



\*1

깃과 깃허브 클래스룸

UT-NodeJS / 03.10.2023

[ut-nodejs.github.io](https://ut-nodejs.github.io)



# Contents / 내용

설치와 사용하는 방법



## 01. git / 깃

로컬에서 관리되는 버전 관리 시스템 (VCS : Version Control System)

## 02. GitHub / 깃허브

클라우드 방식으로 관리되는 버전 관리 시스템(VCS)

## 03. Classroom / 깃허브 클래스룸

과제를 수락하고 로컬 컴퓨터에서 코드를 작성하고 작업을 다시 GitHub Classroom으로 푸시하세요.

# 01 git

로컬에서 관리되는 버전 관리 시스템  
(VCS : Version Control System)



# 깃이 뭐가?

Git은 버전 관리 '프로그램'



# git



## 0. 시작하기

- 오픈 소스 버전 관리 시스템 (VCS: Version Control System)
- 로컬에서 버전 관리
- 소프트웨어 개발 및 소스 코드 관리에 사용

git은 본인의 코드와 그 수정내역을 기록하고 관리하도록 돕는 버전 관리 프로그램이며, 로컬에서 프로젝트의 기록을 스스로 관리할 수 있도록 해줍니다. git을 통해 브랜치를 생성하고 이전 브랜치로 복구, 삭제, 병합이 가능합니다. 하지만 로컬 저장소를 사용하기 때문에 다른 개발자와 실시간으로 작업을 공유할 수 없습니다.

**버전 관리**란 시간에 따라 파일의 변경사항을 추적하고 기록하는 것입니다. 버전 관리 시스템은 이전 버전으로 복구하거나 조회할 수 있는 기능을 제공합니다. 버전 관리는 프로젝트의 수정이 있을 때마다 snapshot을 찍습니다. 따라서 필요한 것을 복구하거나 비교할 때 다양한 버전들을 확인할 수 있습니다.

<https://cococon1787.tistory.com/723>  
<https://escapefromcoding.tistory.com/281>



## 버전 관리는 왜 하는가?

최종 굴레에 빠져본 경험이 있는가?



git

"조별과제\_영미최종.docx"

"조별과제\_영미최종\_철수최종.docx"

"조별과제\_영미최종\_철수최종\_갑수최종.docx"

"조별과제\_영미최종\_철수최종\_갑수최종\_최종제출.docx"

"조별과제\_영미최종\_철수최종\_갑수최종\_최종제출\_진짜최종.docx"

.....

"조별과제\_영미최종\_철수최종\_갑수최종\_최종제출\_진짜최종\_영미수정\_교수님수정.docx"

"조별과제\_영미최종\_철수최종\_갑수최종\_최종제출\_진짜최종\_영미수정\_교수님수정\_제발최종.....docx"

흔히 사용하는 버전 관리 방법이다. **과거 파일을 백업 떠두는 방식**이다. 혼자 작업하거나 작은 파일을 단기간에 걸쳐 수정한다면 나쁘지 않은 방법이다.

하지만 조별과제는 **‘너도나도 여기저기서’** 파일을 수정하고 추가한다.

<https://brunch.co.kr/@anonymdevoo/3>



# 버전 관리는 왜 하는가?

최종 굴레에 빠져본 경험이 있는가?



- 1. 여러 개의 파일 버전을 일관되게 관리할 수가 없다.**
- 2. 누가/무엇을/어떻게 변경했는지 기록하고 내용을 공유하기 어렵다.**
- 3. 의도치 않게 서로의 변경 내역을 덮어쓰거나 지울 수 있다.**
- 4. 수정한 내용 이전 상태로 복구하기 번거롭다.**
- 5. 취합은 조장이 하게 될 것이다.**

git을 쓰면 위 문제점에 대해서 자유로워질 수 있다. 특히 작업물의 크기가 커지면 커질수록 진가가 드러난다.



## 중요한 깃 명령

여러분 기본 git 명령에 익숙해야 합니다.



# git



**git init**

새로운 저장소 만들기

**git clone http://URL** 저장소 받아오기

**git add \*** 변경된 파일은 (인덱스에) 추가

**git commit -m "이번 확정본에 대한 설명"** 저장소에 저장

**git remote add origin <원격 서버 주소>** 원격 서버의 주소 차가

**git push -u origin master** 원격 저장소로 올리기

**git pull** 로컬 저장소를 원격 저장소에 맞춰 갱신하기

The background is white and features various decorative geometric shapes in black and yellow. These include triangles, circles, and crosses, some of which are hollow and others solid. They are scattered around the central black box, with some appearing as if they are floating or falling. The shapes are primarily located in the corners and along the edges of the frame.

# 02

# GitHub

클라우드 방식으로 관리되는  
버전 관리 시스템(VCS)





## 깃허브이 뭐가?

Github는 버전 관리, 소스 코드 공유, 분산 버전 제어 등등이 가능한 원격 저장소



GitHub



### 깃허브 대해서

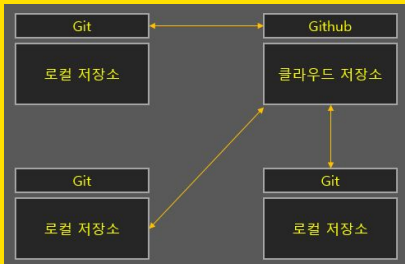
- Git Repository를 위한 웹 기반 호스팅 서비스
- 클라우드 서버를 사용해서 로컬에서 버전 관리한 소스코드를 업로드하여 공유 가능
- 분산 버전 제어, 액세스 제어, 소스 코드 관리, 버그 추적, 기능 요청 및 작업 관리를 제공

github는 git 저장소를 관리하는 클라우드 기반 호스팅 서비스입니다. git 저장소 호스팅 서비스는 클라우드 기반으로 다른 사람과 소스코드 공유가 가능하며 git의 기본적인 기능을 확장하여 제공합니다. 또한 클라우드 서버에 소스를 올리기 때문에 한 프로젝트에 여러 명의 사람이 참여하여 버전 제어 및 공동 작업이 가능합니다.



## 깃허브이 뭐가?

Github는 버전 관리, 소스 코드 공유, 분산 버전 제어 등등이 가능한 원격 저장소



## 깃허브 대해서

간단히 Git은 로컬에서 버전 관리 시스템을 운영하는 방식이고 Github는 저장소를 깃허브에서 제공해주는 클라우드 서버를 이용한다는 것의 차이입니다. 따라서 다른 사람들과 협업할 경우, 오픈소스를 공유하고 다른 사람들의 의견을 듣고 싶은 경우 등은 Github를 써서 편리하게 기능을 사용할 수 있습니다. 만약 혼자 작업하거나 폐쇄적인 범위 내에서의 협업이라면 Git만 사용해도 무방합니다.

따라서 일단 Git으로 로컬 저장소에 작업한 내용을 저장한 뒤 해당 내용을 Github에 업로드하는 형식으로 사용하게 됩니다. 또한 Github에 있는 콘텐츠를 내려받을 수도 있습니다. 이 세 과정을 의미하는 단어가 커밋(Commit), 푸쉬(push), 풀(Pull)입니다.

- 커밋(Commit) : Git(로컬 저장소)에 파일을 추가하거나 변경 내용을 저장하는 작업
- 푸쉬(Push) : Github(또는 원격 저장소)에 파일을 추가하거나 변경 내용을 저장하는 작업
- 풀(Pull) : Github(또는 원격 저장소)에서 파일을 다운로드하는 작업



## 깃은 어떻게 하는가가?

git을 시작하기 위한 간편 안내서. 어렵지 않아요 ;)



### git - 간편 안내서

git을 시작하기 위한 간편 안내서. 어렵지 않아요 ;)



Roger Dudler가 만들었어요.

(@tfnico, @fhd와 Namics의 도움을 받았지요.)

번역은 Juntai Park과 Ardie Hwang이 담당했습니다.

문제 보고는 여기(github)로 해주세요!

간단하게

<https://up1.github.io/git-guide/index.ko.html>

### 누구나 쉽게 이해할 수 있는 Git 입문

버전 관리를 완벽하게 이용해보자!

누구나 쉽게 이해할 수 있는 Git 에 입문하신 것을 환영합니다. 지금부터 Git을 사용한 버전 관리 기능을 함께 공부해 보자구요!!! 총 3가지의 코스가 준비되어 있습니다. Git 초보자 분들은 '입문편'부터 시작해주세요. Git을 사용한 적이 있으신 분은 '발전편'을 추천 합니다. '어? 뭐였지...?' 싶을 때는 '찾아보기'를 확인하세요.



더 깊게

<https://backlog.com/git-tutorial/kr/>



# 03

## Classroom

과제를 수락하고 로컬 컴퓨터에서 코드를 작성하고  
작업을 다시 GitHub Classroom으로 푸시하세요.



# 깃허브 클래스룸이 뭔가요?

교사는 클래스룸을 사용하여 단일 과정에 대한 학생, 조교 및 과제를 구성하고 관리할 수 있습니다.



