

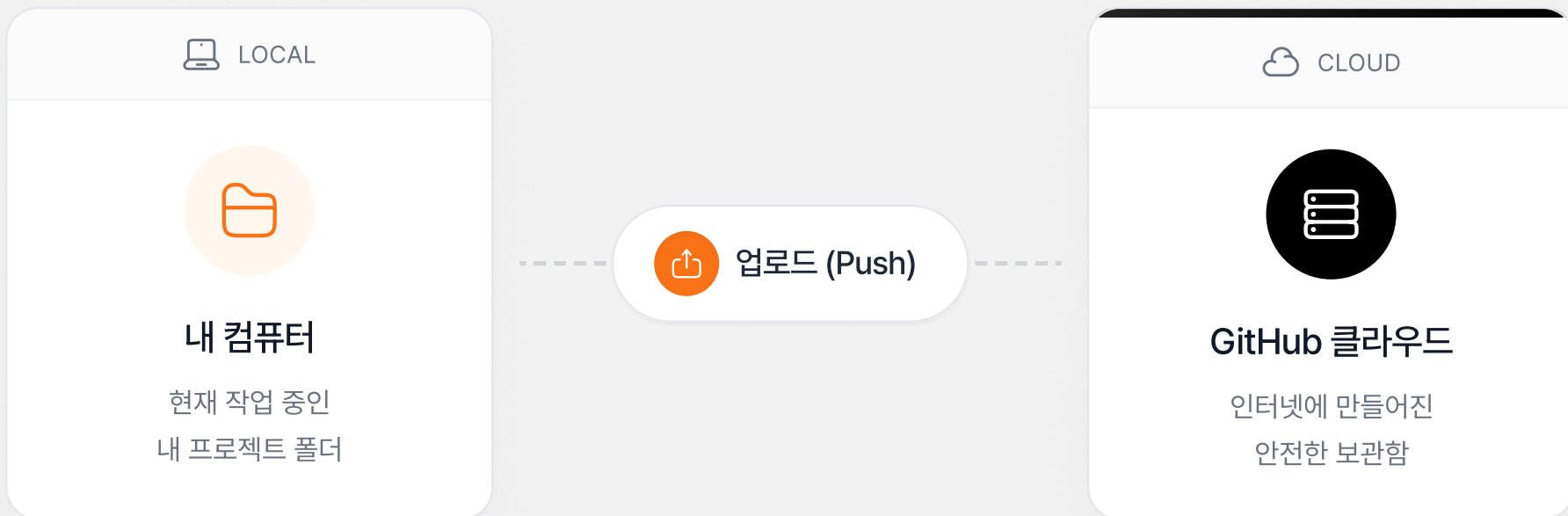
● 환경설정

Github연결

Github와 Claude 연결 안내

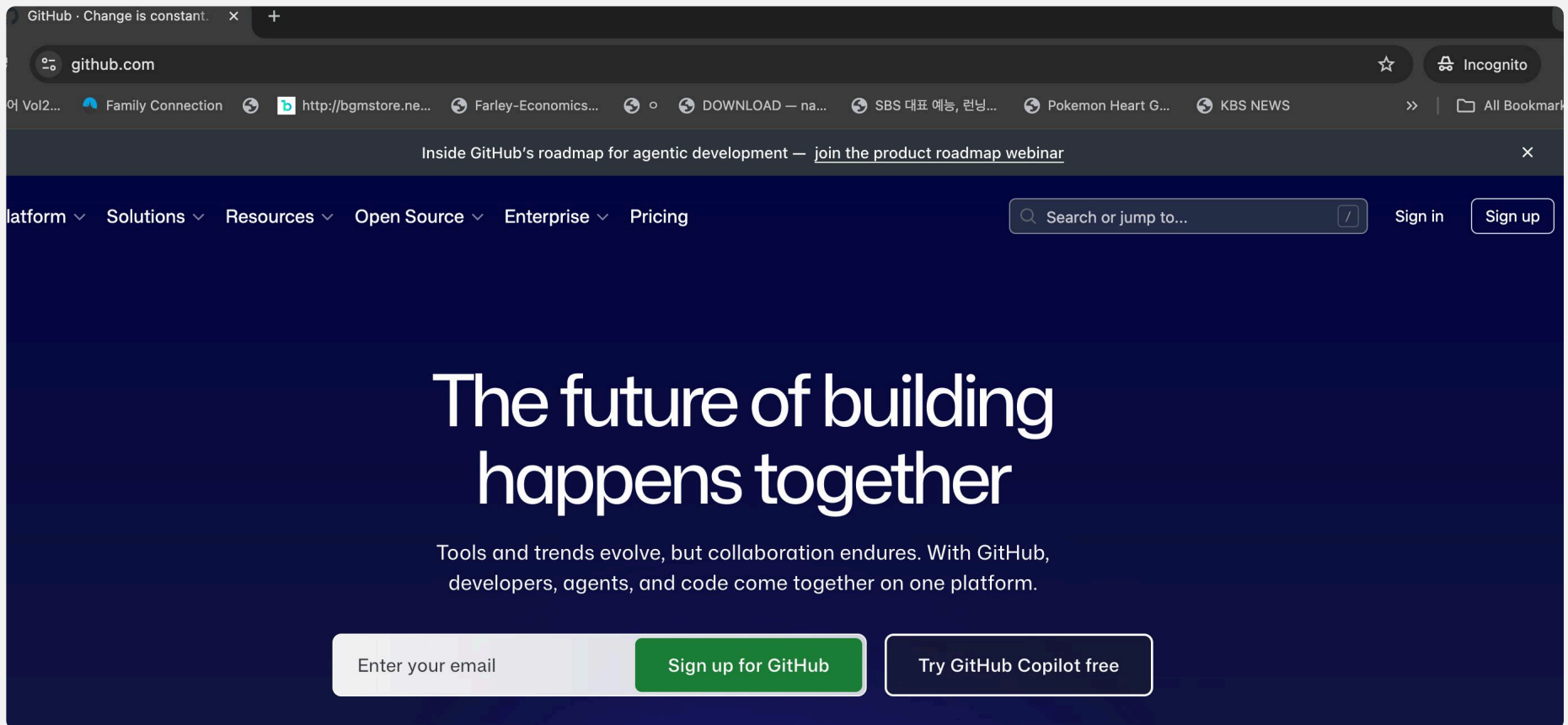
> GitHub 연결은 "내 코드를 안전하게 보관할 장소"를 만드는 과정입니다

