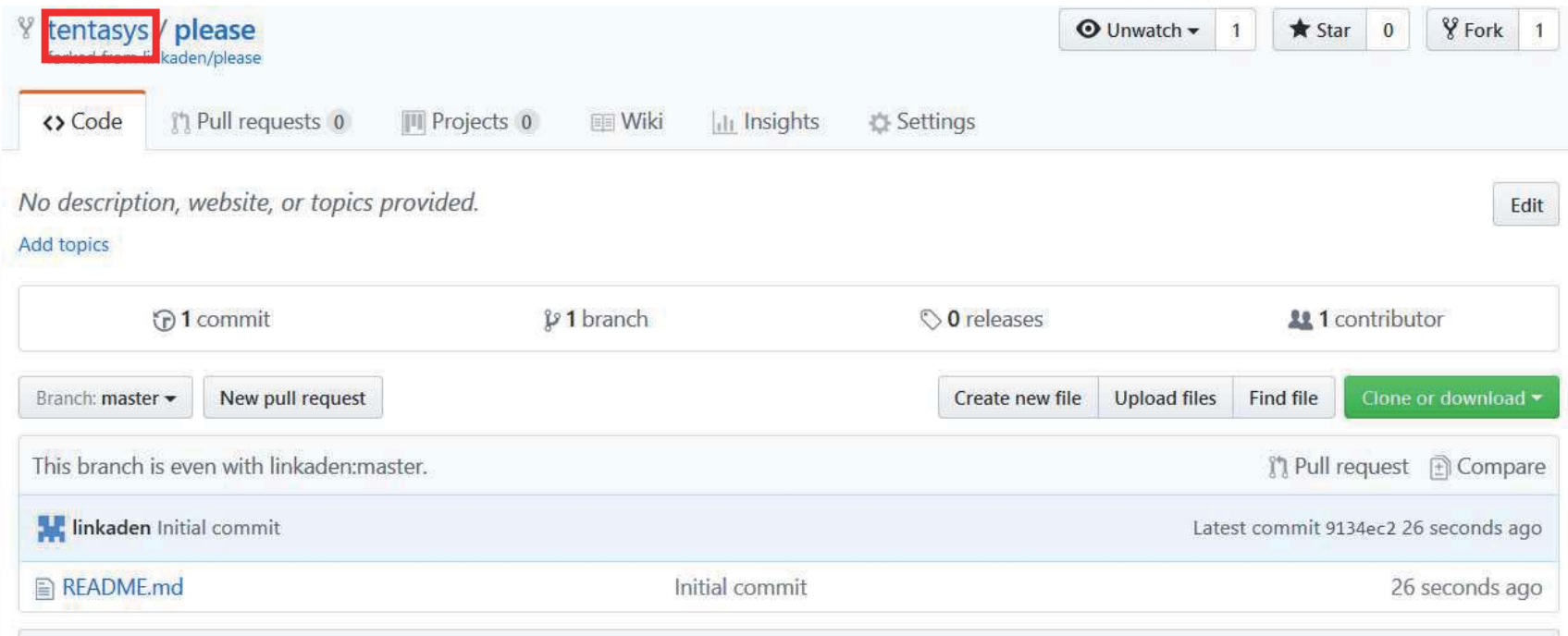
Fork

- 다른 사람의 프로젝트에서 Fork를 클릭



Fork

- 다른 사람의 프로젝트가 내 이름으로 Fork된 것을 확인



The screenshot shows a GitHub repository page for 'tentasys/please'. The repository name 'tentasys' is highlighted with a red box. Above the repository name, it says 'Forked from linkaden/please'. The page shows the repository has 1 commit, 1 branch, 0 releases, and 1 contributor. There are buttons for 'Code', 'Pull requests', 'Projects', 'Wiki', 'Insights', and 'Settings'. A message states 'No description, website, or topics provided.' with an 'Edit' button. Below this, there are buttons for 'Branch: master', 'New pull request', 'Create new file', 'Upload files', 'Find file', and 'Clone or download'. A commit history section shows 'linkaden Initial commit' with the latest commit hash '9134ec2' and timestamp '26 seconds ago'. A file named 'README.md' is also listed with the same commit information.