- Git Repository
- 클라우드
- 분산 버전

github는 git  
서비스는 클라우  
확장하여 제공  
사람이 참여하

## CSE 142 Intro to Programming

ClassroomExampleClasses

Assignments 5

Students 0

TAs and Admins 1

Settings

### Assignments

New assignment

Assignment 1: Calculator  
Individual assignment

Invite link

Assignment 2: Temperature Conversion Tool  
Individual assignment

Invite link

Assignment 3: Rock, Paper, Scissors  
Individual assignment

Invite link

Assignment 4: Tic Tac Toe  
Individual assignment

Invite link

Assignment 5: To Do List  
Individual assignment

Invite link

유 가능  
관리를 제공

호스팅  
인 기능을  
여러 명의





# Classroom / 깃허브 클래스룸

온라인 클래스룸 아니고 학생들의 저장소 관리 시스템이다



## 01. Teacher View / 교사 보기

수업, 과제 목록, 과제 참여 코드, 자동 채점, 수업 명단

## 02. Student View / 학생 보기

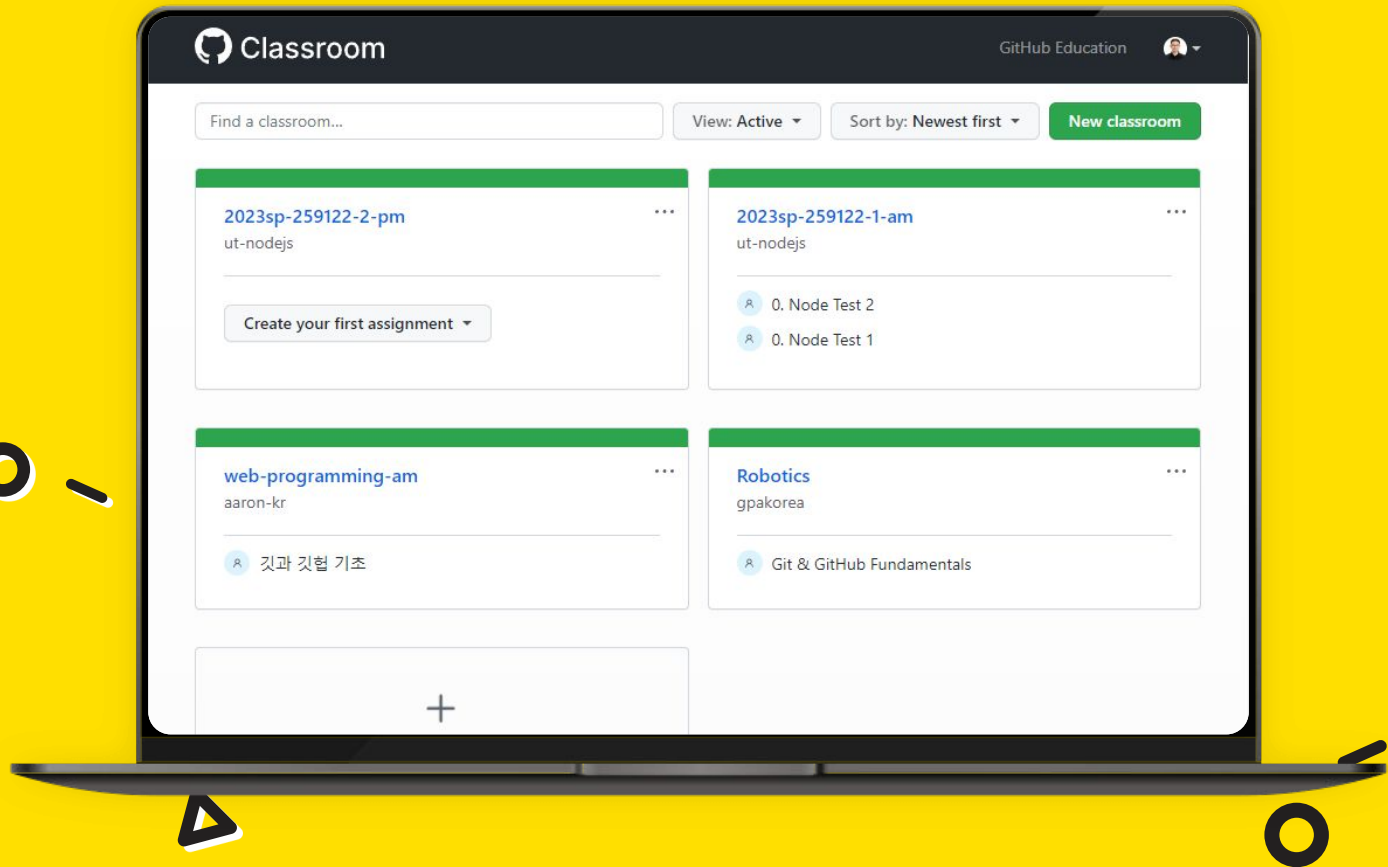
홈페이지, 과제 참여 링크, 개별 저장소, GitHub 조직

## 03. Assignments / 과제 제출 방법

과제 참여 링크, VS Code 링크, 코딩, commit & push, 테스트 통과 확인

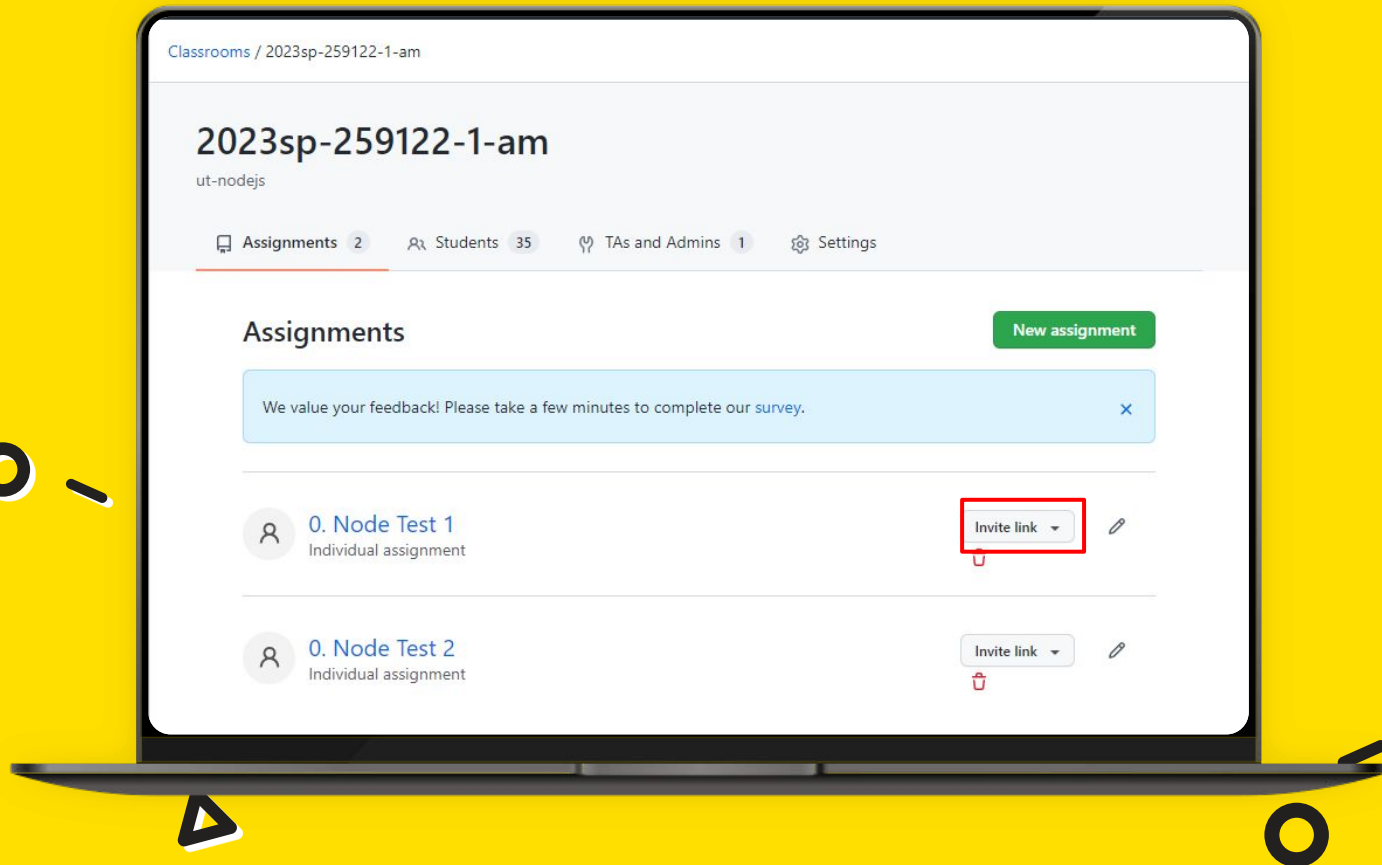
## 교사 보기

1. 수업,
2. 과제 목록,
3. 과제 참여 코드,
4. 자동 채점,
5. 수업 명단



## 교사 보기

1. 수업,
2. 과제 목록,
3. 과제 참여 코드,
4. 자동 채점,
5. 수업 명단





# 교사 보기

1. 수업,
2. 과제 목록,
3. 과제 참여 코드,
4. 자동 채점,
5. 수업 명단



Classrooms / 2023sp-259122-1-am / 0. Node Test 1

