

Github + vs Code

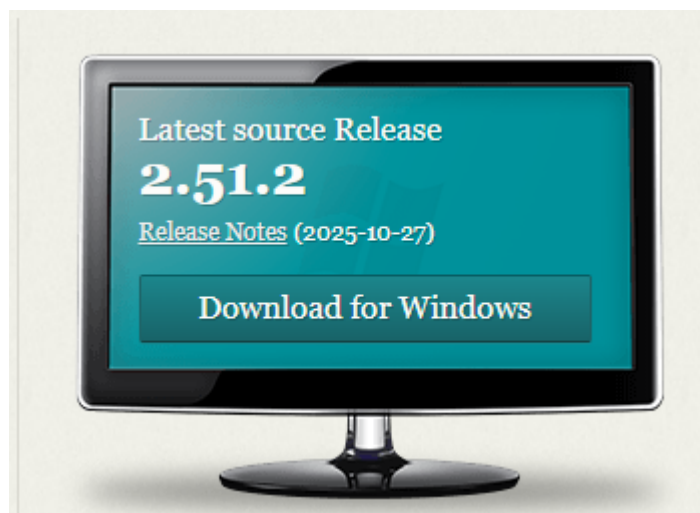
1. 준비하기

1. 깃허브 회원 가입

<https://github.com/>

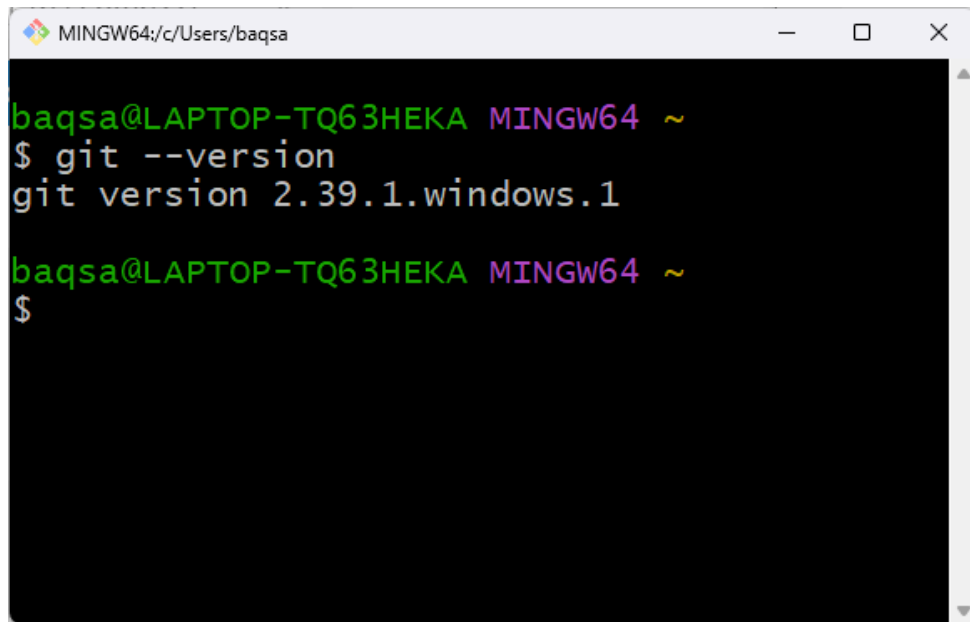
2. 깃 설치

<https://git-scm.com/>



3. 깃 배시에서 깃 버전 확인

```
git --version
```

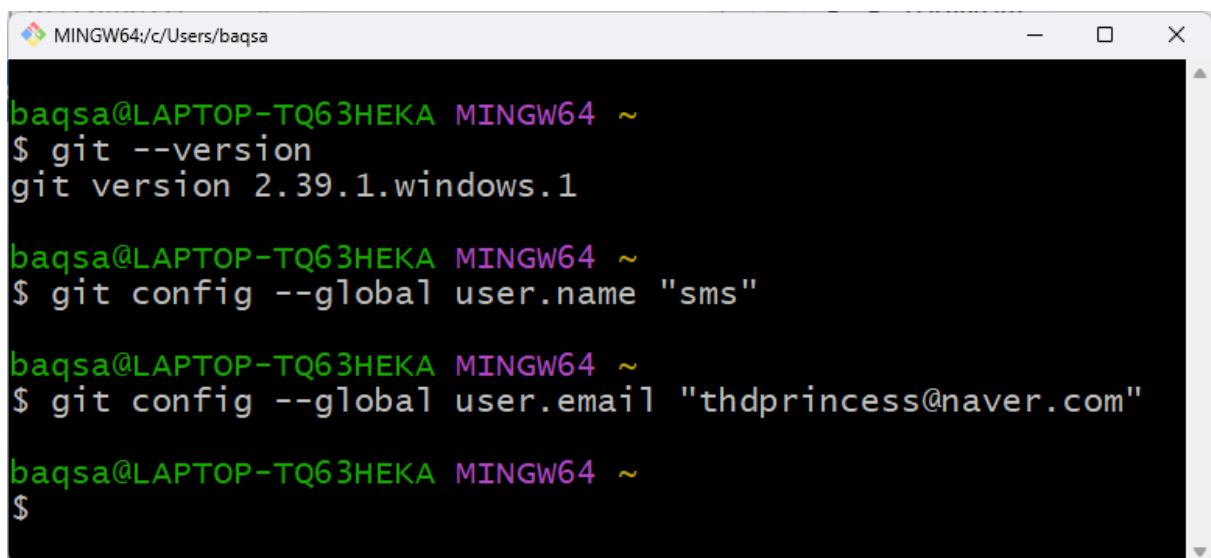
A terminal window titled 'MINGW64:/c/Users/baqsa' showing the command 'git --version' being executed. The output is 'git version 2.39.1.windows.1'.

```
baqsa@LAPTOP-TQ63HEKA MINGW64 ~  
$ git --version  
git version 2.39.1.windows.1  
  
baqsa@LAPTOP-TQ63HEKA MINGW64 ~  
$
```

4. 사용자 정보 설정(1회 설정)