GitHub은 코드를 인터넷에 안전하게 저장할 수 있는 공간으로, 구글 드라이브의 코드 버전이라고 생각하시면 됩니다.



> GitHub 계정이 없다면 1분이면 만들 수 있습니다

<https://github.com>에 접속하여 'Sign up'을 클릭한 후 이메일과 비밀번호를 입력하면 가입이 완료됩니다. 이미 계정이 있으신 경우에는 바로 다음 단계로 진행해 주시기 바랍니다.



› Claude가 GitHub에 접속하려면 "나만의 열쇠(토큰)"가 필요합니다

본인이 직접 비밀 열쇠를 생성하면, Claude가 해당 열쇠를 사용해 GitHub에 대신 접속하는 구조입니다.

지금부터 총 7단계에 걸쳐 토큰을 생성해 보겠습니다. 화면에 보이는 내용을 그대로 따라와 주세요.

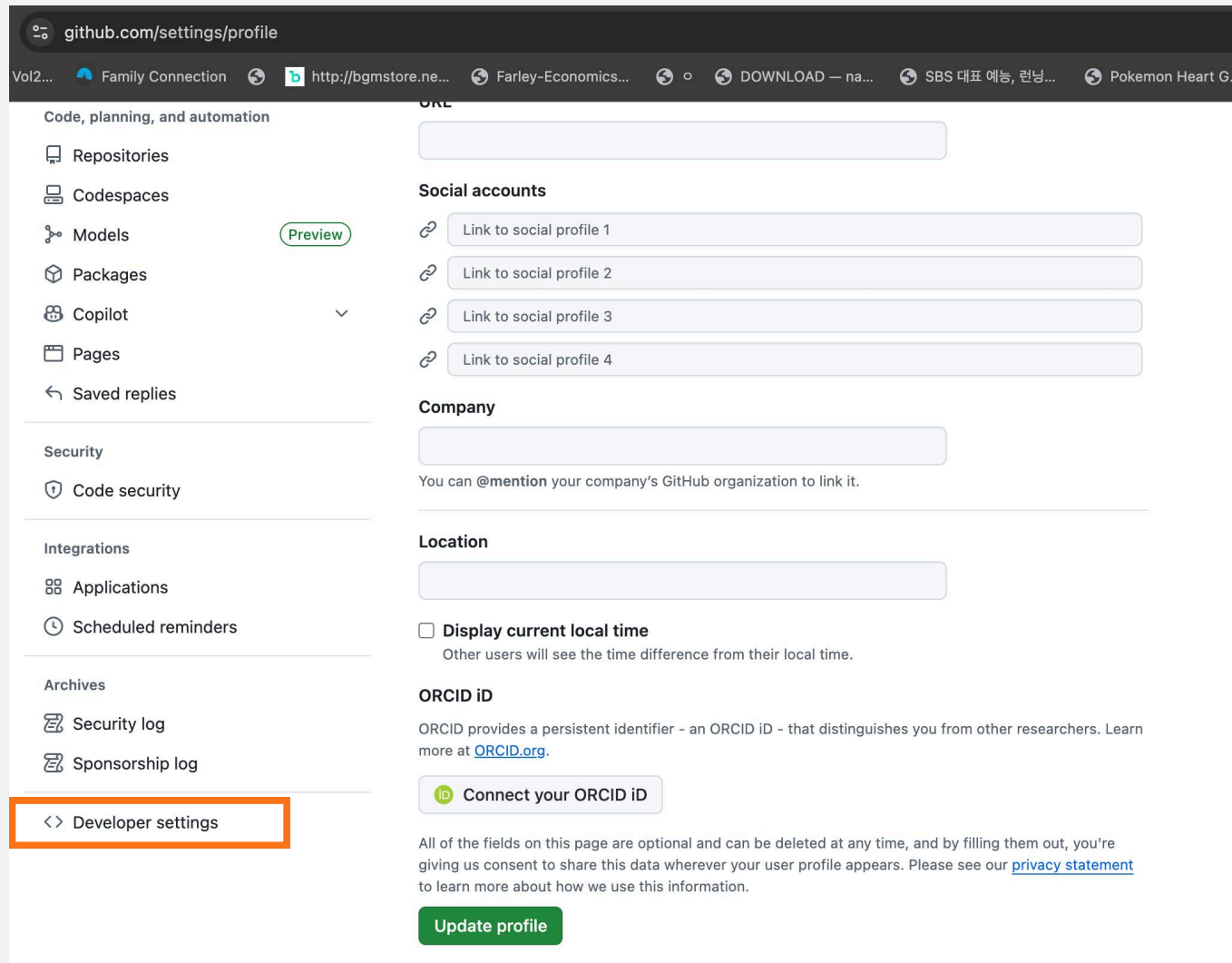
› 프로필 아이콘을 클릭하신 후, Settings 메뉴로 이동해 주세요.