## 0. Node Test 1

Individual assignment • Active VS Code <https://classroom.github> Edit Download

Search by GitHub username or student identifier

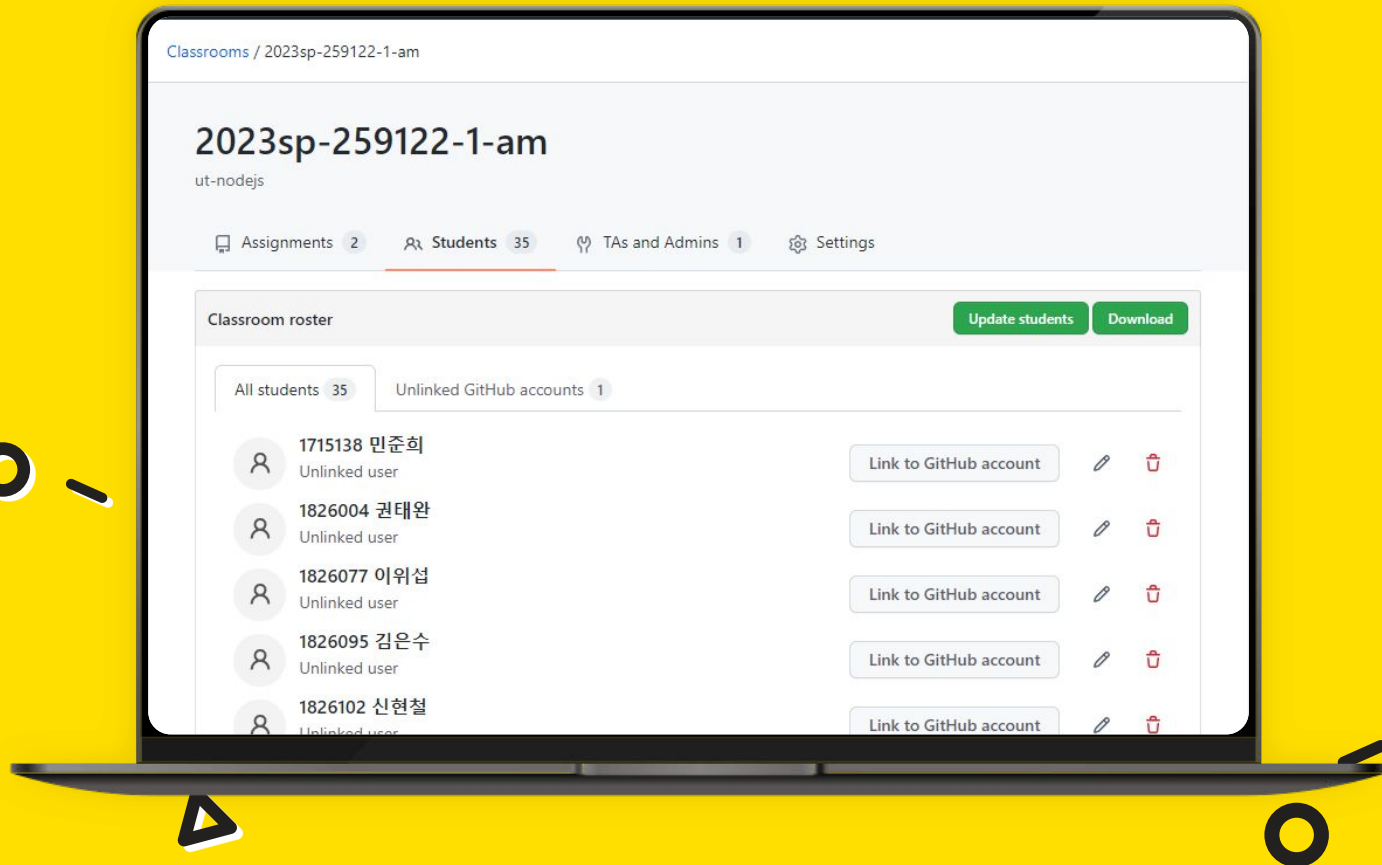
**Classroom roster** Unlinked accounts Accepted Submitted Passing Sort

 <b>AaronKR</b> @aaronkr-trainer	<input checked="" type="checkbox"/> Latest commit passed	18 commits	Autograding
 <b>jekkilekki</b> <a href="#">Link to student identifier</a>	<input type="checkbox"/> Latest commit failed	0 commits	Autograding Repository

**Statistics:**  
Rostered students: 35  
Added students: 1  
Accepted students: 2  
Assignment submissions: 0  
Passing students: 1/2

## 교사 보기

1. 수업,
2. 과제 목록,
3. 과제 참여 코드,
4. 자동 채점,
5. 수업 명단



## 학생 보기

1. [홈페이지](#),
2. 과제 참여 링크,
3. 개별 저장소,
4. GitHub 조직

New Tab x +

← → ↻ 🔒 http://

🏠 | 소개 | 일정 | 슬라이드 | **[과제]** | 프로젝트 | 시험 | 성적 | 설문 ↗

# Web Programming Application 2023

한국교통대학교, 충주 | KNUT (Korea National University of Transportation)

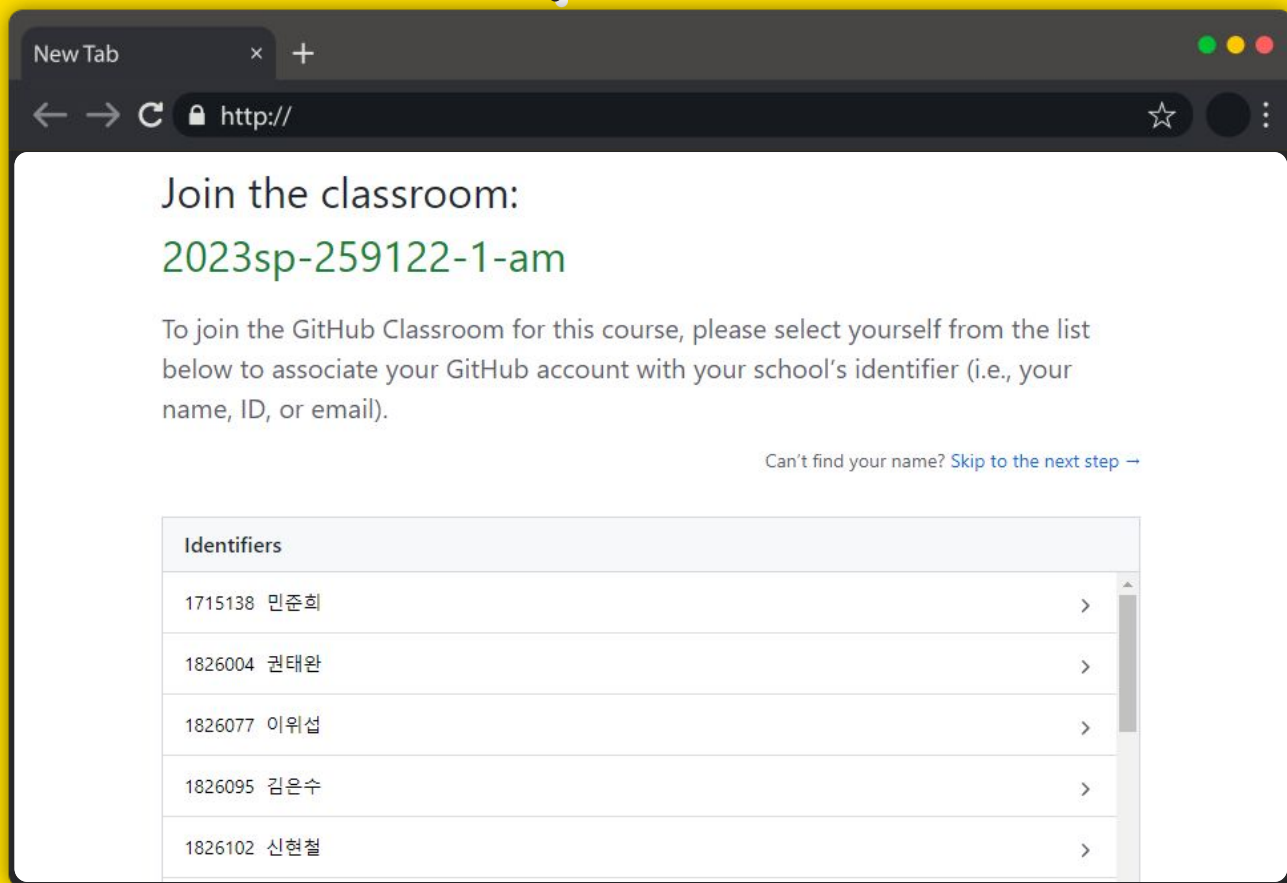
## Practice / 과제

Practice Labs will be assigned and collected in **GitHub** Classroom and accessible there. Follow the instructions below to understand how to access and submit labs.