이름, 이메일 주소

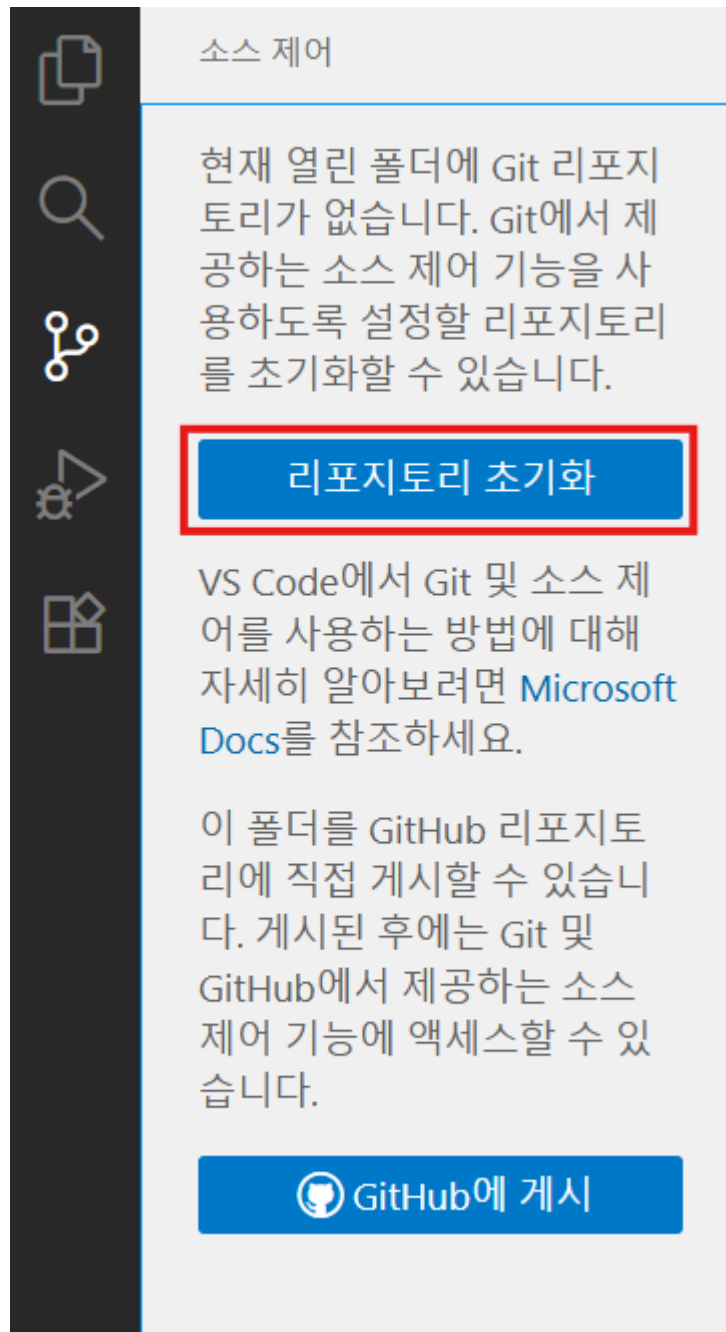
```
git config --global user.name "이름"  
git config --global user.email "이메일 주소"
```

A terminal window titled 'MINGW64:/c/Users/baqsa' showing the execution of 'git --version' followed by 'git config --global user.name "sms"' and 'git config --global user.email "thdprincess@naver.com"'.

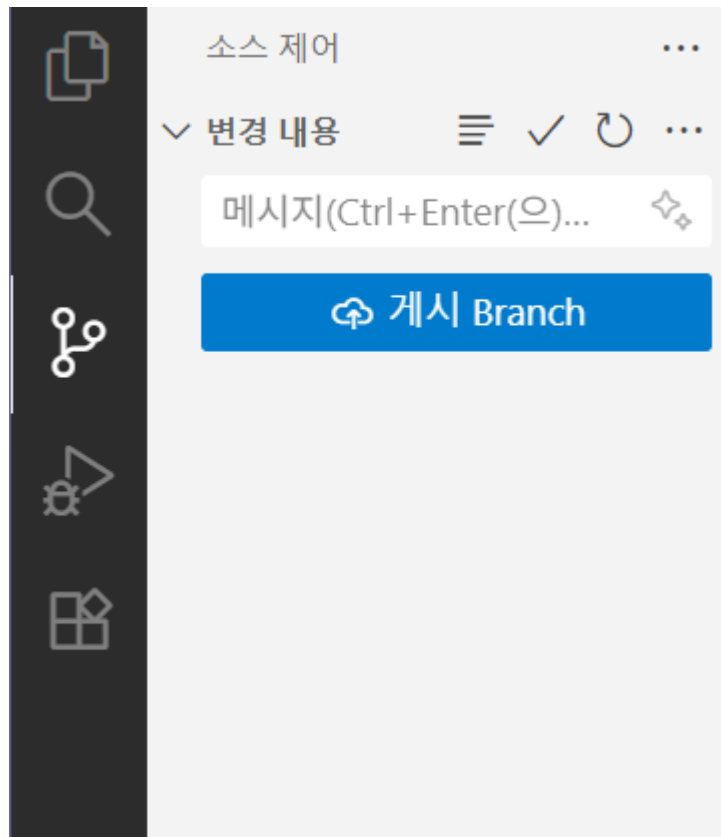
```
baqsa@LAPTOP-TQ63HEKA MINGW64 ~  
$ git --version  
git version 2.39.1.windows.1  
  
baqsa@LAPTOP-TQ63HEKA MINGW64 ~  
$ git config --global user.name "sms"  
  
baqsa@LAPTOP-TQ63HEKA MINGW64 ~  
$ git config --global user.email "thdprincess@naver.com"  
  
baqsa@LAPTOP-TQ63HEKA MINGW64 ~  
$
```

2. 시작하기

1. 폴더 생성 후 vs code 실행
2. 깃 저장소 만들기 - 소스 제어 영역에서 레포지토리 초기화(Initialize Repository)



3. 게시 Branch 확인



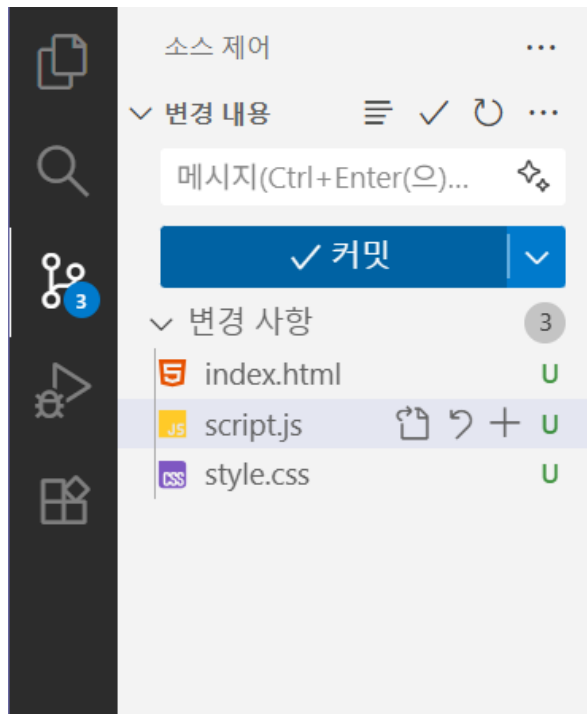
3. 코드 구현하기

4. 첫번째 커밋하기

- 깃 저장소를 만들었기 때문에 코드 변경 사항을 저장하고 추적할 수 있음.
- 코드의 변경 내용이나 파일 이름, 변경 시간 등을 기록해 두는 것을 커밋이라고 함