오른쪽 위 프로필 아이콘을 클릭하신 후 [Settings]를 선택해 주세요.

The screenshot shows the GitHub Dashboard interface. On the left, there's a sidebar with the text "Create your first project" and buttons for "Create repository" and "Import repository". The main content area is titled "Home" and features a search bar, navigation buttons like "Agent", "Create issue", "Write code", "Git", and "Pull requests", and a video player for "GitHub for beginners on YouTube". On the right, a user profile dropdown menu is open for "innovatewithjki...". The "Settings" option in this menu is highlighted with an orange border. Other options include "Set status", "Profile", "Repositories", "Stars", "Gists", "Organizations", "Enterprises", "Sponsors", "Copilot settings", "Feature preview", "Appearance", "Accessibility", "Try Enterprise", and "Sign out".

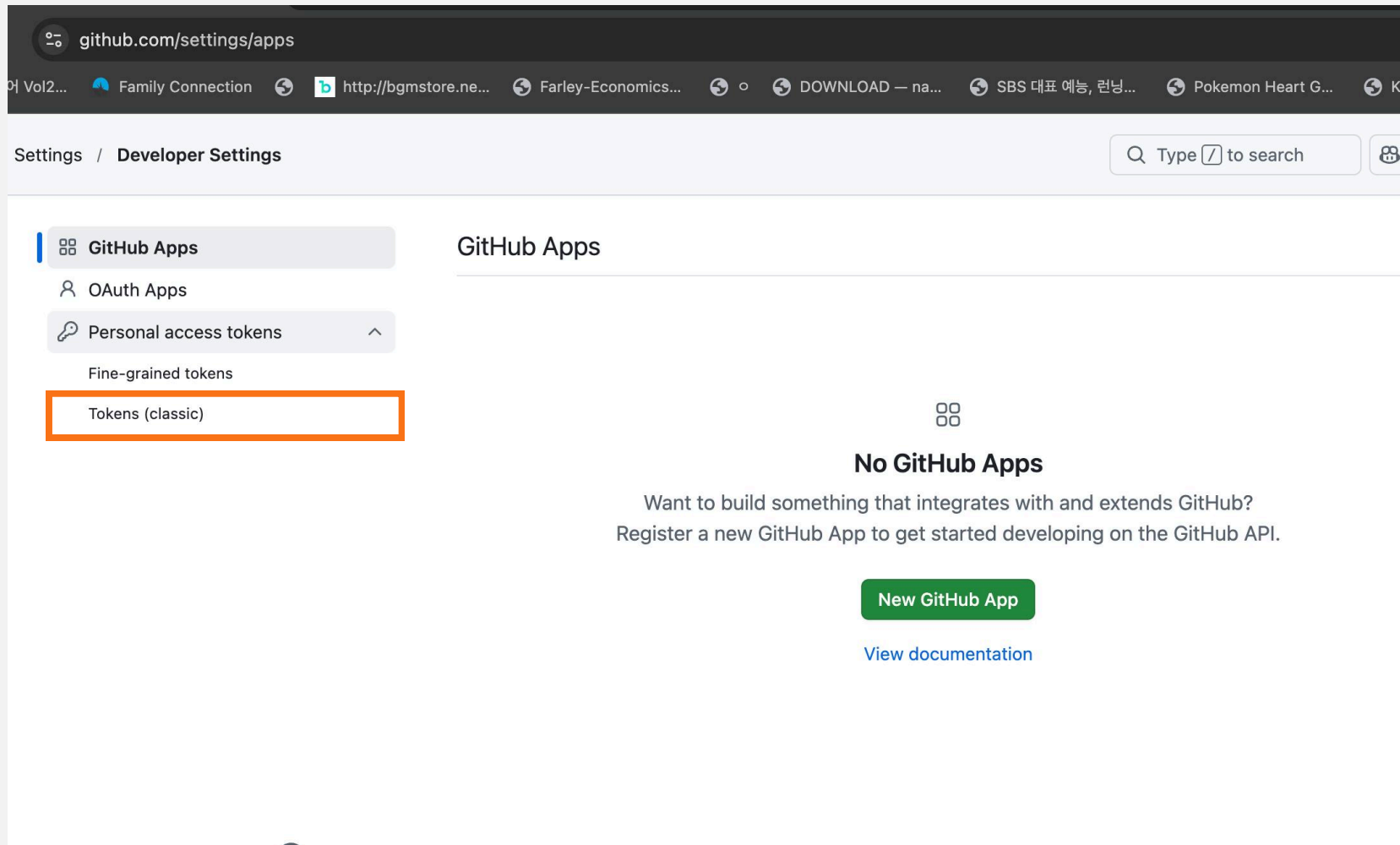
› 왼쪽 메뉴 맨 아래에 있는 'Developer settings'를 클릭해 주세요.