Week	Date	과제	am	pm
1	3월3일	-	-	-
2	3월10일	0. Starting Node / 노드 시작	<b>오전</b>	오후

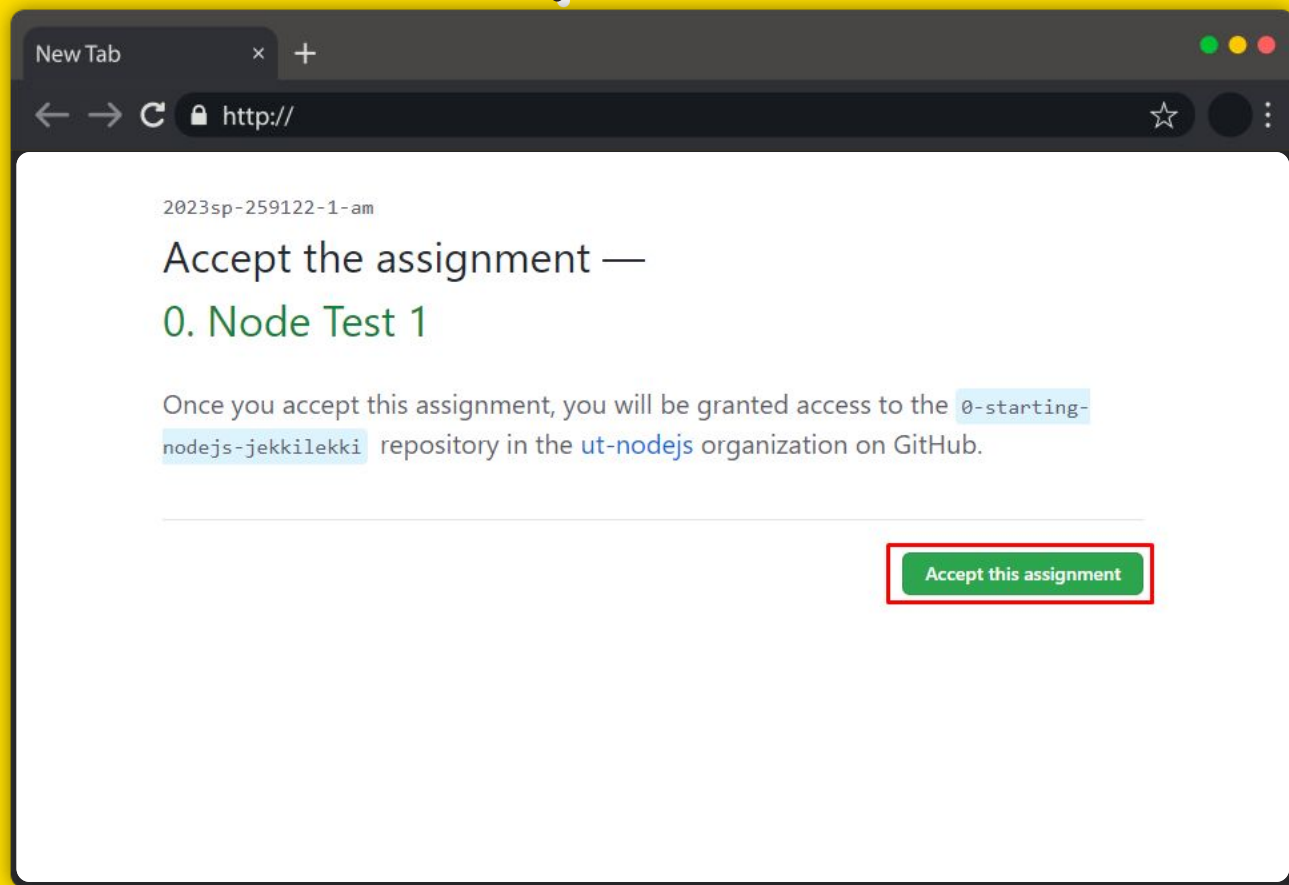
## 학생 보기

1. 홈페이지,
- 2. 과제 참여 링크,**
3. 개별 저장소,
4. GitHub 조직



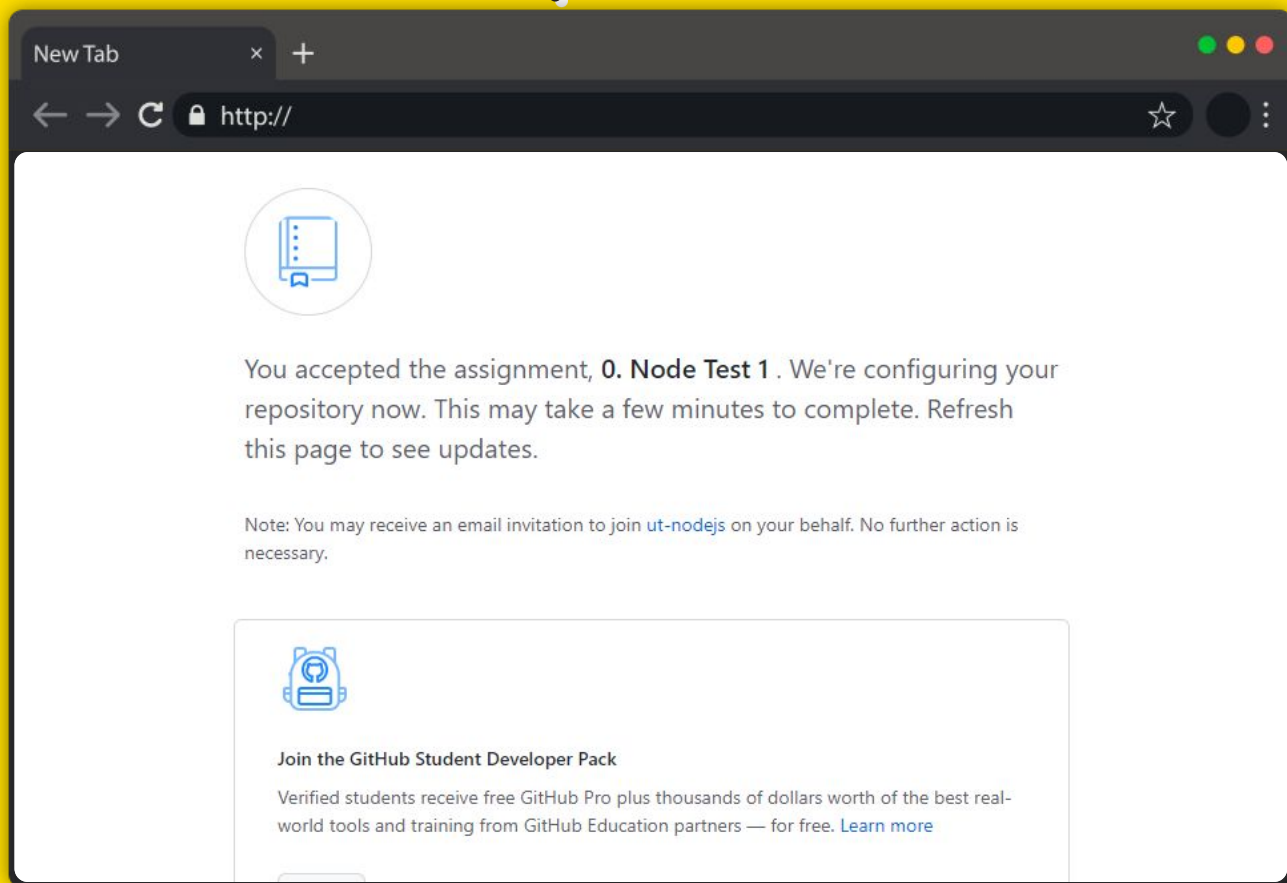
## 학생 보기

1. 홈페이지,
2. 과제 참여 링크,
3. 개별 저장소,
4. GitHub 조직