Pull request

- Fork한 repository를 clone을 이용해서 복사

```
heejung@heejung:~/test2/test$ cd ~
heejung@heejung:~$ git clone https://github.com/tentasys/please.git
'please'에 복제합니다...
remote: Counting objects: 13, done.
remote: Total 13 (delta 0), reused 0 (delta 0), pack-reused 13
오브젝트 묶음 푸는 중: 100% (13/13), 완료.
연결을 확인하는 중입니다... 완료.
```

- clone한 저장소로 이동한 뒤 git checkout -b test : test라는 branch를 만들고 그 branch로 바로 이동
- git branch test, git checkout test를 연달아 수행하는 것과 git checkout -b test는 같음
- 파일을 편집하고 commit 진행

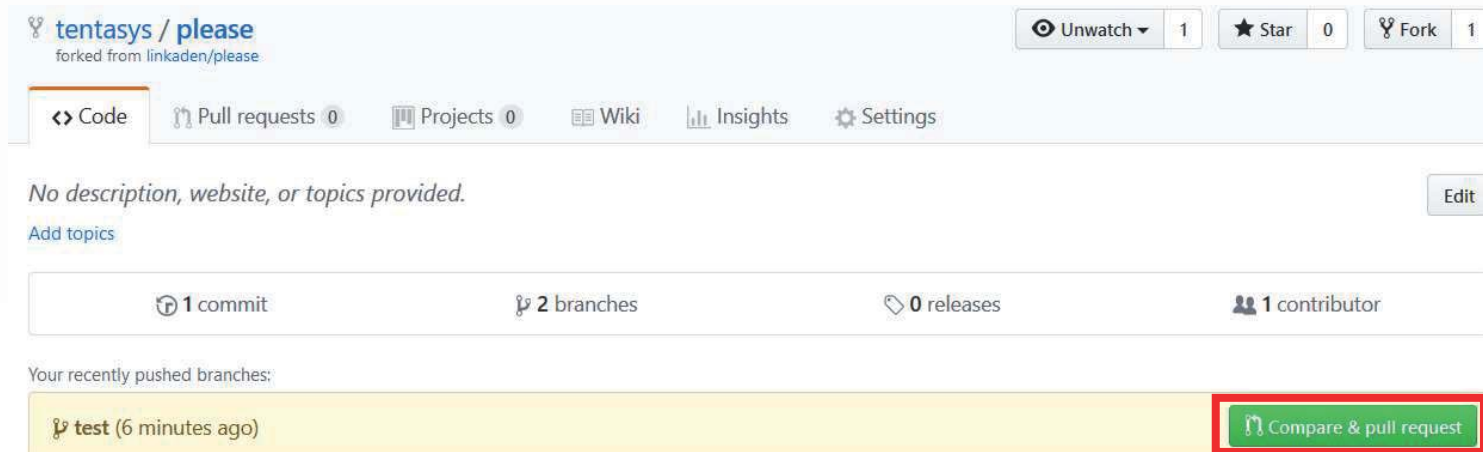
```
heejung@heejung:~$ cd please
heejung@heejung:~/please$ git checkout -b test
새로 만든 'test' 브랜치로 전환합니다
heejung@heejung:~/please$ vim test
heejung@heejung:~/please$ git add --all
heejung@heejung:~/please$ git commit -m "pull request"
[test 174393d] pull request
1 file changed, 1 insertion(+), 1 deletion(-)
```


Pull request

- git push origin **test**로 새로 만든 branch에 push 진행

```
heejung@heejung:~/please$ git push origin test
Username for 'https://github.com': tentasys
Password for 'https://tentasys@github.com':
오브젝트 개수 세는 중: 6, 완료.
오브젝트 압축하는 중: 100% (4/4), 완료.
오브젝트 쓰는 중: 100% (6/6), 519 bytes | 0 bytes/s, 완료.
Total 6 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), done.
To https://github.com/tentasys/please.git
 * [new branch]      test -> test
```

- GitHub의 해당 repository에서 pull request가 활성화된 것을 확인하고 클릭



Pull request

- Pull request에 대한 설명을 작성하고 Create pull request를 클릭

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 fork: linkaden/please ▾ base: master ▾  head fork: tentasys/please ▾ compare: test ▾

✓ **Able to merge.** These branches can be automatically merged.



pull request

Write

Preview

AA ▾ B i

“ < > ↻

≡ ≡ ≡

↶ @

코드를 수정했습니다. pull request를 보냅니다.

Attach files by dragging & dropping, [selecting them](#), or pasting from the clipboard.

☒ Allow edits from maintainers. [Learn more](#)

Create pull request


Pull request

- pull request를 보낸 후의 화면

pull request #1

 Open tentasys wants to merge 1 commit into linkaden:master from tentasys:test

 Conversation 0

 Commits 1

 Files changed 1



tentasys commented just now



코드를 수정했습니다. pull request를 보냅니다.



pull request

3ab4732

Add more commits by pushing to the **test** branch on tentasys/please.