메뉴가 길어서 찾기 어려우실 수 있습니다. 화면 맨 아래까지 스크롤하시면 확인하실 수 있습니다.



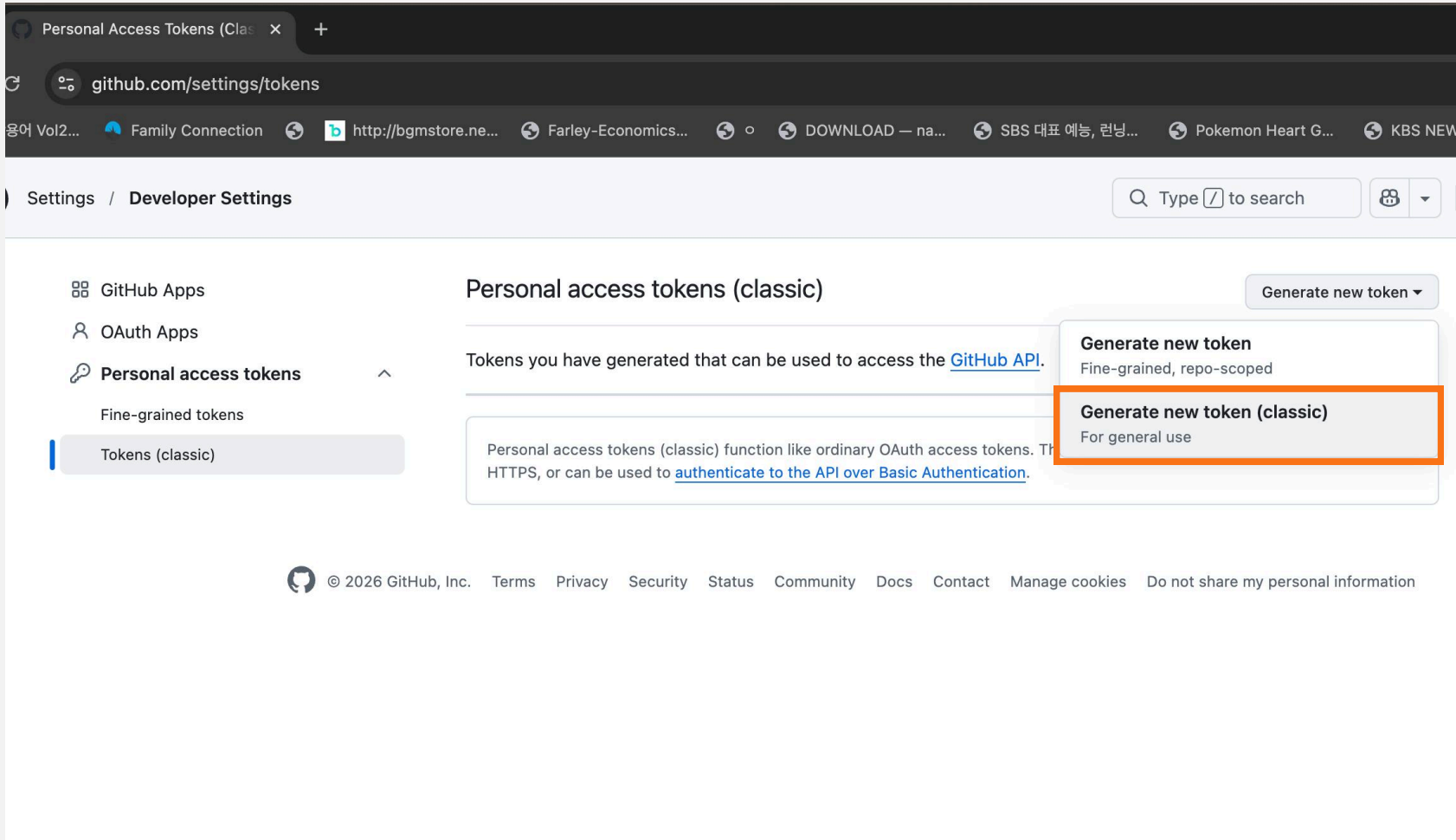
The screenshot shows the GitHub profile settings page at `github.com/settings/profile`. The left sidebar contains a list of settings categories. The 'Developer settings' option at the bottom of the sidebar is highlighted with an orange box. The main content area shows various settings sections: 'Code, planning, and automation' (Repositories, Codespaces, Models, Packages, Copilot, Pages, Saved replies), 'Security' (Code security), 'Integrations' (Applications, Scheduled reminders), 'Archives' (Security log, Sponsorship log), 'Social accounts' (four 'Link to social profile' buttons), 'Company' (text input field), 'Location' (text input field and 'Display current local time' checkbox), and 'ORCID iD' (text input field and 'Connect your ORCID iD' button). At the bottom, there is an 'Update profile' button and a privacy statement link.

- **Personal access tokens 메뉴에서 Tokens (classic)을 클릭해 주세요.**
메뉴가 길어서 찾기 어려우실 수 있습니다. 화면 맨 아래까지 스크롤하시면 확인하실 수 있습니다.



› Generate new token (classic) 버튼을 클릭해 주세요.

메뉴가 길어서 찾기 어려우실 수 있습니다. 화면 맨 아래까지 스크롤하시면 확인하실 수 있습니다.