1. 소스 제어 창 확인

- 생성된 파일명 옆에 U라는 글자 표시
- Untracked(한번도 커밋하지 않았음)



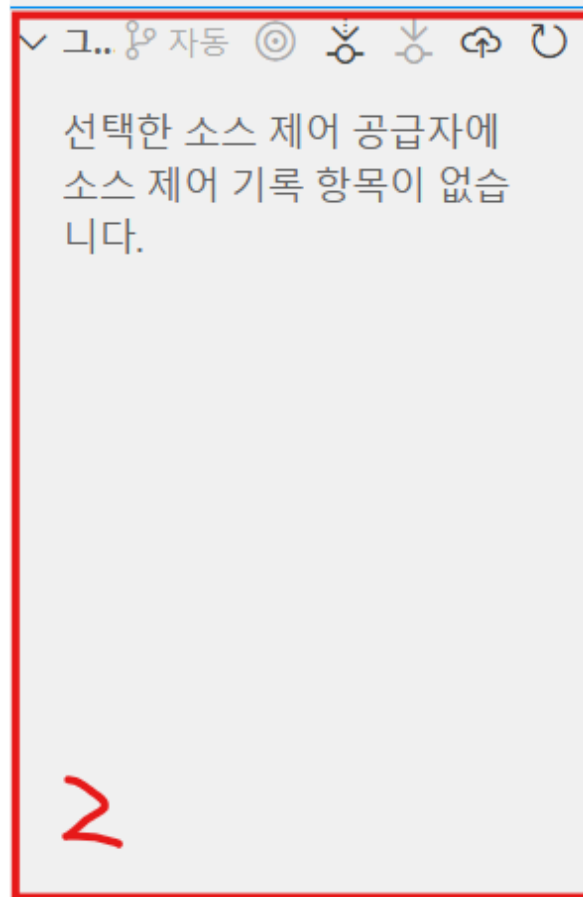
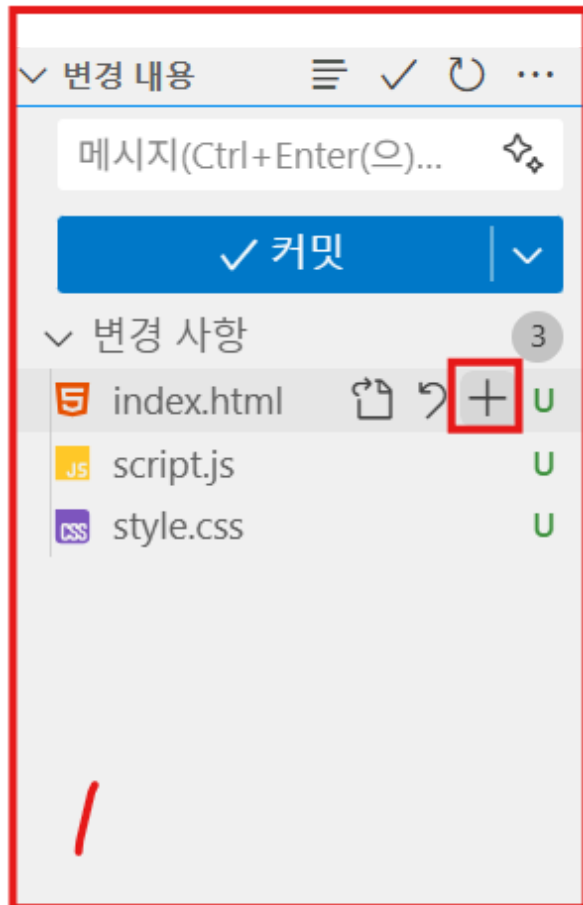
2. 스테이징하기