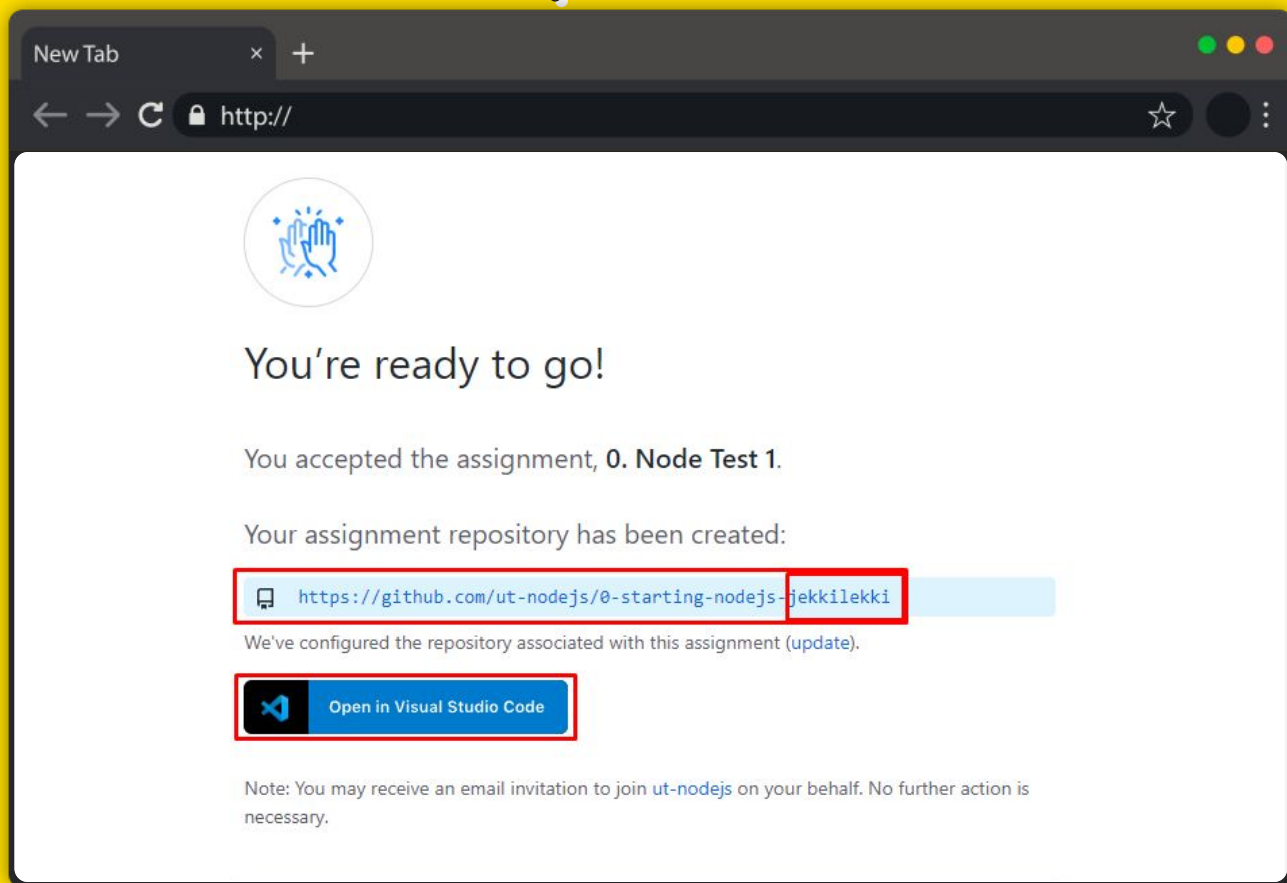
## 학생 보기

1. 홈페이지,
2. 과제 참여 링크,
3. 개별 저장소,
4. GitHub 조직



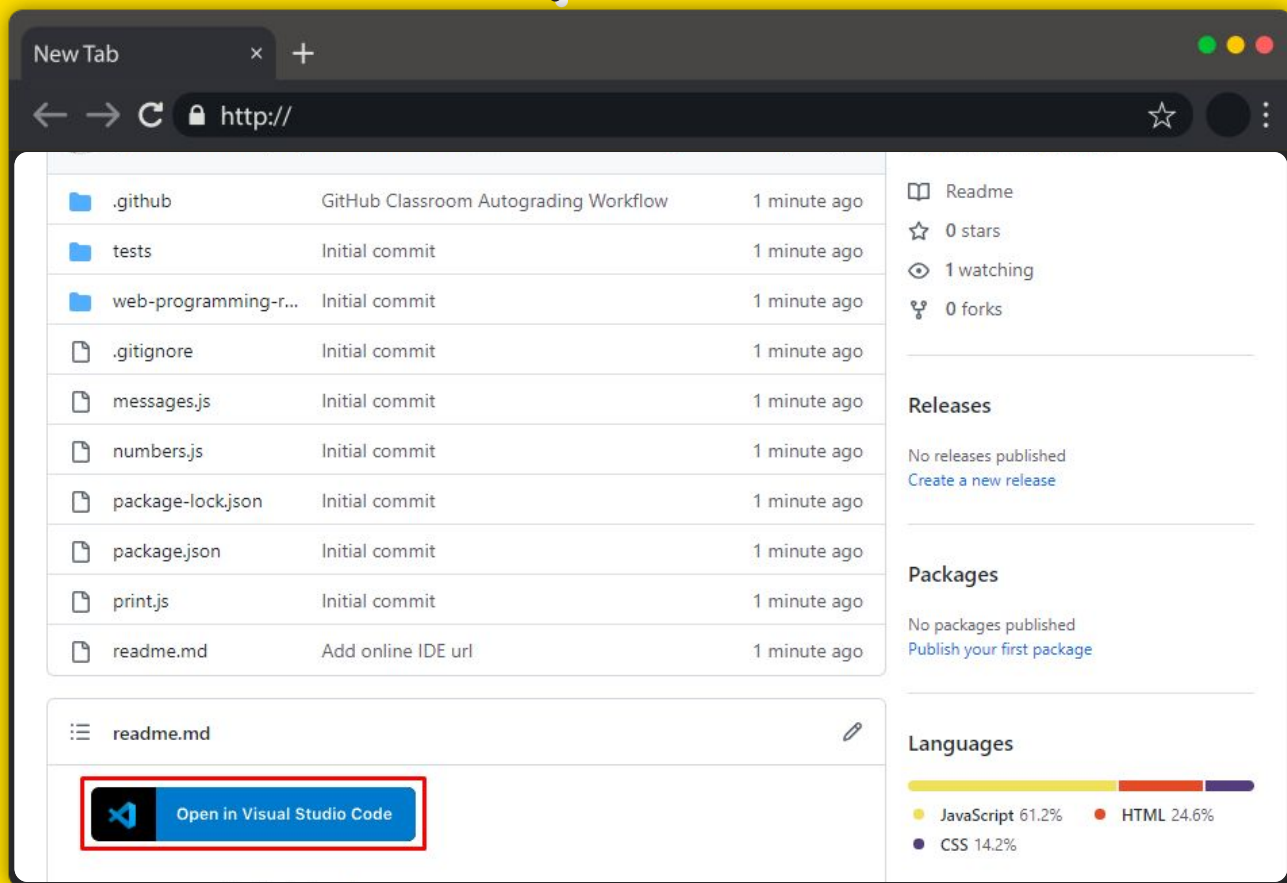
## 학생 보기

1. 홈페이지,
2. 과제 참여 링크,
3. 개별 저장소,
4. GitHub 조직



## 학생 보기

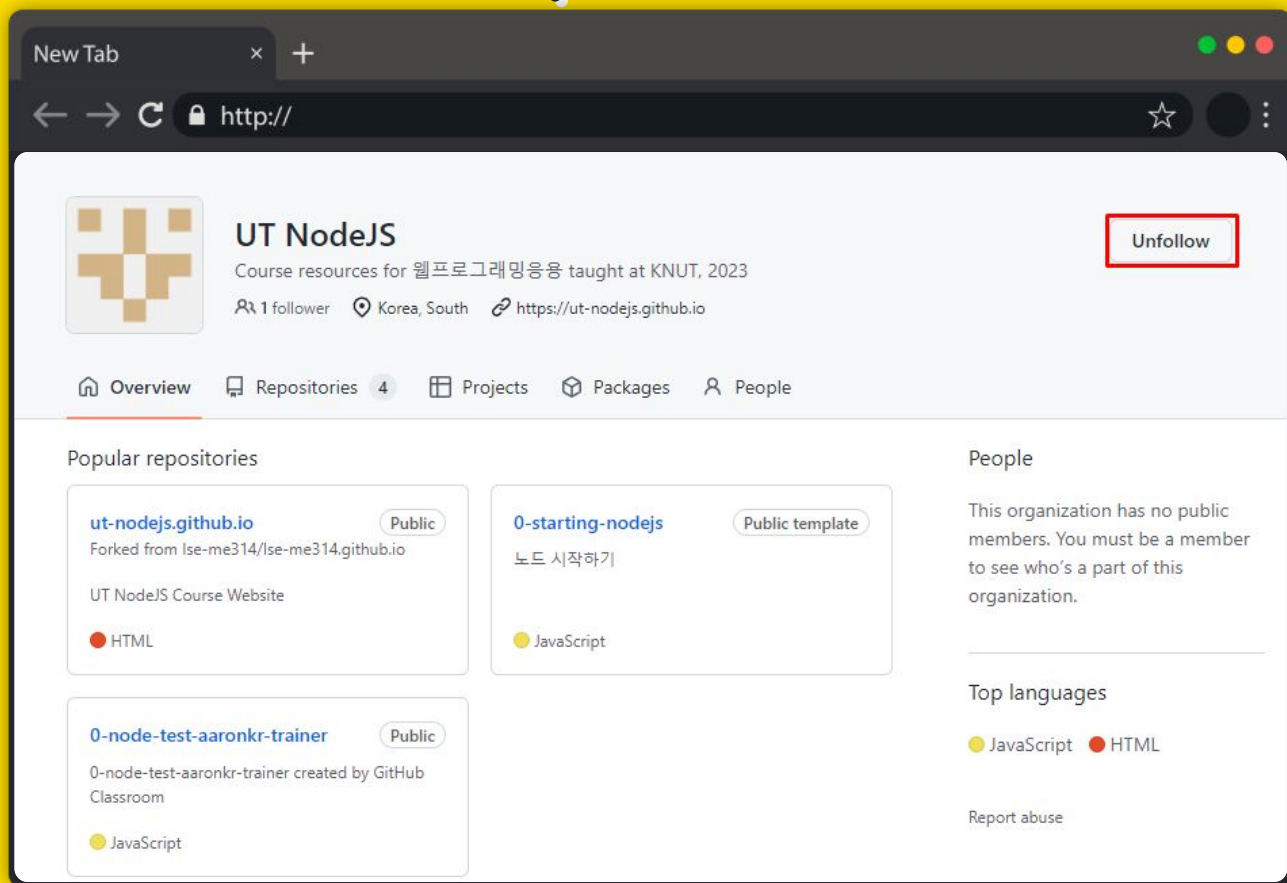
1. 홈페이지,
2. 과제 참여 링크,
- 3. 개별 저장소,**
4. GitHub 조직