The screenshot shows the GitHub 'Personal access tokens (classic)' settings page. The page title is 'Personal access tokens (classic)' and it includes a 'Generate new token' dropdown menu. The dropdown menu is open, showing two options: 'Generate new token' (Fine-grained, repo-scoped) and 'Generate new token (classic)' (For general use). The 'Generate new token (classic)' option is highlighted with an orange border. The page also features a search bar, a sidebar with navigation links (GitHub Apps, OAuth Apps, Personal access tokens), and a footer with copyright information and links to Terms, Privacy, Security, Status, Community, Docs, Contact, Manage cookies, and Do not share my personal information.

› 이름은 "claude-code"로, 유효기간은 90일로 설정해 주시기 바랍니다.

claude-code 라고 입력하세요 (다른 이름도 괜찮아요!). Expiration은 90 days를 선택해 주시면 됩니다.

New personal access token (classic)

Personal access tokens (classic) function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to [authenticate to the API over Basic Authentication](#).

Note

What's this token for?

Expiration

The token will expire on the selected date

➤ 권한은 **repo**, **workflow**, **read:org** 이렇게 정확히 3개만 체크해 주세요.

이번 설정 시 '**repo**'는 코드 업로드 및 수정 권한을 위해, '**workflow**'는 자동 배포 권한을 위해 필요합니다. '**read:org**'는 팀 확인을 위한 권한이며, 이 외의 다른 항목들은 절대 변경하지 않으셔도 됩니다.

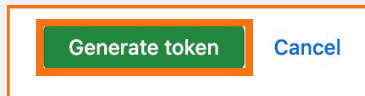
Select scopes

Scopes define the access for personal tokens. [Read more about OAuth scopes.](#)

<input checked="" type="checkbox"/> repo	Full control of private repositories
<input checked="" type="checkbox"/> repo:status	Access commit status
<input checked="" type="checkbox"/> repo_deployment	Access deployment status
<input checked="" type="checkbox"/> public_repo	Access public repositories
<input checked="" type="checkbox"/> repo:invite	Access repository invitations
<input checked="" type="checkbox"/> security_events	Read and write security events
<input checked="" type="checkbox"/> workflow	Update GitHub Action workflows
<input type="checkbox"/> write:packages	Upload packages to GitHub Package Registry
<input type="checkbox"/> read:packages	Download packages from GitHub Package Registry
<input type="checkbox"/> delete:packages	Delete packages from GitHub Package Registry
<input type="checkbox"/> admin:org	Full control of orgs and teams, read and write org projects
<input type="checkbox"/> write:org	Read and write org and team membership, read and write org projects
<input checked="" type="checkbox"/> read:org	Read org and team membership, read org projects
<input type="checkbox"/> manage_runners:org	Manage org runners and runner groups

➤ **Generate token** 클릭 후, 나온 토큰을 즉시 복사해 주세요.

ghp_로 시작하는 긴 문자열이 나올 예정입니다. 복사 버튼을 눌러서 메모장에 바로 붙여넣어 저장해 주세요.



Personal access tokens (classic) Generate new token ▾

Tokens you have generated that can be used to access the [GitHub API](#).

Make sure to copy your personal access token now. You won't be able to see it again!

✓ ghp_Vqd4bYfmbk1		Delete
-------------------	--	--------


Personal access tokens (classic) function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to [authenticate to the API over Basic Authentication](#).

› 클로드코드에 아래 내용을 입력해주세요.

GitHub CLI(gh)를 제 컴퓨터에 설치해주세요.

먼저 제 운영체제를 확인하고, 필요한 사전 도구(macOS면 brew, Windows면 winget, 그리고 git)가 깔려 있는지 점검해서 빠진 게 있으면 함께 설치해주세요. 비밀번호나 관리자 권한이 필요한 단계는 멈추고 어떻게 해야 하는지 알려주세요. 모든 과정은 한국어로 설명하고, 에러가 나면 멈추지 말고 해결 방법을 찾아주세요.

마지막에 'gh --version'이 정상 출력되는지 확인해주세요.