- 커밋하기 전에 먼저 스테이징 필요
- 커밋할 파일들을 임시 공간인 스테이지에 추가

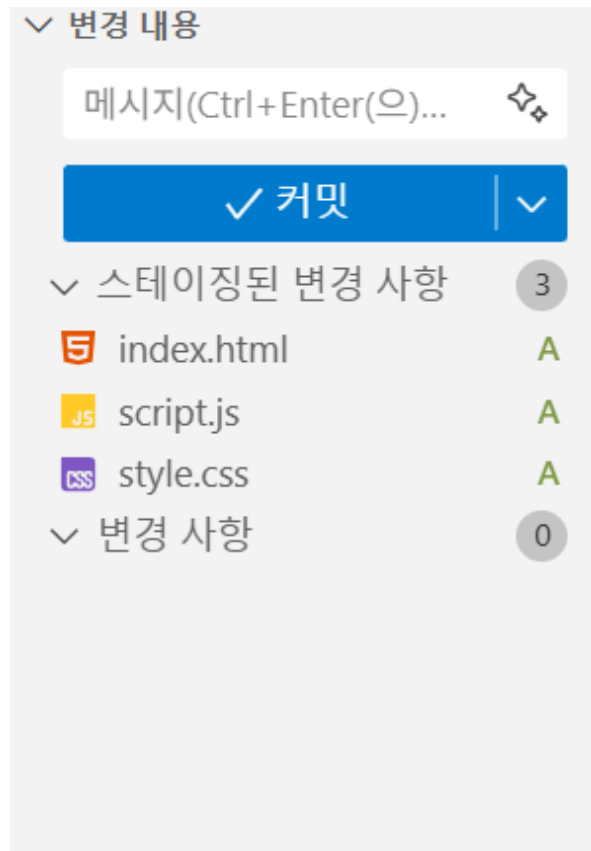
상단영역(1) : Changes
영역

하단영역(2) : Graph 영
역

- index.html 파일 우
측의 + 버튼 클릭
- style.css, script.js
파일 우측의 + 버튼
클릭

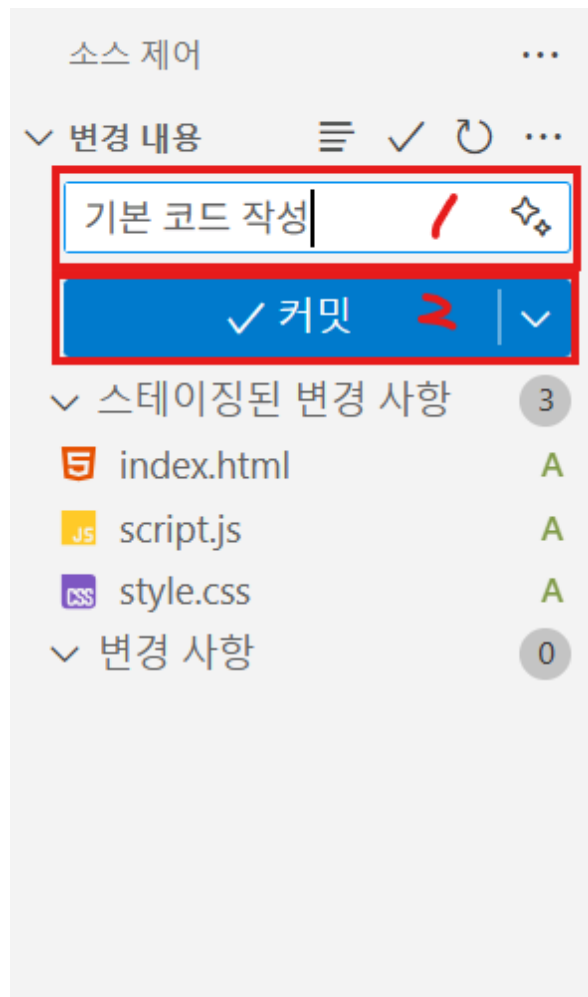


3. 스테이징 확인



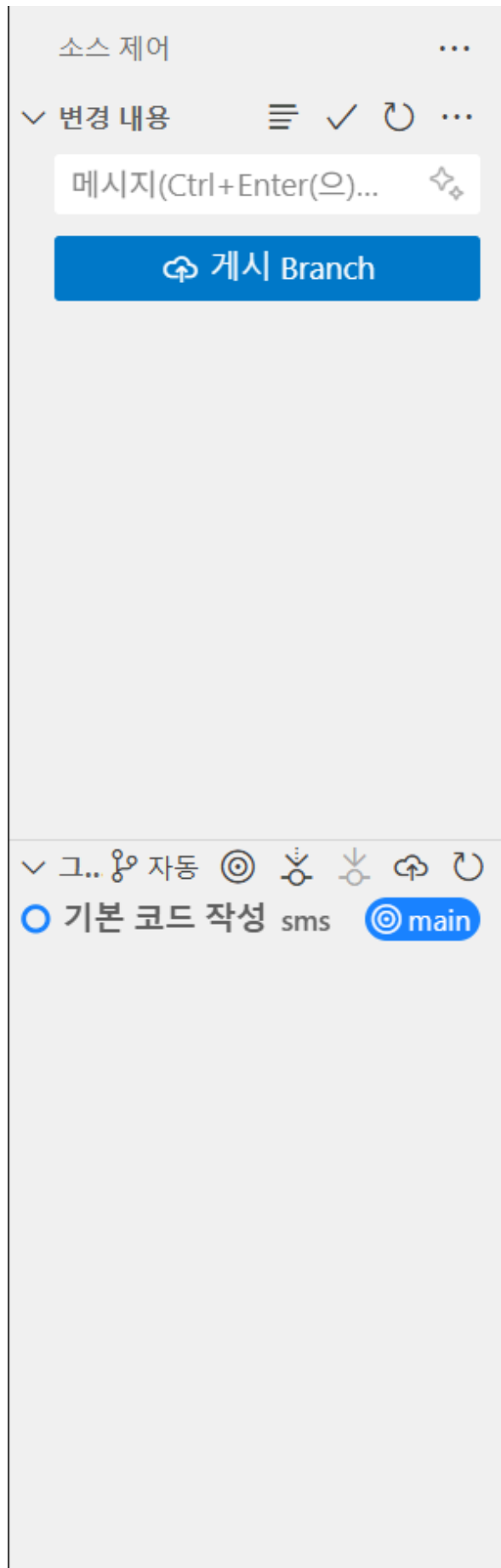
4. 커밋하기

- 커밋 메시지 입력 후 커밋 버튼 클릭

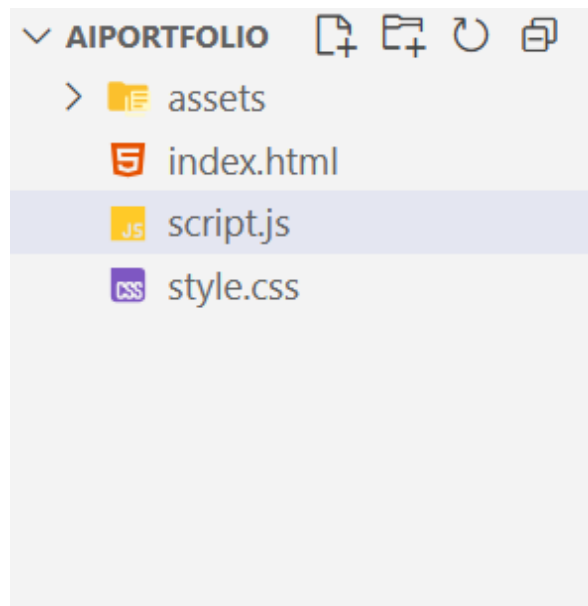


5. 커밋 후 각 영역의 모습 확인하기

- Graph 영역에 커밋 내용
확인 가능
- 커밋 메시지 위로 hover
시 변경 내용 요약



