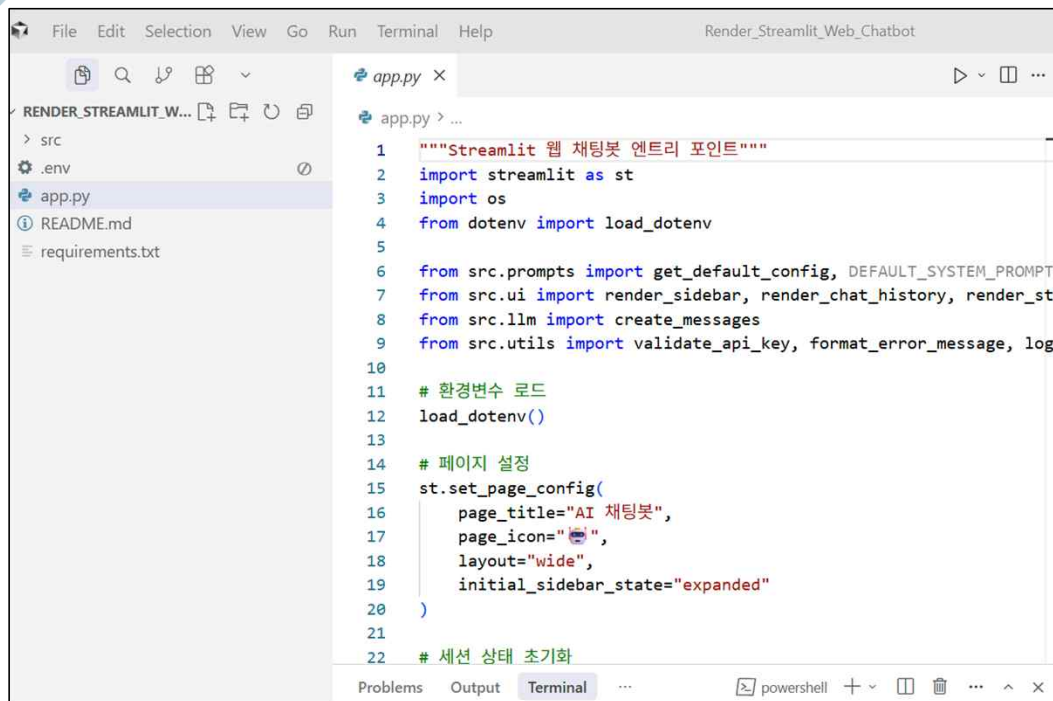


Render로 Streamlit Web Chatbot 배포하기

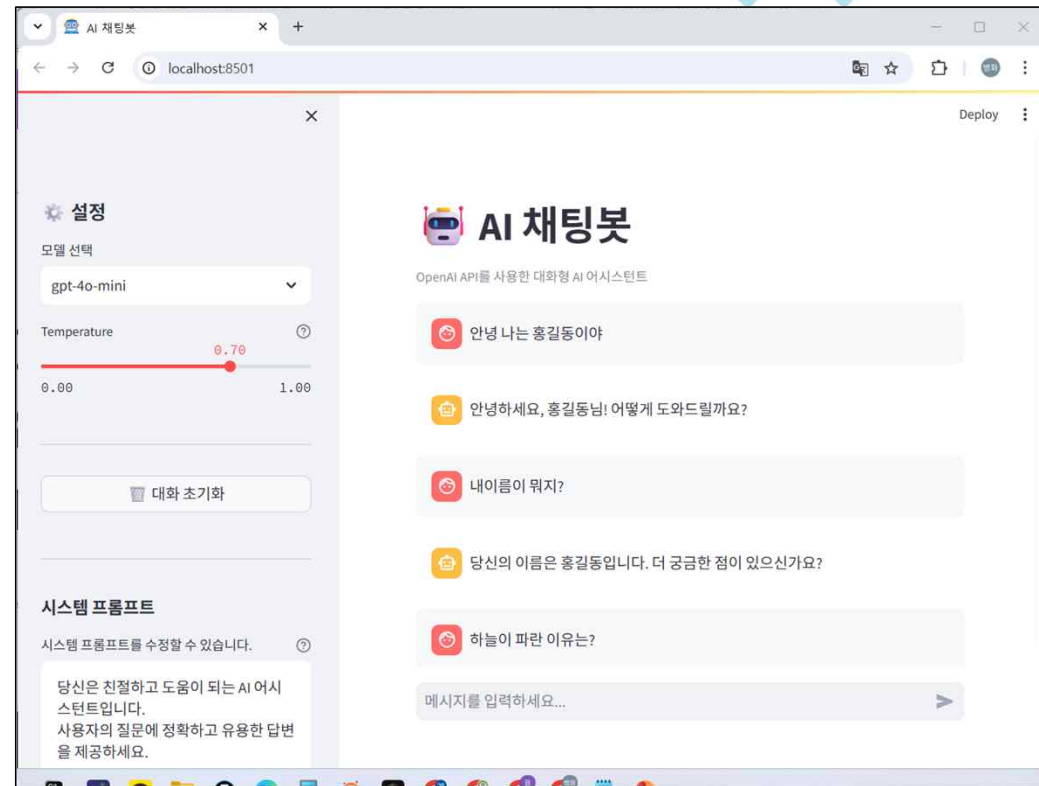


1. Streamlit 웹 챗봇 소스 준비

Streamlit을 사용한 웹 챗봇 소스를 복사해와서 Render_Streamlit_Web_Chatbot 폴더로 이름을 변경하고 커서를 사용하여 실행시키고 동작을 확인한다



```
1 """Streamlit 웹 채팅봇 엔트리 포인트"""
2 import streamlit as st
3 import os
4 from dotenv import load_dotenv
5
6 from src.prompts import get_default_config, DEFAULT_SYSTEM_PROMPT
7 from src.ui import render_sidebar, render_chat_history, render_st
8 from src.llm import create_messages
9 from src.utils import validate_api_key, format_error_message, log
10
11 # 환경변수 로드
12 load_dotenv()
13
14 # 페이지 설정
15 st.set_page_config(
16     page_title="AI 채팅봇",
17     page_icon="🤖",
18     layout="wide",
19     initial_sidebar_state="expanded"
20 )
21
22 # 세션 상태 초기화
```



- **Requirements.txt** 파일 수정

python-dotenv는 .env를 사용하지 않으므로 필요 없으므로 삭제한다(주석 처리)

streamlit

openai

python-dotenv