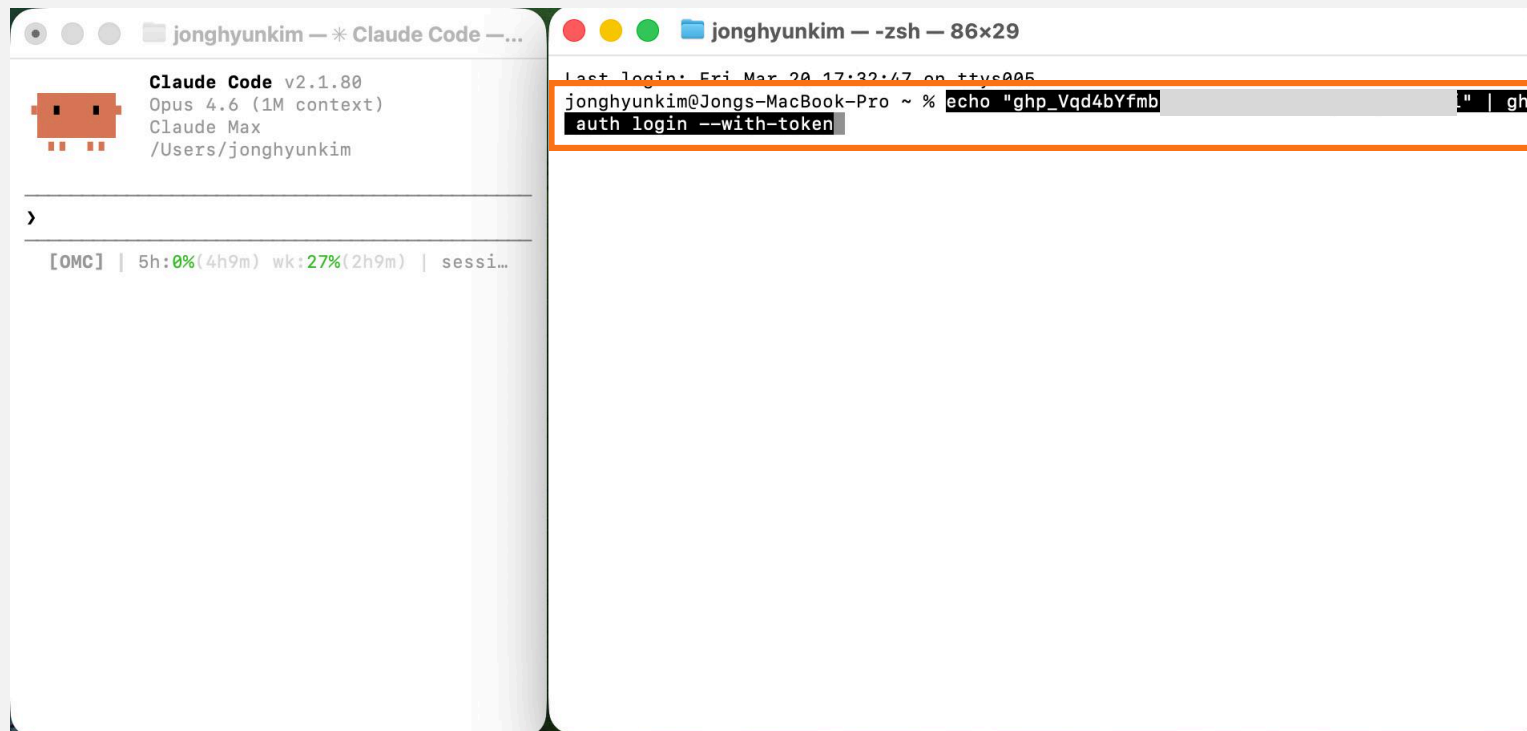
```
henrychen@macbook-pro workspace %  
 Claude Code v2.1.126  
Opus 4.7 (1M context) with xhigh effort · Claude Max  
~/Desktop/workspace  
  
› GitHub CLI(gh)를 제 컴퓨터에 설치해주세요.  
  
먼저 제 운영체제를 확인하고, 필요한 사전 도구(macOS면 brew,  
Windows면 winget, 그리고 git)가 깔려 있는지 점검해서 빠진 게  
있으면 함께 설치해주세요. 비밀번호나 관리자 권한이 필요한 단계는  
멈추고 어떻게 해야 하는지 알려주세요. 모든 과정은 한국어로 설명하고,  
에러가 나면 멈추지 말고 해결 방법을 찾아주세요.  
  
마지막에 'gh --version'이 정상 출력되는지 확인해주세요.█
```

➤ 새 터미널 창을 여신 후, 토큰을 사용하여 GitHub에 연결해 주시기 바랍니다.

현재 Claude Code가 실행 중인 화면은 그대로 두시고, 터미널을 새로 하나 더 열어주세요. Mac 사용자는 Cmd + N을 누르시면 되고, Windows 사용자는 PowerShell을 다시 검색하여 실행하시면 됩니다.

그다음, 아래 명령어에서 "복사한_토큰" 부분을 미리 메모장에 저장해둔 토큰 값으로 교체하여 입력해 주세요.
예시: `echo "ghp_abc123..." | gh auth login --with-token`

```
echo "복사한_토큰" | gh auth login --with-token
```



연결이 안 되는 경우, 다음의 두 가지 원인과 해결 방법을 확인해야 합니다.

gh: command not found 오류가 발생하는 경우, Claude Code 화면으로 돌아가서 "gh auth login 해줘"라고 입력해 보시기 바랍니다. 그러면 Claude가 로그인을 대신 진행해 줄 것입니다.

만약 bad credentials 에러가 발생한다면, 토큰을 붙여넣는 과정에서 오류가 있었을 수 있습니다. 따옴표(") 안에 토큰만 정확히 입력되었는지 다시 한번 확인해 주세요.

연결 상태를 확인하고 싶을 때는 Claude Code에서 "gh auth status 확인해줘"라고 입력하시면 됩니다. 결과에 본인의 계정 이름이 나타나면 정상적으로 연결된 것입니다.

```

Claude Code v2.1.80
Opus 4.6 (1M context) · Claude Max
/Users/jonghyunkim

/clear
└ (no content)

gh auth status 확인해줘

● Bash(gh auth status)
└ Running...

Bash command

gh auth status
Check GitHub CLI auth status

This command requires approval

Do you want to proceed?
1. Yes
2. Yes, and don't ask again for: gh auth:*
3. No

Esc to cancel · Tab to amend · ctrl+e to explain