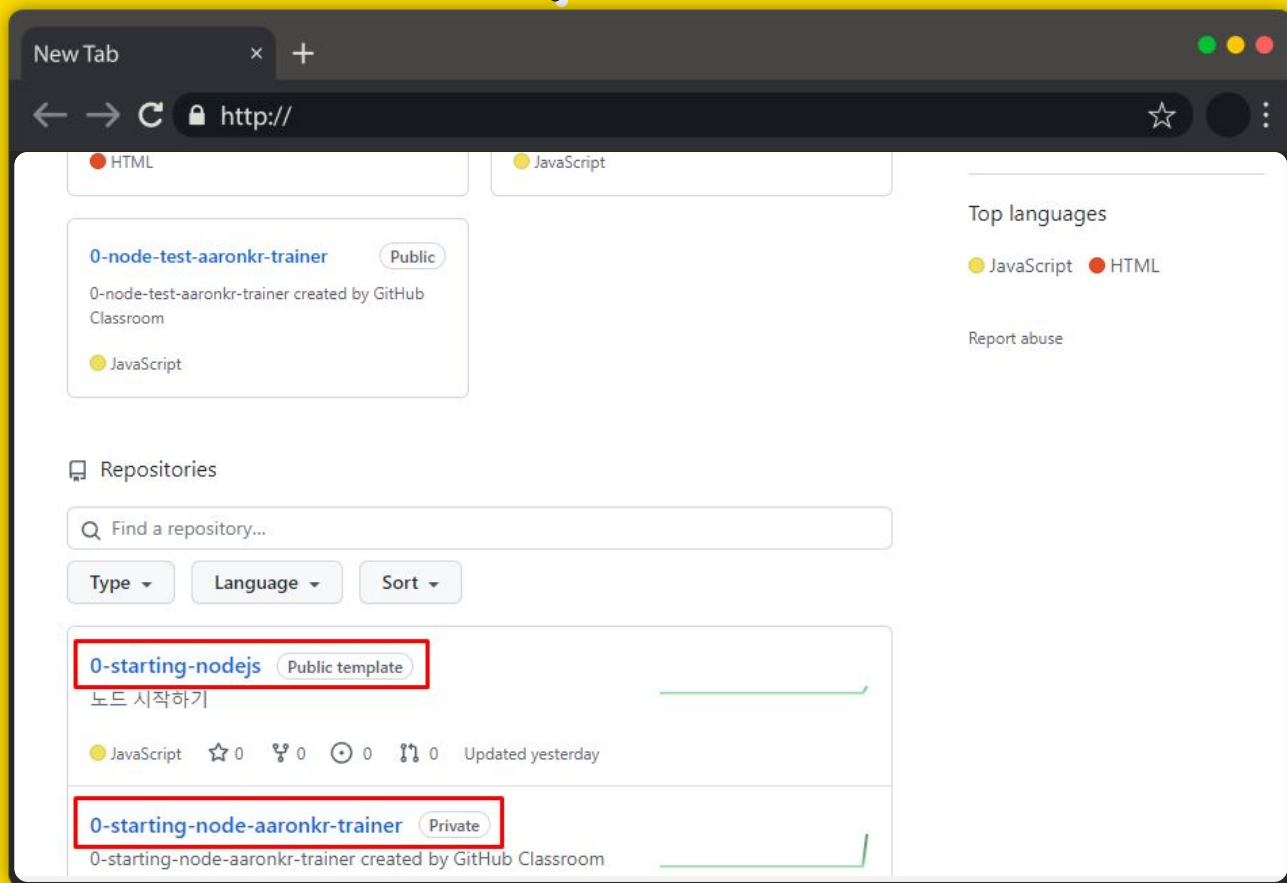
## 학생 보기

1. 홈페이지,
2. 과제 참여 링크,
3. 개별 저장소,
4. **GitHub** 조직



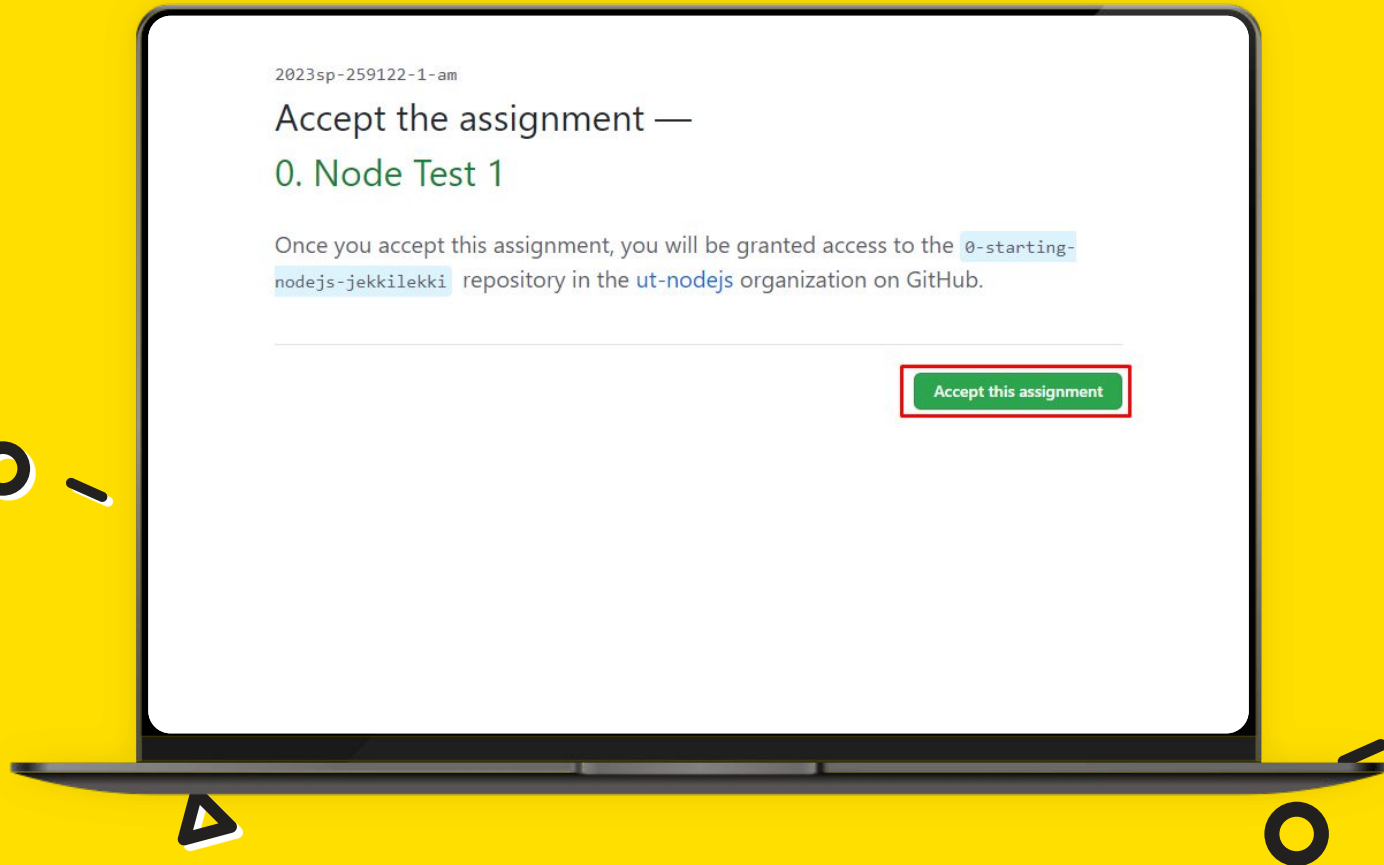
## 학생 보기

1. 홈페이지,
2. 과제 참여 링크,
3. 개별 저장소,
4. **GitHub** 조직



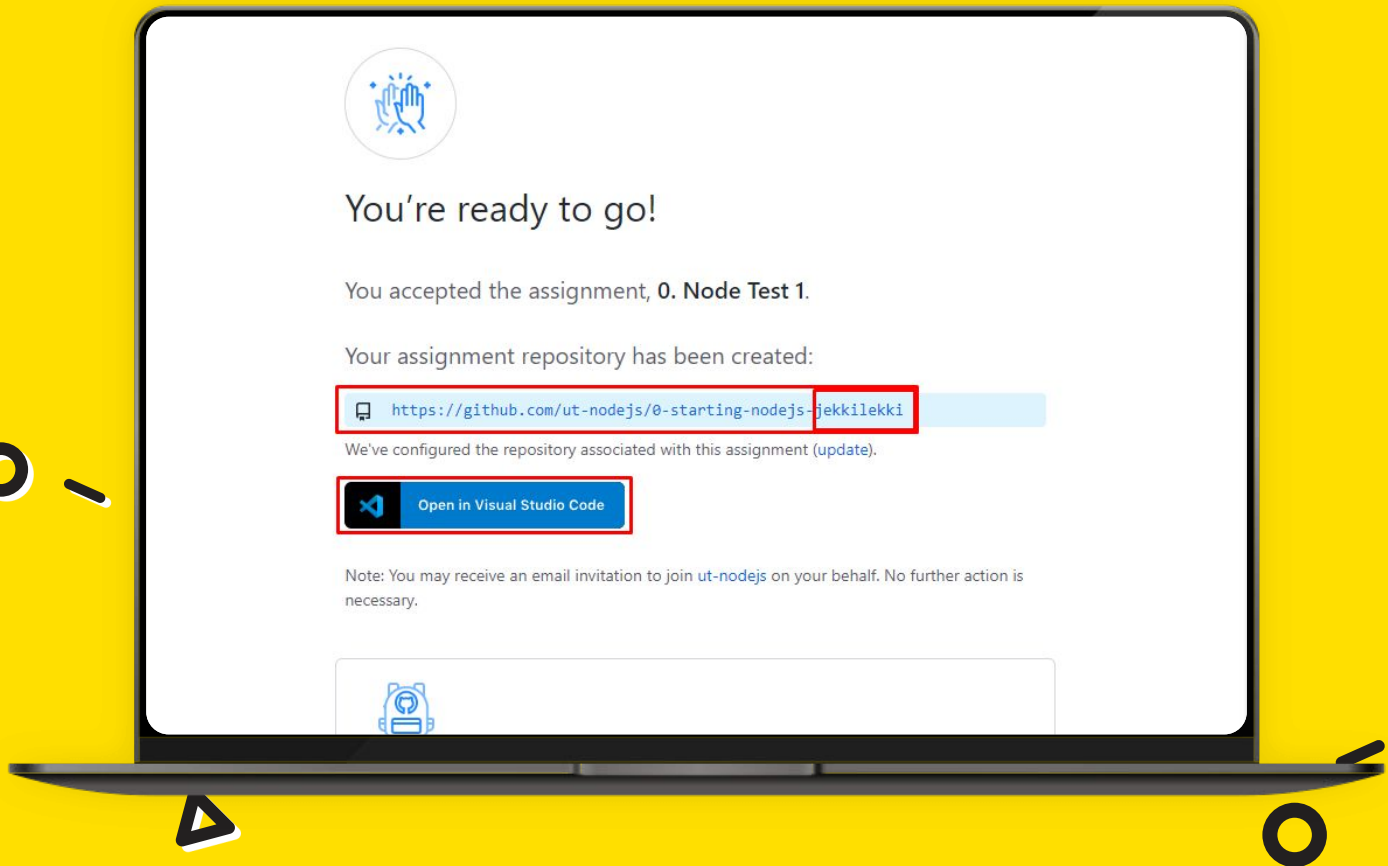
## 과제 제출 방법

1. 과제 참여 링크,
2. VS Code 링크,
3. 코딩,
4. commit & push
5. 테스트 통과 확인



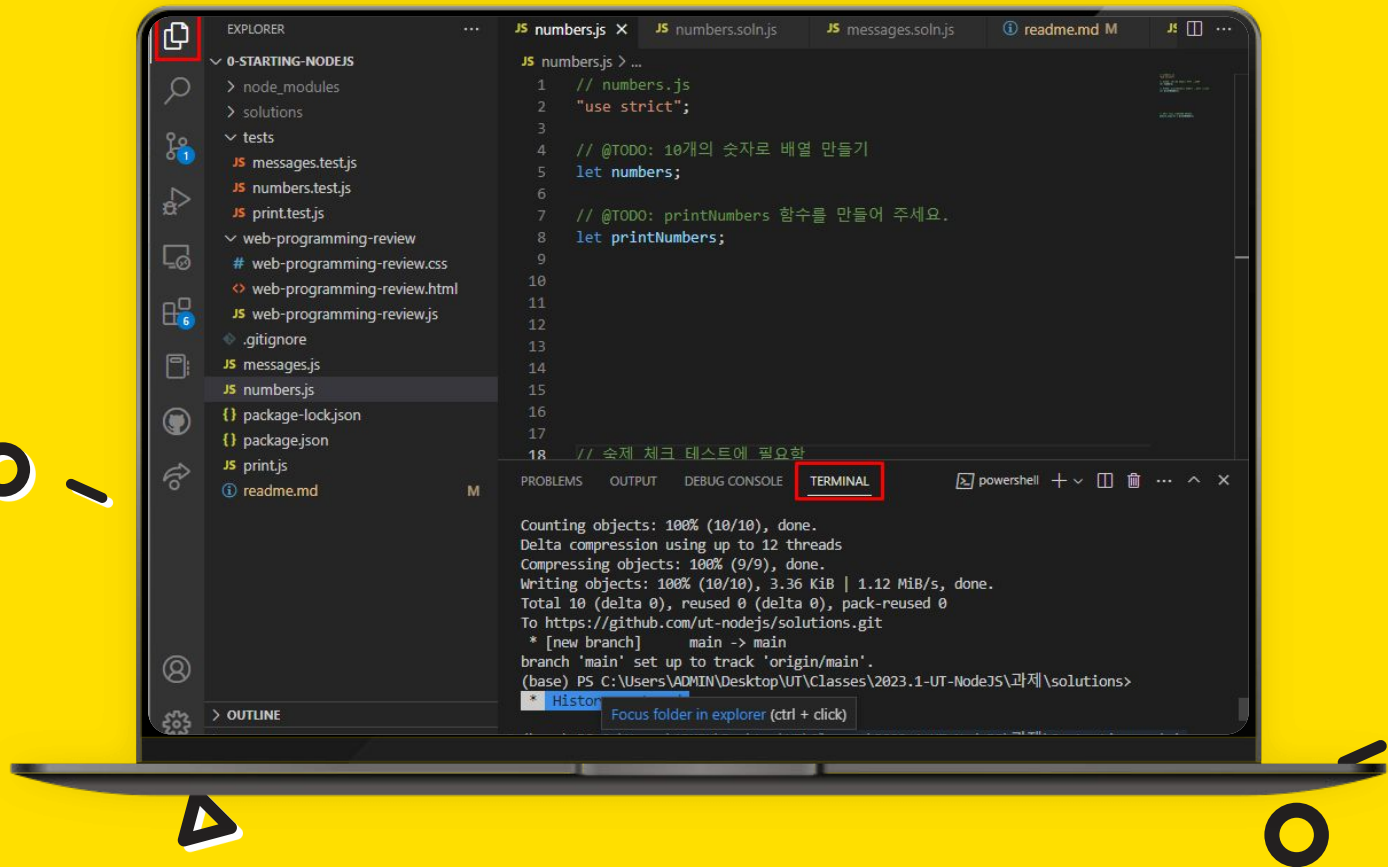