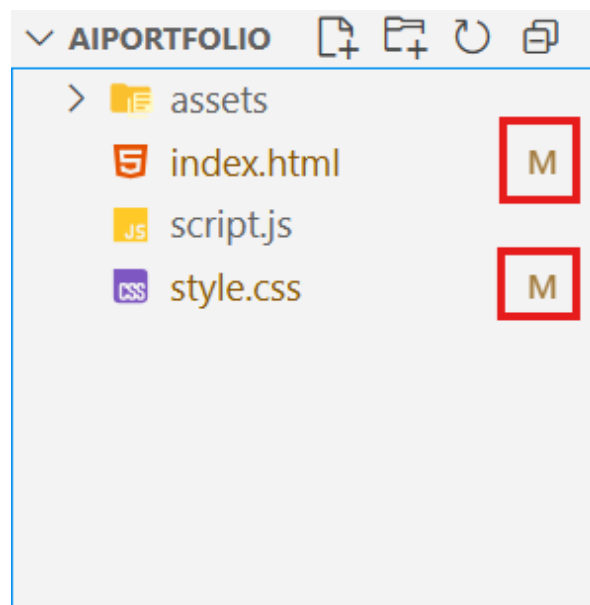
6. 파일 탐색창의 U 제거 확인



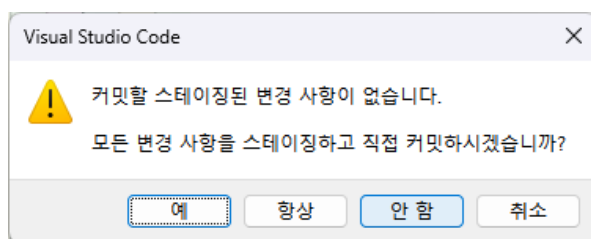
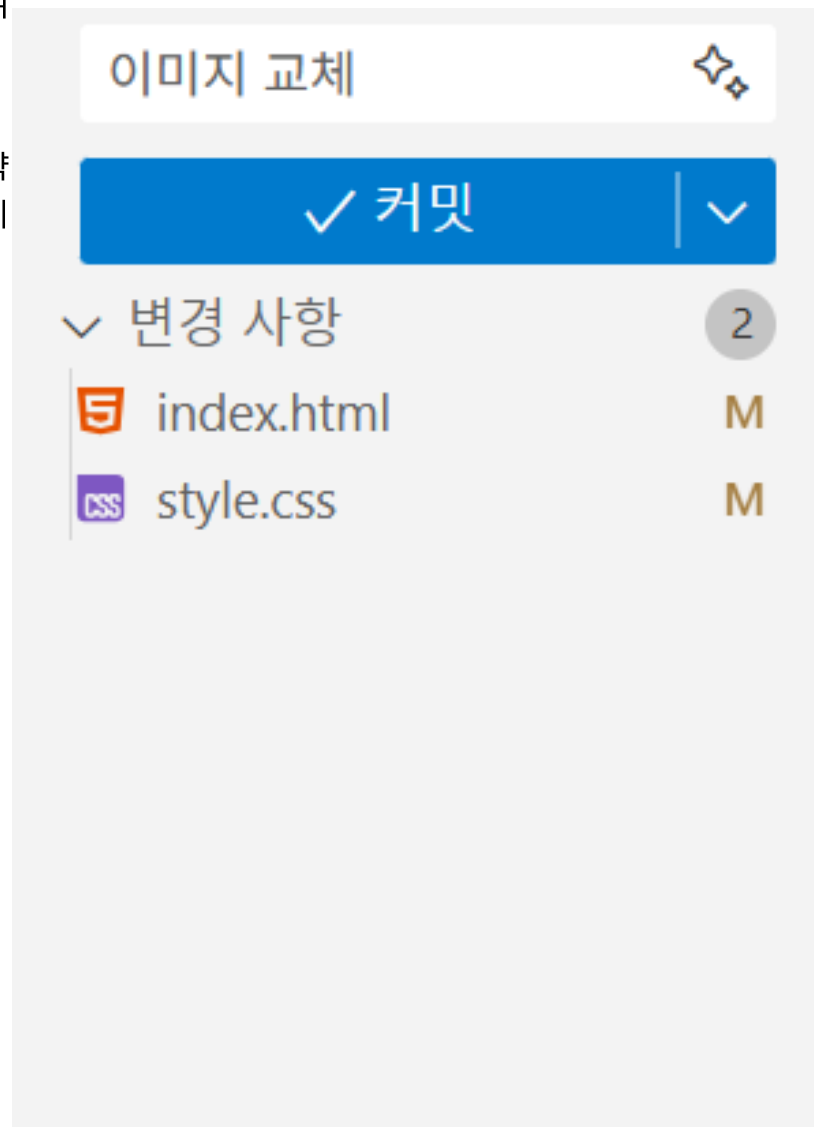
5. 변경 사항 발생시 수정하기

6. 커밋하기

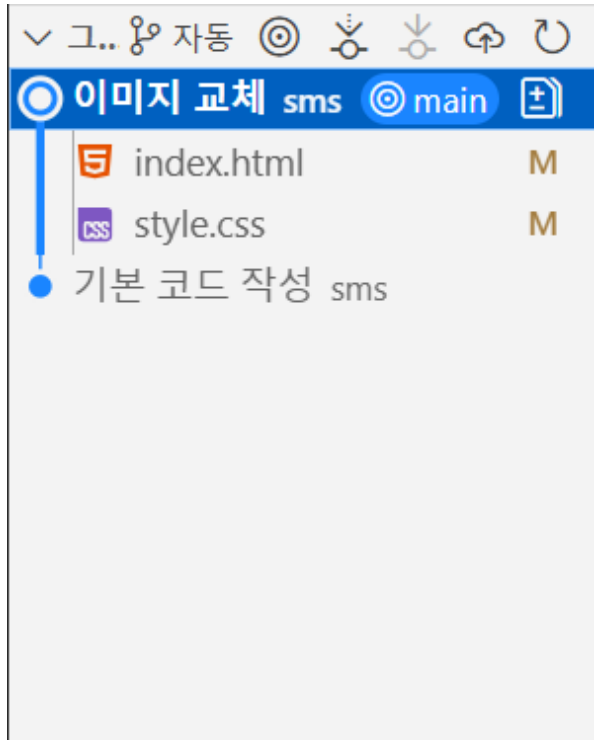
- 수정된 내용 발생



- 소스 제어창으로 이동하여 커밋
- 기본 순서는 스테이징 후 커밋이며 스테이징을 생략할 수 있음 ⇒ 메시지창 제공



- Graph 창에서 커밋 내용 확인 가능



- 파일 탐색 창의 M 제거 확인

7. 개념이해

- 위의 과정들은 해당 작업 폴더 내의 로컬 저장소에 코드의 변경내용이나 파일 이름, 변경 시간 등을 기록하여 저장한 과정들임
- 추후 진행 과정은 원격 저장소에 저장하는 방법에 대한 것임 ⇒ 배포