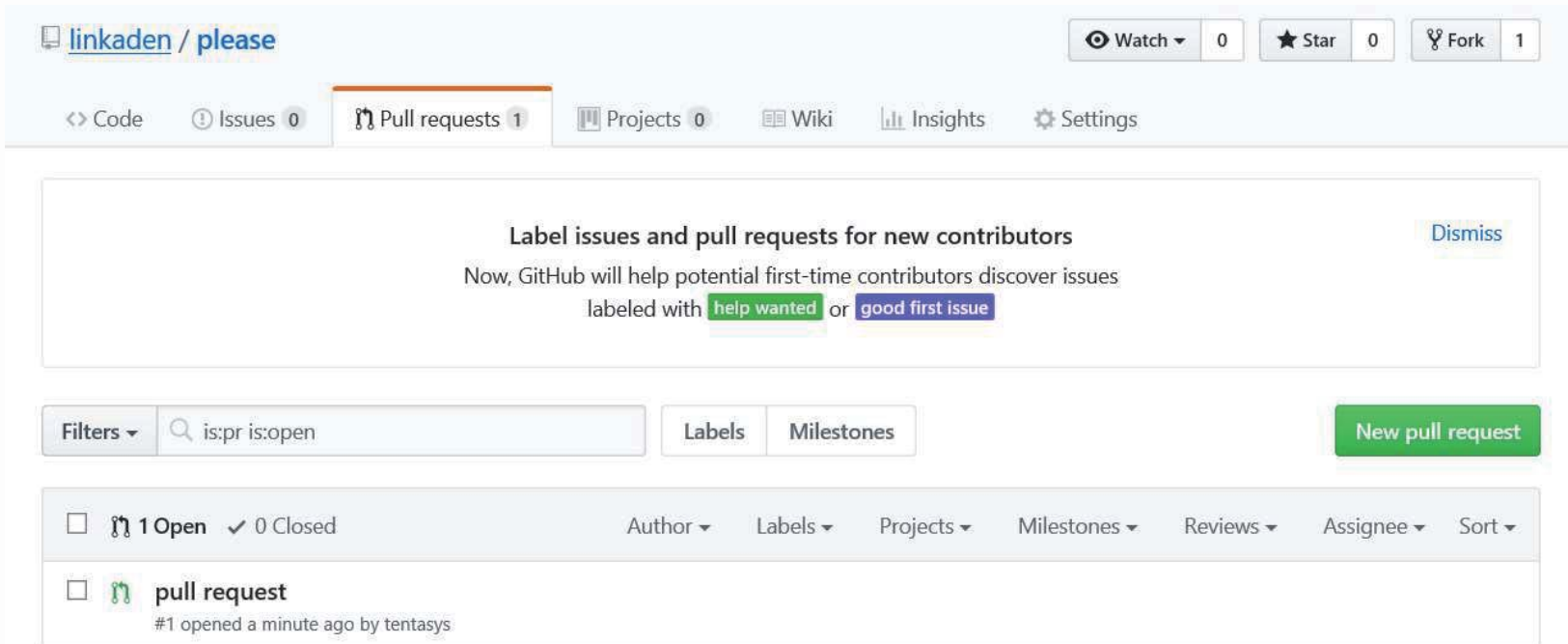


This branch has no conflicts with the base branch

Only those with [write access](#) to this repository can merge pull requests.