## 과제 제출 방법

1. 과제 참여 링크,
2. **VS Code** 링크,
3. 코딩,
4. commit & push
5. 테스트 통과 확인



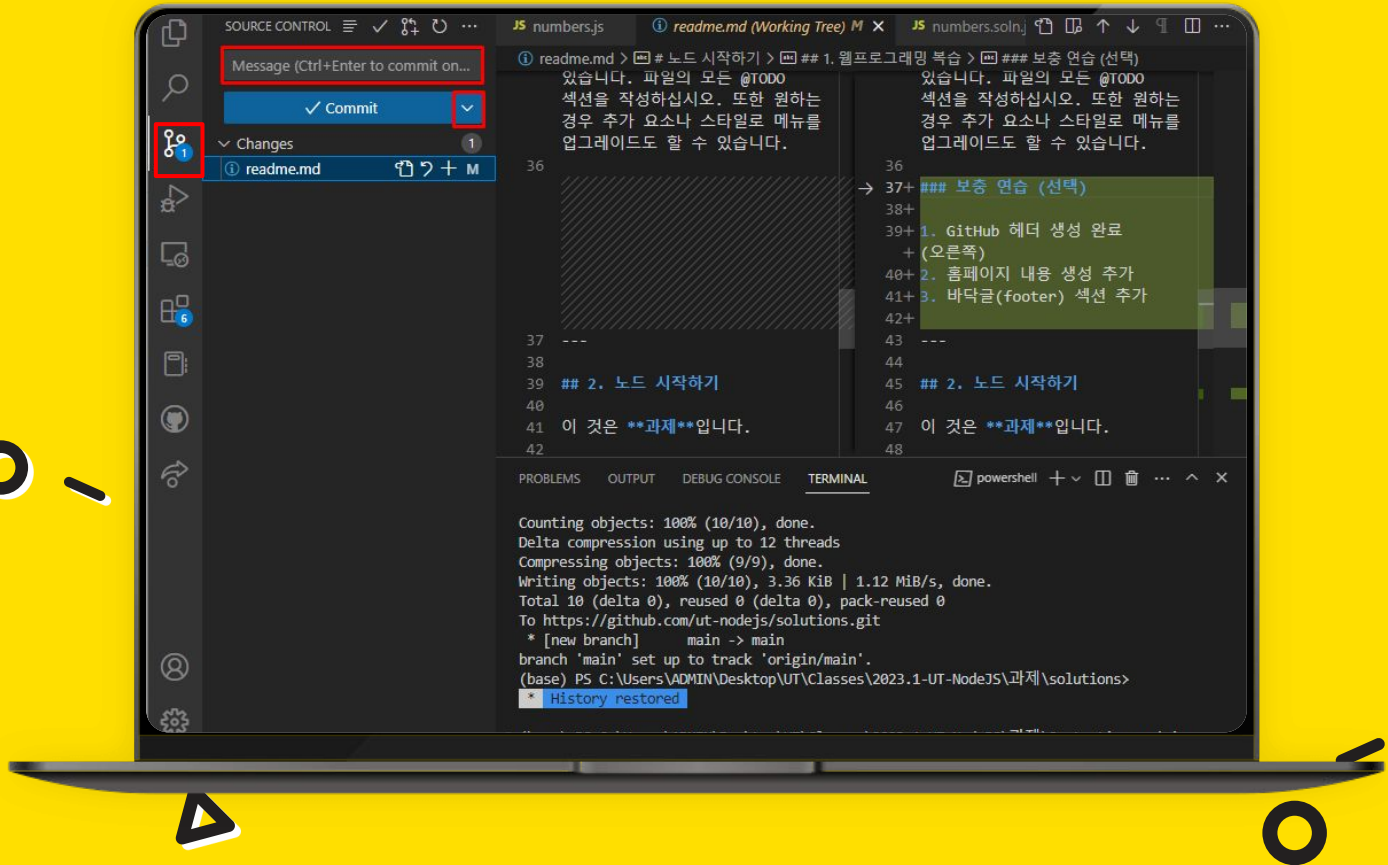
## 과제 제출 방법

1. 과제 참여 링크,
2. VS Code 링크,
3. 코딩,
4. commit & push
5. 테스트 통과 확인



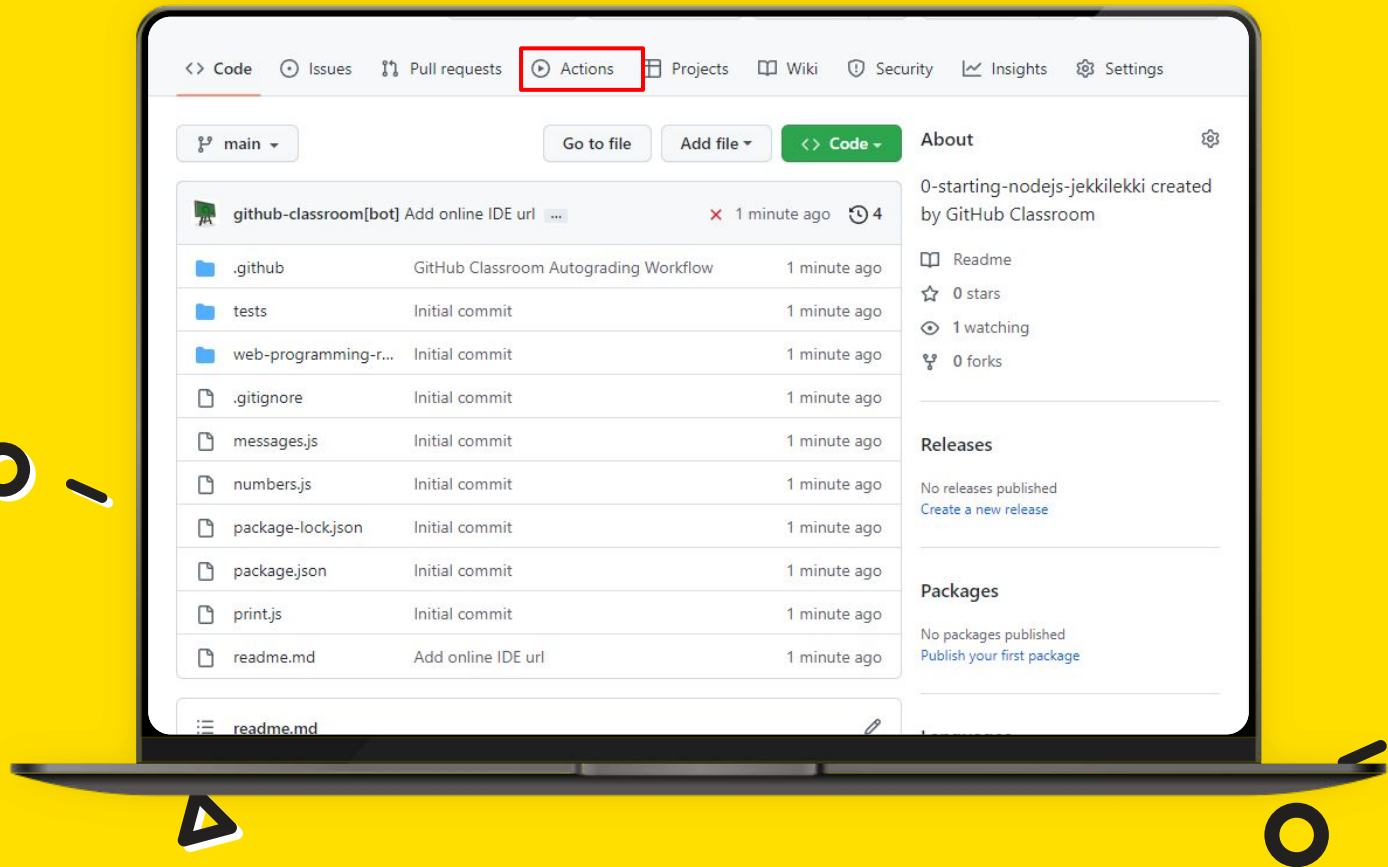
## 과제 제출 방법

1. 과제 참여 링크,
2. VS Code 링크,
3. 코딩,
4. **commit & push**
5. 테스트 통과 확인