8. 깃허브에 저장소 만들기

- 깃허브 로그인 후 New repository 생성

예) aiPortfolio

- 웹브라우저의 주소 표시줄에서 현재 저장소의 주소 확인 가능 ⇒ 복사

<https://github.com/계정/저장소명>

예) <https://github.com/thdprincess/aiPortfolio>

- 생성 페이지 내에서 정보 제공 확인 가능

Quick setup — if you've done this kind of thing before

Set up in Desktop

or

HTTPS

SSH

<https://github.com/thdprincess/aiPortfolio.git>

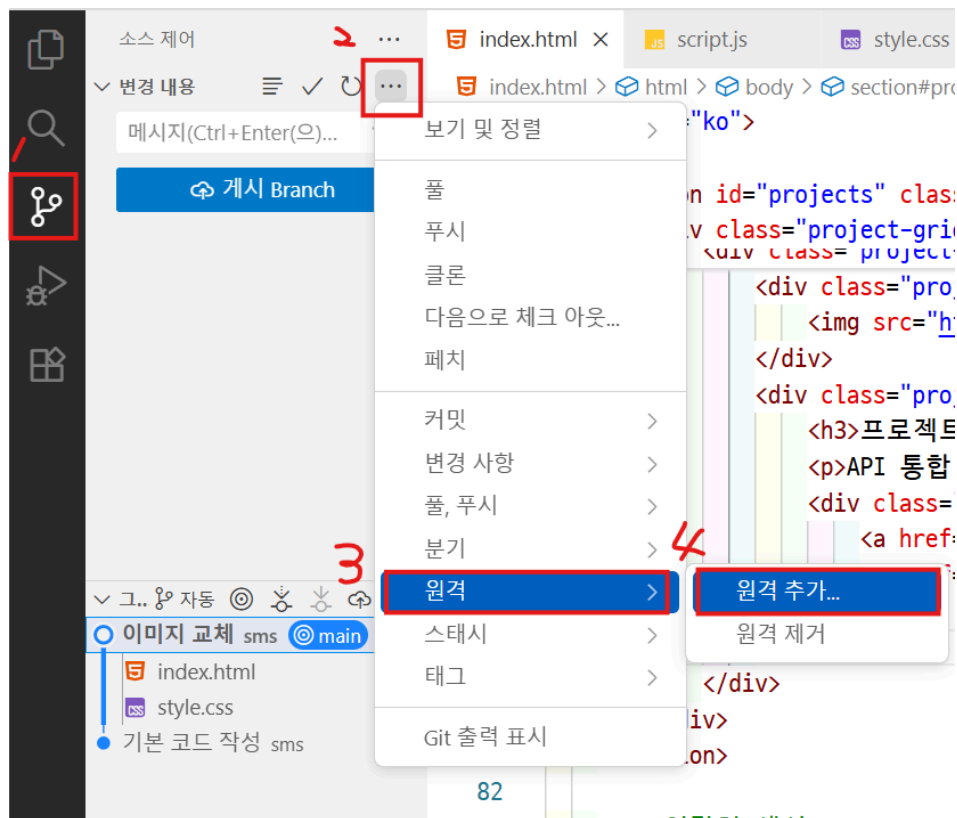


Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

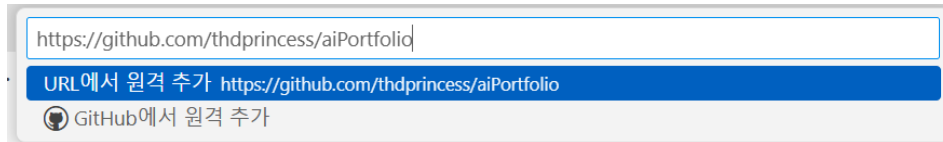
9. vs Code와 깃허브 연동하기

- vs code와 깃허브를 연동하면 몇 번 클릭만으로 깃허브에 코드 업로드 가능하여 개발 과정에서 더욱 효율적으로 관리 가능

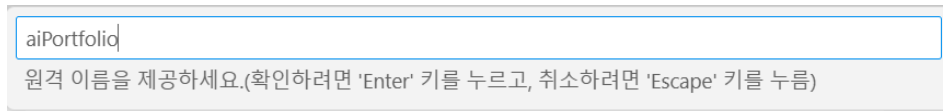
- 소스 제어 창 전환하여 changes 영역의 제목 위로 마우스 포인터 올려 더보기 아이콘 (...) 클릭
- 메뉴 중 Remote → Add Remote 클릭