Pull request

- Repository의 owner 시점에서 Pull request를 누르면 Pull request를 확인할 수 있음

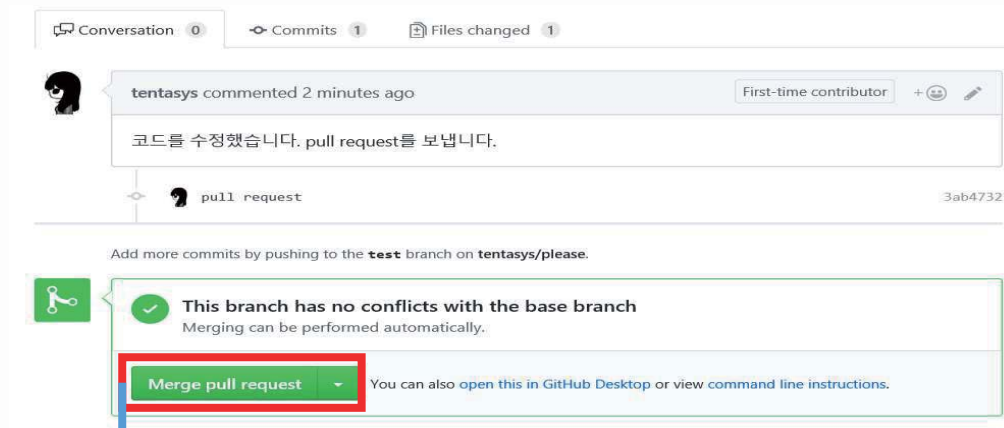


The screenshot shows the GitHub interface for the repository 'linkaden / please'. At the top, there are buttons for 'Watch' (0), 'Star' (0), and 'Fork' (1). Below these are tabs for 'Code', 'Issues' (0), 'Pull requests' (1), 'Projects' (0), 'Wiki', 'Insights', and 'Settings'. A notification banner states: 'Label issues and pull requests for new contributors. Now, GitHub will help potential first-time contributors discover issues labeled with **help wanted** or **good first issue**.' Below the banner, there is a 'Filters' section with a search bar containing 'is:pr is:open' and buttons for 'Labels' and 'Milestones'. A green button labeled 'New pull request' is on the right. The main content area shows a list of pull requests with a header row: '1 Open' and '0 Closed'. Below this, a single pull request is listed: 'pull request #1 opened a minute ago by tentasys'.

💡 ProTip! `no:milestone` will show everything without a milestone.

Pull request


- 해당 Pull request를 클릭하고 Merge pull request를 클릭하면 수정한 내용이 원래 프로젝트에 합쳐진다.



Pull request

- owner가 merge를 완료하면 Delete branch가 나타나고 이를 클릭하면 Fork해서 생성된 branch가 삭제됨

pull request #1

 **Merged** linkaden merged 1 commit into linkaden:master from tentasys:test 18 seconds ago

 Conversation 0  Commits 1  Files changed 1



tentasys commented 4 minutes ago

코드를 수정했습니다. pull request를 보냅니다.



pull request

3ab4732



linkaden merged commit c2e5483 into linkaden:master 18 seconds ago

[Revert](#)



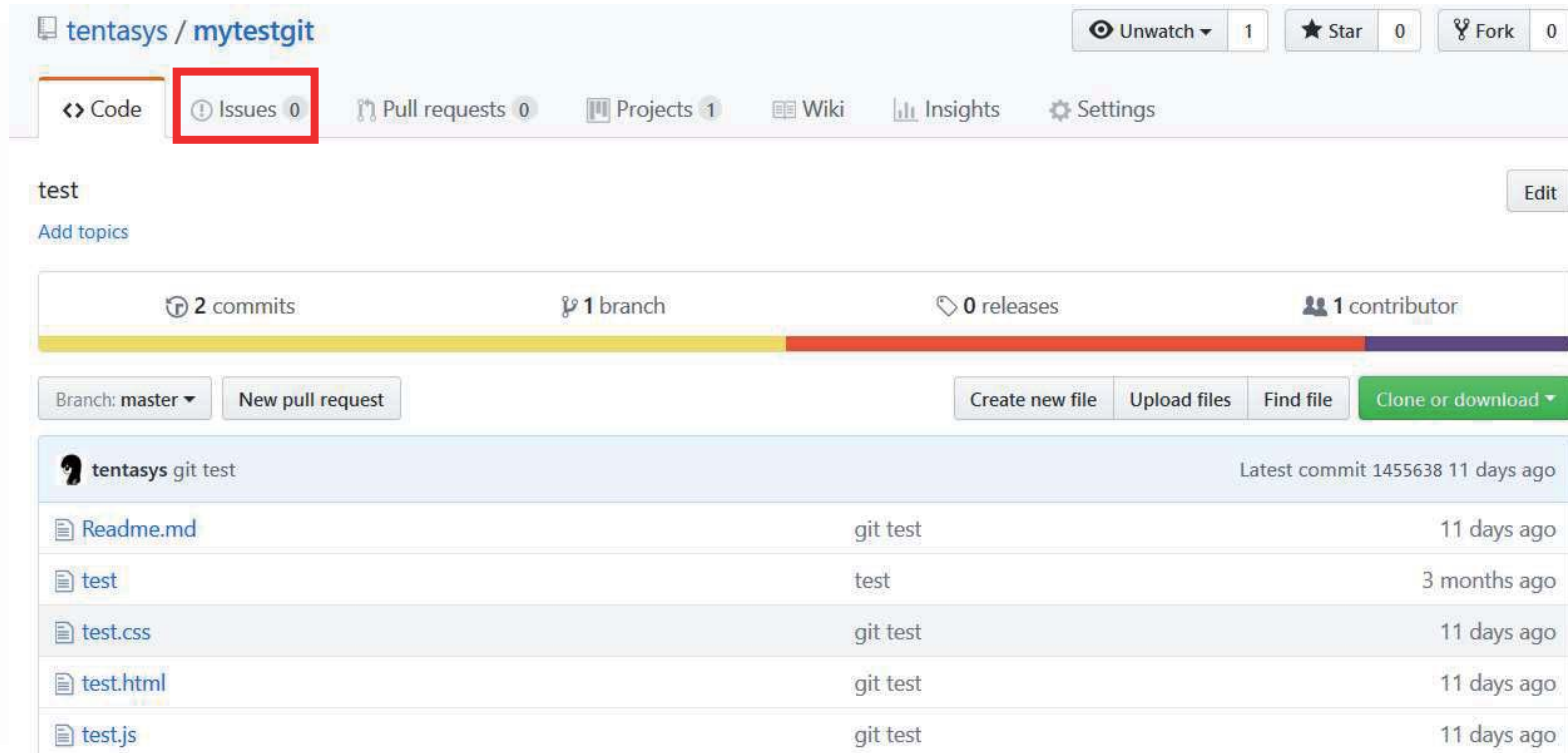
Pull request successfully merged and closed