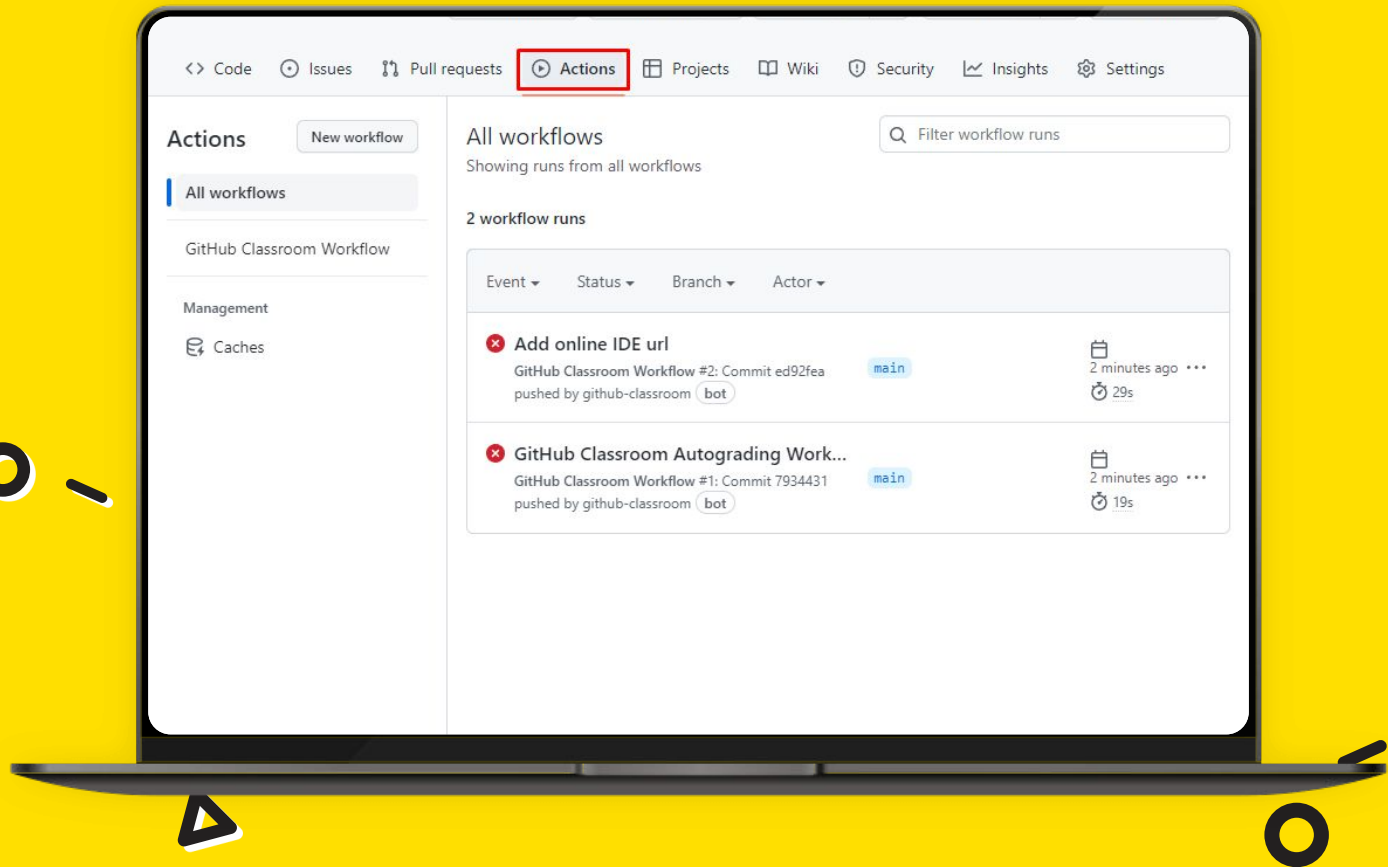
## 과제 제출 방법

1. 과제 참여 링크,
2. VS Code 링크,
3. 코딩,
4. commit & push
5. 테스트 통과 확인



## 과제 제출 방법

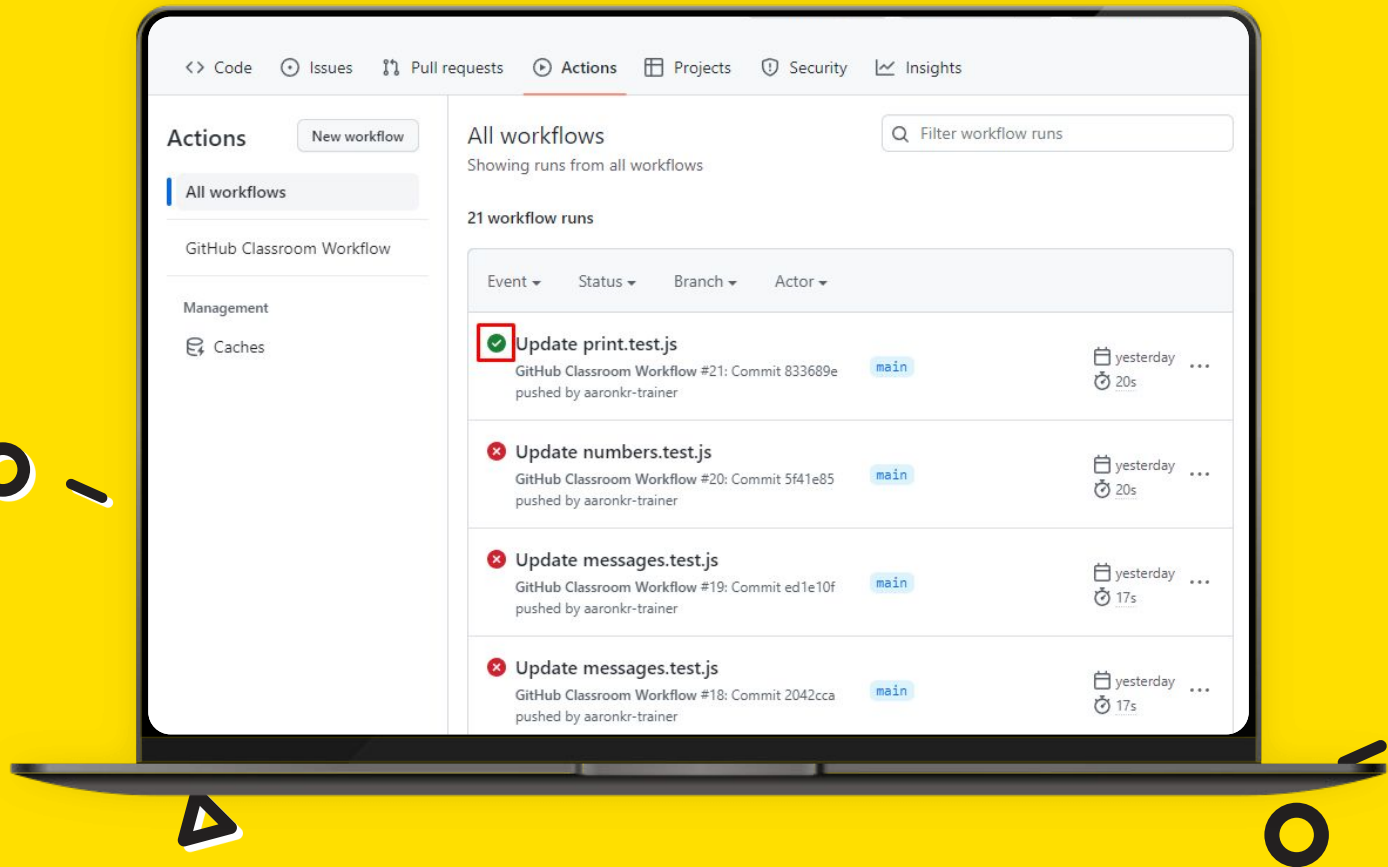
1. 과제 참여 링크,
2. VS Code 링크,
3. 코딩,
4. commit & push
5. 테스트 통과 확인





## 과제 제출 방법

1. 과제 참여 링크,
2. VS Code 링크,
3. 코딩,
4. commit & push
5. 테스트 통과 확인



# Questions?

한번 해 보자!~