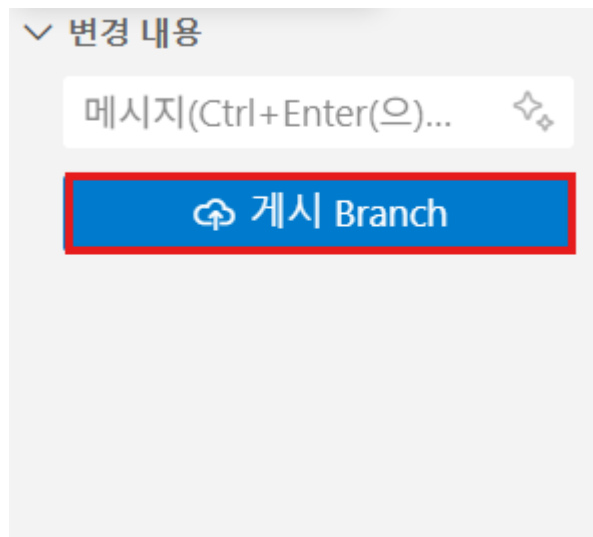
- 복사했던 원격 저장소를 붙여 넣고 enter



4. 원격 이름 동일하게 부여 후 enter

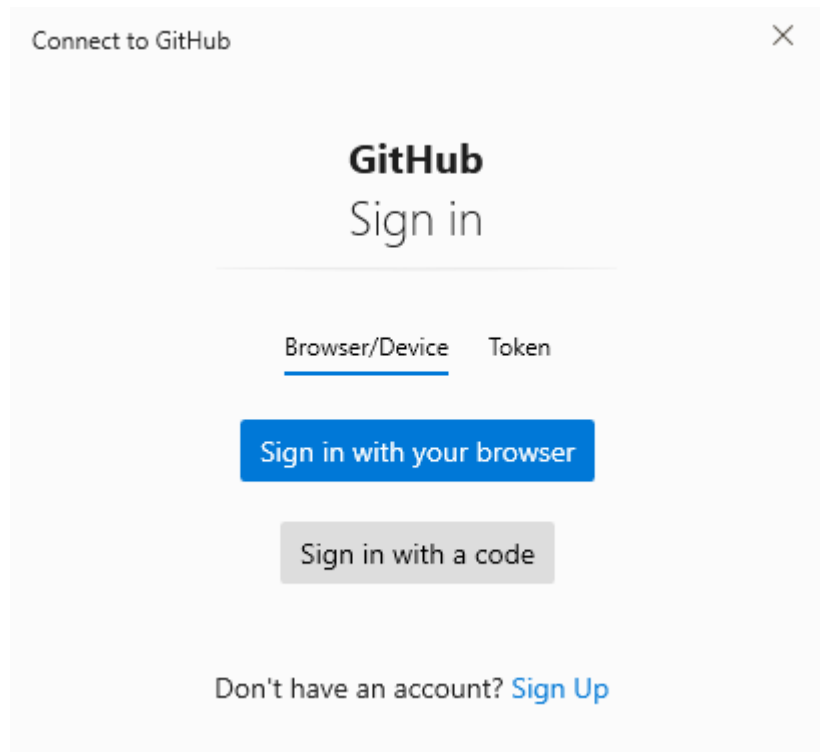


5. 로컬 저장소에 만들었던 커밋을 깃허브에 업로드하기 위해 소스 제어 창에서 게시 Branch(Publish Branch) 클릭

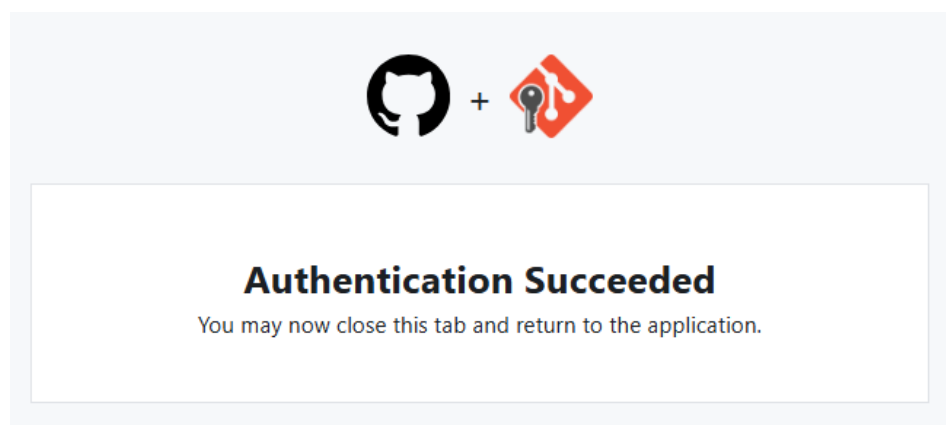


6. 깃허브 연동 알림은 [Allow] 클릭

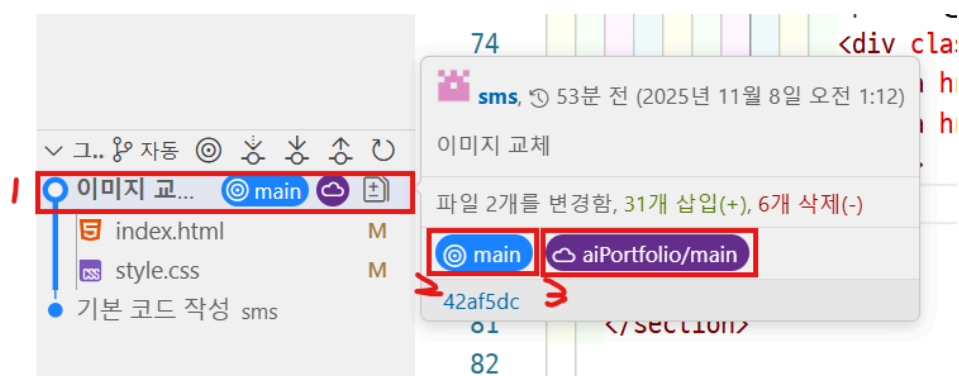
7. 깃허브 로그인 창은 [Sign in with your browser] 클릭