You're all set—the tentasys:test branch can be safely deleted.

[Delete branch](#)

Issue

- Repository 상단의 Issues 클릭



tentasys / mytestgit

Unwatch 1 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 1 Wiki Insights Settings

test Edit

Add topics

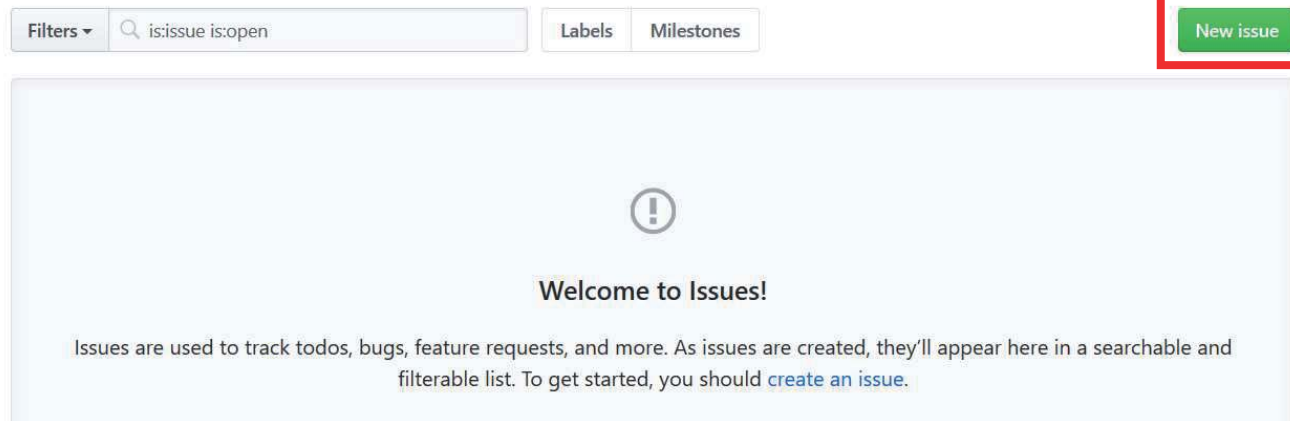
2 commits 1 branch 0 releases 1 contributor

Branch: master New pull request Create new file Upload files Find file Clone or download

tentasys git test	Latest commit 1455638 11 days ago
Readme.md	git test 11 days ago
test	test 3 months ago
test.css	git test 11 days ago
test.html	git test 11 days ago
test.js	git test 11 days ago