```

```

gh auth status 확인해줘

● Bash(gh auth status)
└ github.com
  └ Logged in to github.com account innovatewithjkim-code (keyring)
    - Active account: true
    ... +9 lines (ctrl+o to expand)

GitHub CLI 인증 상태입니다 :

활성 계정 : innovatewithjkim-code (현재 사용 중)
- 프로토콜 : HTTPS
- 토큰 권한 : read:org, repo, workflow

비활성 계정 : jonghyun5995
- 프로토콜 : HTTPS
- 토큰 권한 : read:org, repo, workflow

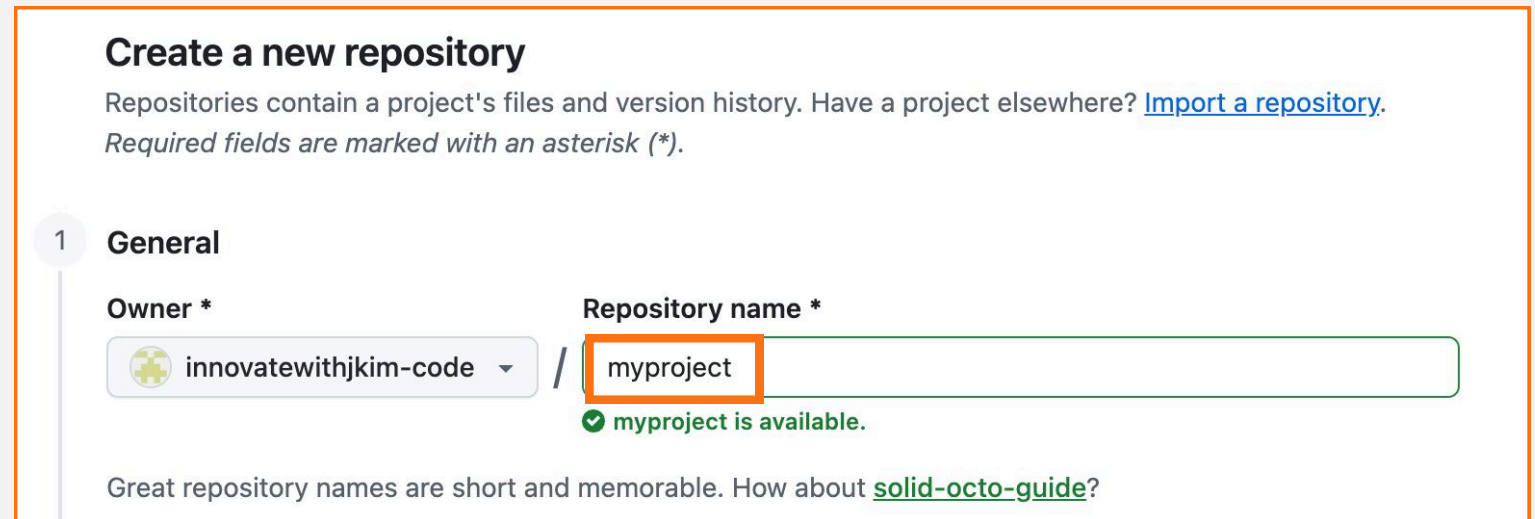
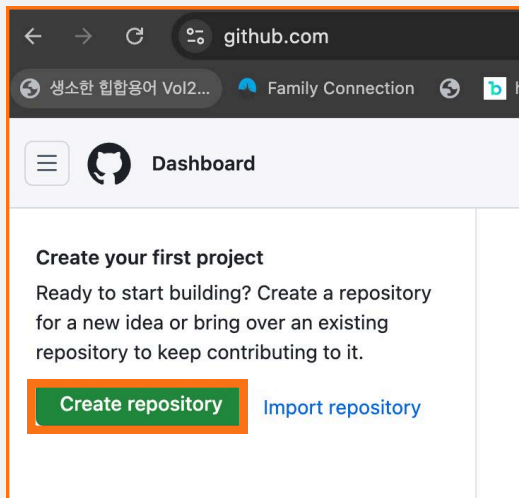
두 계정 모두 정상적으로 로그인되어 있고, 현재 innovatewithjkim-code 계정이 활성화되어 있습니다.

[OMC] | 5h:11%(2h1m) wk:28%(0h1m) | thinking | session:0m | ~$0.0369 | 21.3k | ...

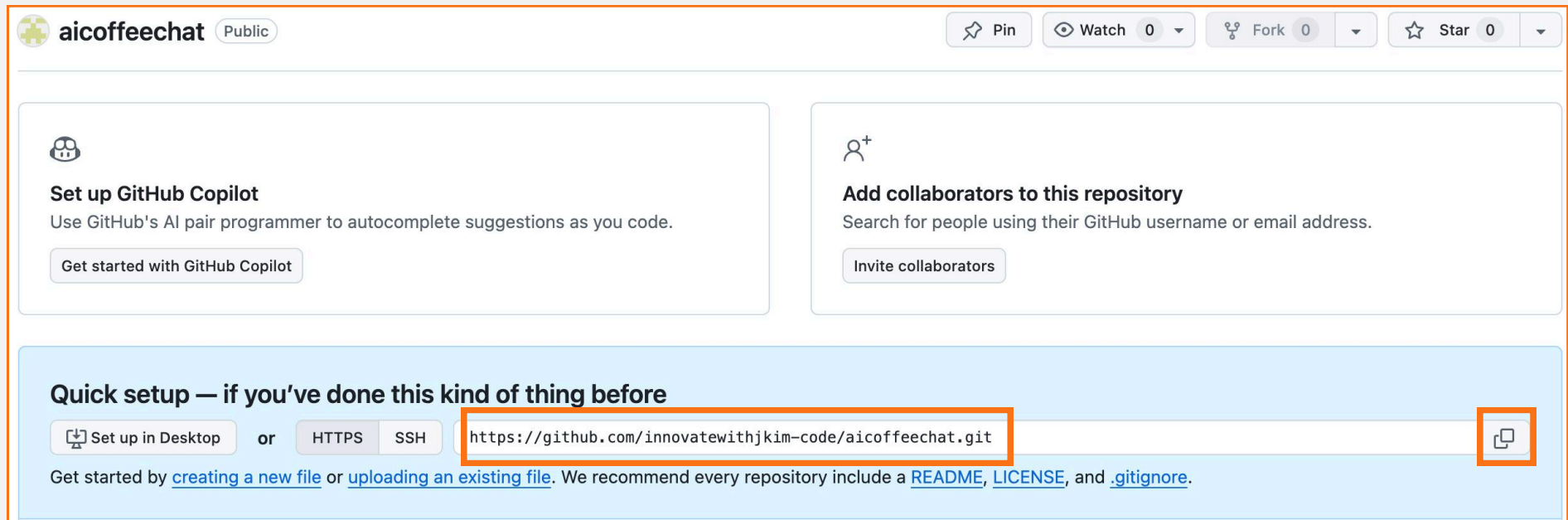
```