8. 로그인 성공 화면 확인



9. Graph 영역의 커밋 목록 확인



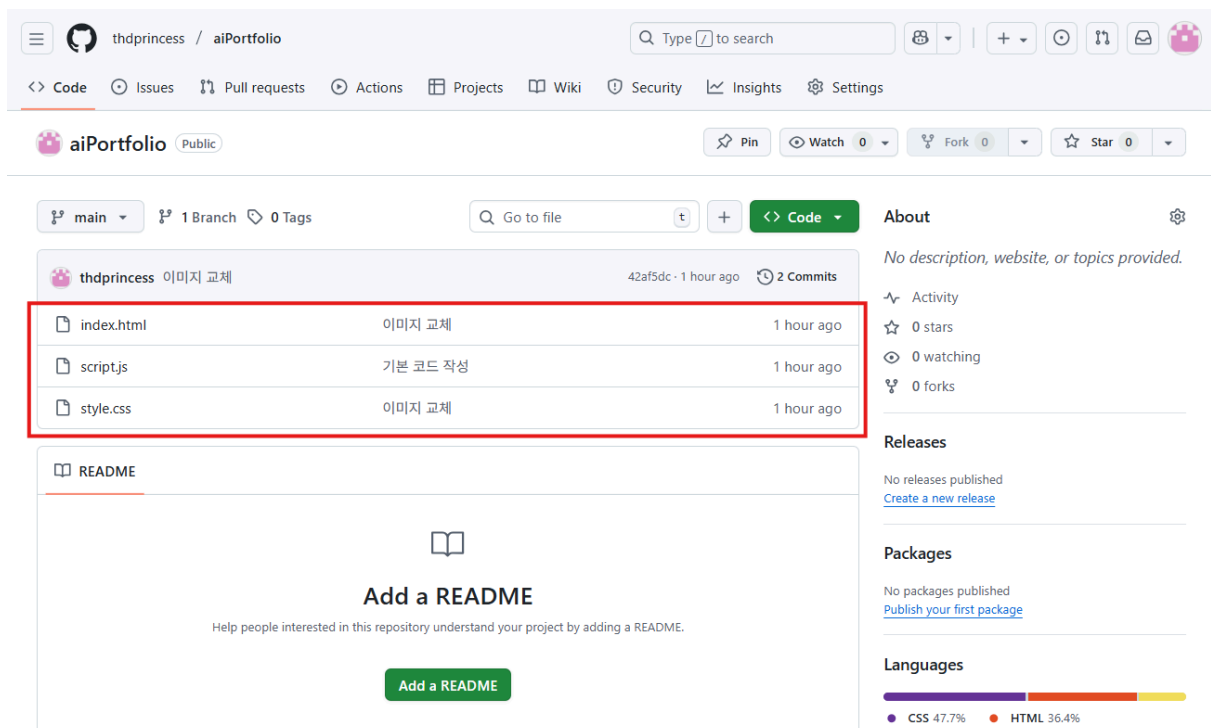
main(2) : 컴퓨터의 저장소

aiPortfolio/main(3) : 깃허브의 저장소

- main 브랜치 오른쪽에 구름 아이콘 생성

⇒ 깃허브 저장소에 코드가 업로드되었고, 깃허브 저장소에도 같은 커밋이 있음 의미

10. 깃허브 저장소 접속하여 파일 업로드 확인

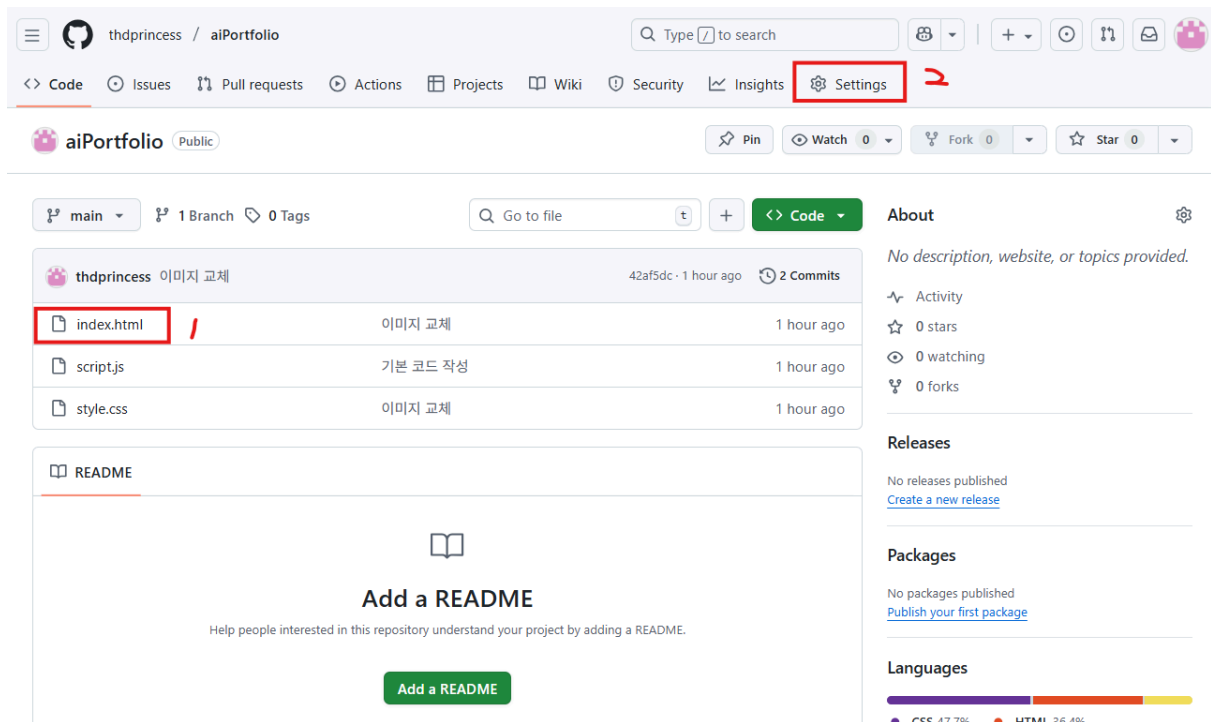


11. 추후 업로드 방법

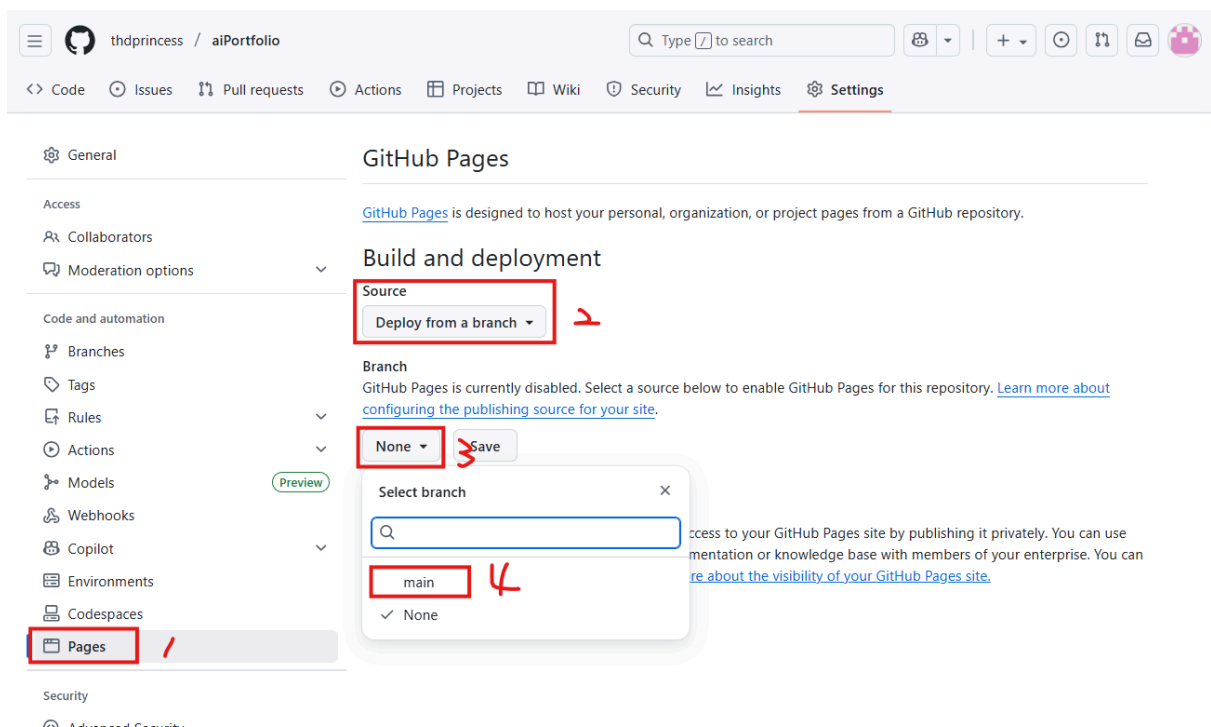
- 현재 컴퓨터의 로컬 저장소와 깃허브의 원격 저장소가 연결된 상태
- 파일 수정 후 메시지와 함께 커밋하고, 게시 Branch(Publish Branch) 클릭하여 깃허브로 업로드

10. 깃허브 페이지 배포

1. 깃허브 저장소에 index.html 파일 존재 확인
2. 웹 브라우저에서 깃허브 저장소 접속 - Setting



3. pages - source 항목 : Deploy from a branch 확인 - Branch : none → main 선택 후 save 클릭



4. 일정 시간 후 설정 화면의 가운데 위쪽에 웹 사이트 주소 표시

<https://계정.github.io/저장소명>

예) <https://thdprincess.github.io/aiPortfolio>

thdprincess / aiPortfolio

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

General

Access

Collaborators

Moderation options

Code and automation

Branches

Tags

Rules

Actions

Models

Webhooks

Copilot

Environments

Codespaces

Pages

Security

Advanced Security

GitHub Pages

GitHub Pages is designed to host your personal, organization, or project pages from a GitHub repository.

Your site is live at <https://thdprincess.github.io/aiPortfolio/>
Last deployed by thdprincess 2 minutes ago

Visit site Unpublish site

Build and deployment

Source

Deploy from a branch

Branch

Your GitHub Pages site is currently being built from the main branch. [Learn more about configuring the publishing source for your site.](#)

main / (root) Save

Learn how to [add a Jekyll theme](#) to your site.

Your site was last deployed to the [github-pages](#) environment by the [pages build and deployment](#) workflow. [Learn more about deploying to GitHub Pages using custom workflows](#)

Custom domain

Custom domains allow you to serve your site from a domain other than [github-pages.github.io](#). [Learn more about](#)