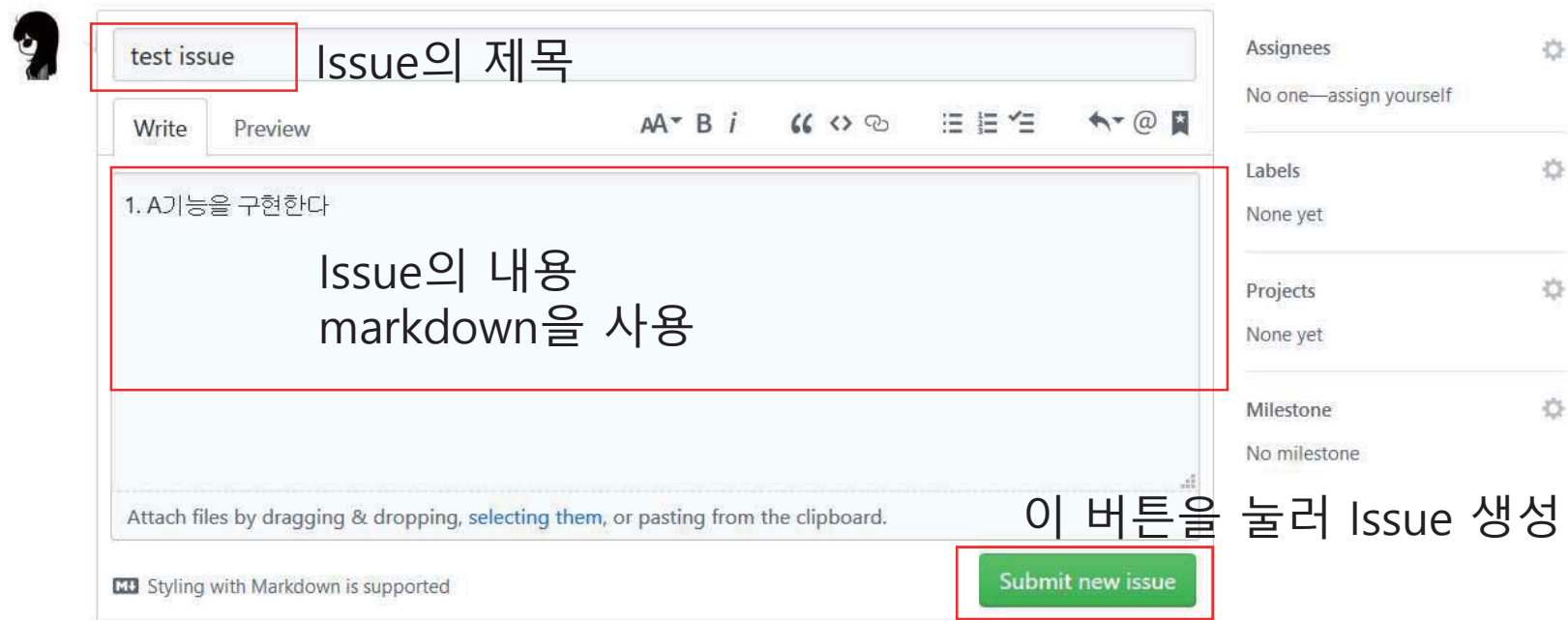
Issue

- New Issue 클릭



Issue

- New Issue 클릭하면 다음과 같은 창이 등장



test issue Issue의 제목

Write Preview

1. A기능을 구현한다

Issue의 내용
markdown을 사용

Attach files by dragging & dropping, selecting them, or pasting from the clipboard.