› 코드를 저장할 저장소를 생성해 주세요. 클릭 세 번이면 간편하게 완료하실 수 있습니다.

<https://github.com>에 접속하신 후, 초록색 'New' 버튼을 클릭해 주세요. 'Repository name'에 원하시는 이름(예: my-project)을 입력하고 'Create repository' 버튼을 클릭하시면 완료됩니다.



- **생성된 저장소 주소를 복사해 주세요. 해당 주소는 다음 단계에서 바로 사용될 예정입니다.**
생성된 저장소 주소를 복사해 주세요. 해당 주소는 다음 단계에서 바로 사용될 예정입니다.



The screenshot shows the GitHub repository page for 'aicoffeechat'. At the top, there are buttons for 'Pin', 'Watch 0', 'Fork 0', and 'Star 0'. Below this, there are two main sections: 'Set up GitHub Copilot' and 'Add collaborators to this repository'. The 'Set up GitHub Copilot' section includes a 'Get started with GitHub Copilot' button. The 'Add collaborators to this repository' section includes an 'Invite collaborators' button. At the bottom, there is a 'Quick setup — if you've done this kind of thing before' section. This section has three tabs: 'Set up in Desktop', 'HTTPS', and 'SSH'. The 'HTTPS' tab is selected, and the URL 'https://github.com/innovatewithjkim-code/aicoffeechat.git' is displayed in a text input field, which is highlighted with a red box. To the right of the text input is a copy icon, also highlighted with a red box. Below the text input, there is a note: 'Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).'

> Claude Code에 아래 문장을 붙여넣으시면 자동으로 업로드됩니다.

아래에서 [복사한 주소] 부분을 방금 복사한 주소로 바꿔주세요!

주황색 네모 박스에 있는 내용을 복사해서 Claude Code에 그대로 붙여넣으세요.

방금 만든 GitHub 저장소에 현재 폴더를 연결하고 올려줘. 주소는 [복사한 주소]야. 중간에 멈추지 말고 끝까지 완료해줘

Claude가 알아서 처리해줄 거예요. 잠시만 기다려주세요. (나오는 질문에 모두 "Yes"로 답해주시면 됩니다!)

```
jonghyunkim@Jongs-MacBook-Pro myproject % claude
Claude Code v2.1.80

Welcome back Jh!



Opus 4.6 (1M context) · Claude Max ·
jhkim5995@gmail.com's Organization
~/Desktop/myproject

Tips for getting started
Ask Claude to create a new app or clone a repository

Recent activity
No recent activity

방금 만든 GitHub 저장소에 현재 폴더를 연결하고 올려줘. 주소는 [https://github.com/innovatewithjkim-code/aicoffeechat.git]야. 중간에
멈추지 말고 끝까지 완료해줘

현재 폴더 상태를 먼저 확인하겠습니다.

• Bash(git status 2>&1 || echo "NOT_A_GIT_REPO")
└ fatal: not a git repository (or any of the parent directories): .git
  NOT_A_GIT_REPO

Read 1 file (ctrl+o to expand)

빈 폴더이고 git 저장소가 아직 없네요. 초기화하고 원격 저장소에 연결하겠습니다.

• Bash(git init && git remote add origin https://github.com/innovatewithjkim-code/aicoffeechat.git)
└ Running...
```

› "push 완료" 메시지가 확인되면 GitHub 연결이 정상적으로 완료된 거 입니다.

GitHub 저장소 페이지를 새로고침해 보시면 업로드된 파일들을 확인하실 수 있습니다.

완료되었습니다! 정리하면 :

1. `git init` - 로컬 Git 저장소 초기화
2. `git remote add origin` - GitHub 원격 저장소 연결
3. `README.md` 생성 후 초기 커밋
4. `main` 브랜치로 푸시 완료

저장소 주소 : `https://github.com/innovatewithjkim-code/aicoffeechat.git`

› ■