Styling with Markdown is supported

Submit new issue

Assignees
No one—assign yourself

Labels
None yet

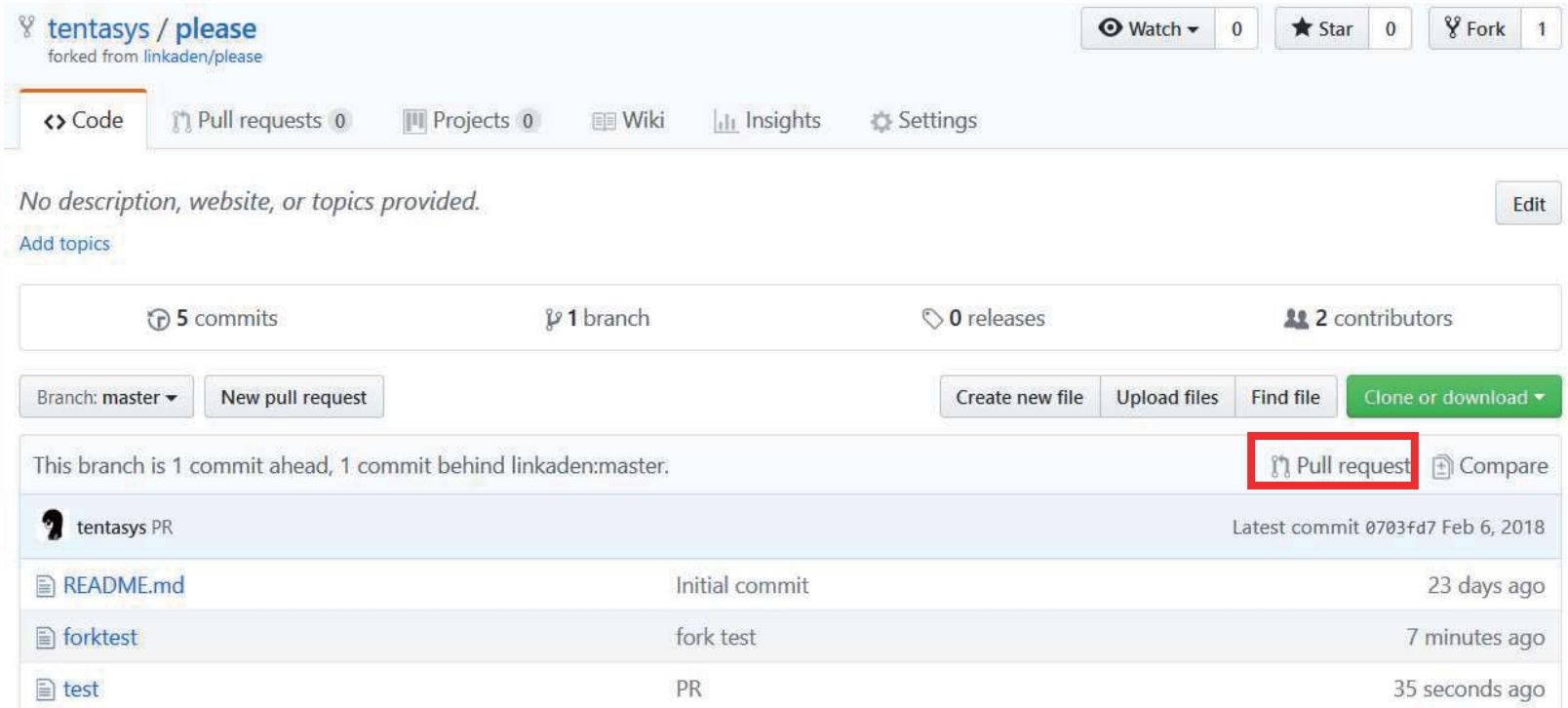
Projects
None yet

Milestone
No milestone

이 버튼을 눌러 Issue 생성

실습 과제 - Fork & PR

- Fork한 repository에서 Pull request 보내는 방법(B)
- Fork한 repository로 진입
- Pull request 클릭



The screenshot shows the GitHub interface for a repository named 'tentasys / please', which is a fork of 'linkaden/please'. The repository has 5 commits, 1 branch, 0 releases, and 2 contributors. The 'master' branch is selected. A red box highlights the 'Pull request' button in the top right corner of the repository view. Below this, a table lists the commit history:

File	Commit Message	Time
README.md	Initial commit	23 days ago
forktest	fork test	7 minutes ago
test	PR	35 seconds ago




실습 과제 - Fork & PR

- Create pull request 클릭

Comparing changes

Choose two branches to see what's changed or to start a new pull request. If you need to, you can also [compare across forks](#).



base fork: linkaden/please ▾


base: master ▾

←


head fork: tentasys/please ▾


compare: master ▾


✓ **Able to merge.** These branches can be automatically merged.


 **Create pull request**


Discuss and review the changes in this comparison with others.






 1 commit

 1 file changed

 0 commit comments

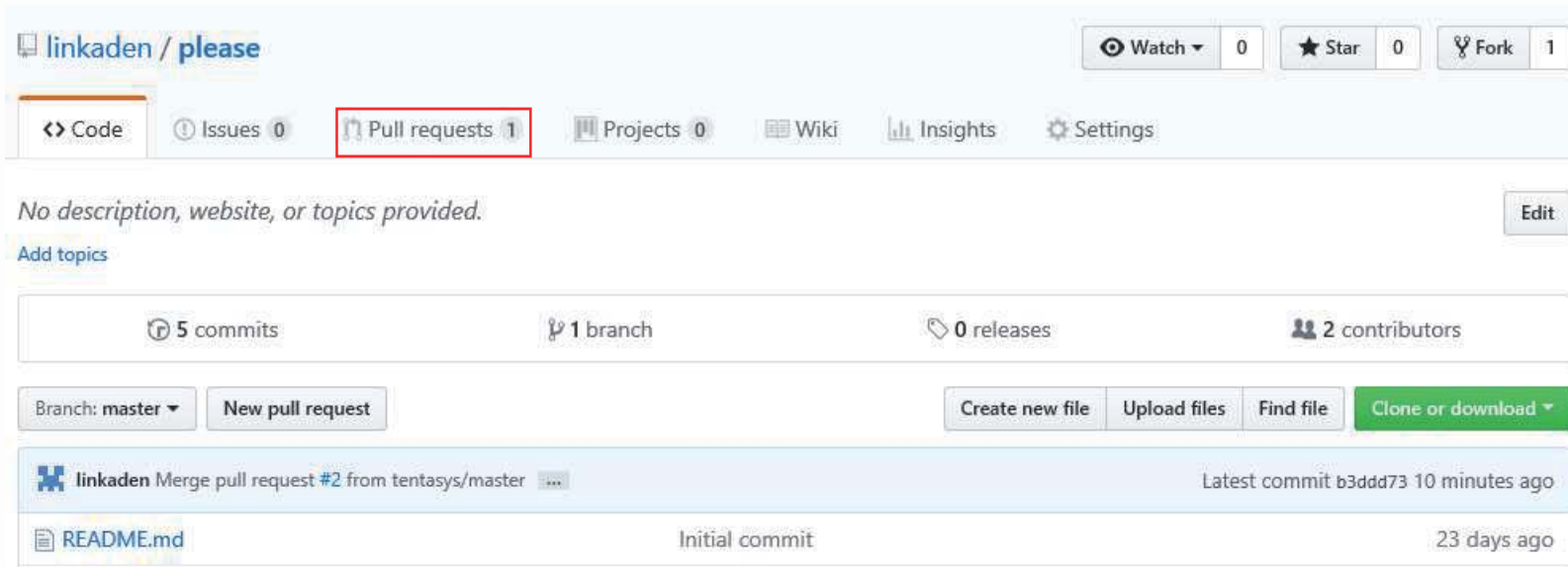
 1 contributor

 Commits on Feb 06, 2018

  tentasys PR 0703fd7

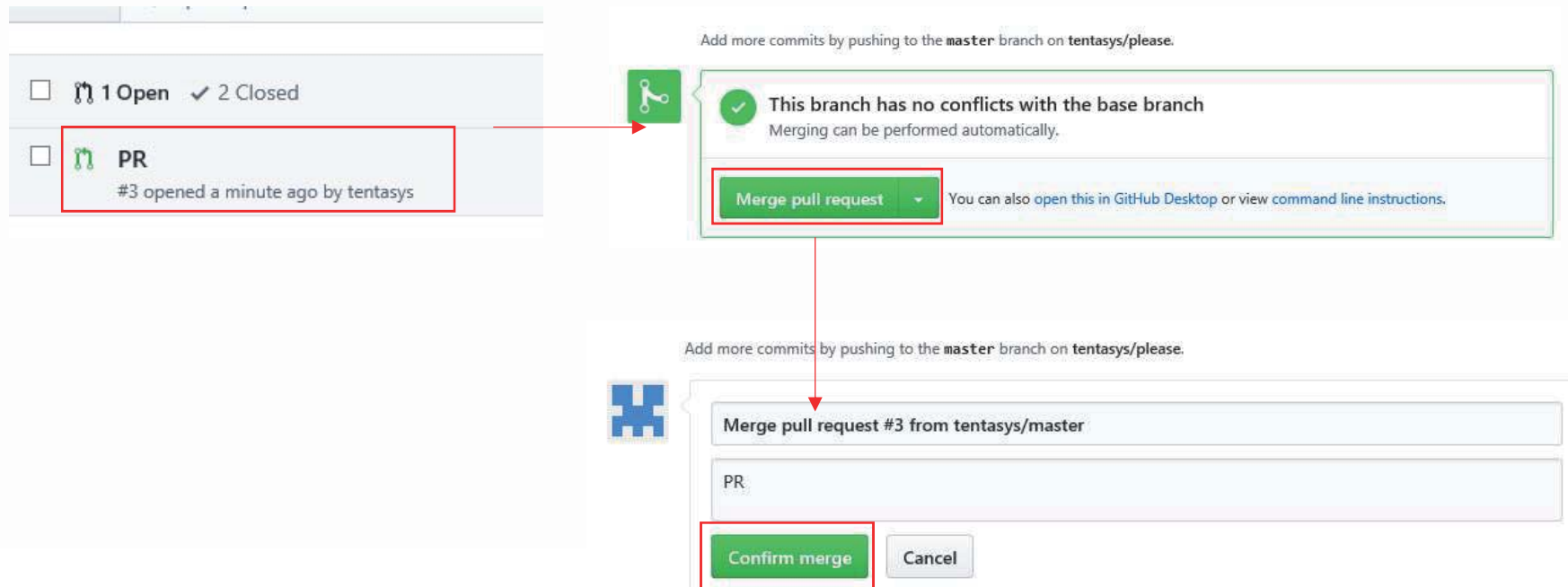
실습 과제 - Fork & PR

- Pull request 수락하는 방법(A)
- GitHub repository에서 Pull request에 숫자가 있는 것을 확인하고 클릭



실습 과제 - Fork & PR

- Pull request 창에서 활성화된 Pull request 클릭
- Merge pull request를 누르고 Confirm merge를 클릭하여 merge 수행



The screenshot illustrates the process of merging a pull request on GitHub. It is divided into three main sections connected by red arrows:

- Pull Request List:** A list of pull requests is shown. The first item is "1 Open" and the second is "2 Closed". A red box highlights a pull request titled "PR" with the description "#3 opened a minute ago by tentasys".
- Pull Request Details:** The details of the selected pull request are shown. A green checkmark indicates "This branch has no conflicts with the base branch". Below this, a green button labeled "Merge pull request" is highlighted with a red box.
- Merge Confirmation:** A modal dialog box appears with the title "Merge pull request #3 from tentasys/master". It contains a text input field with "PR" and two buttons: "Confirm merge" (highlighted with a red box) and "Cancel".