- **app.py** 소스에 가서 **dotenv** 사용 부분(파란색 2줄)을 주석 처리하고 저장한다

"""Streamlit 웹 채팅봇 엔트리 포인트"""

import streamlit as st

import os

from dotenv import load_dotenv

from src.prompts import get_default_config, DEFAULT_SYSTEM_PROMPT, DEFAULT_MODEL,
DEFAULT_TEMPERATURE

from src.ui import render_sidebar, render_chat_history, render_streaming_response

from src.llm import create_messages

from src.utils import validate_api_key, format_error_message, logger

환경변수 로드

load_dotenv()

- src/lmm.py 소스에 가서 dotenv 사용 부분(파란색 2줄)을 주석 처리하고 저장한다

```
"""OpenAI API 호출 및 스트리밍 처리 모듈"""  
import os  
from typing import List, Dict, Optional, Iterator  
from openai import OpenAI  
# from dotenv import load_dotenv  
import streamlit as st
```

```
from src.utils import format_error_message, validate_api_key, logger
```

```
# 환경변수 로드  
# load_dotenv()
```

.gitignore 파일을 만들고 아래 내용을 넣어 커밋 시 무시해야 할 파일을 지정한다.

```
# Python
__pycache__/
*.pyc
*.pyo
*.pyd

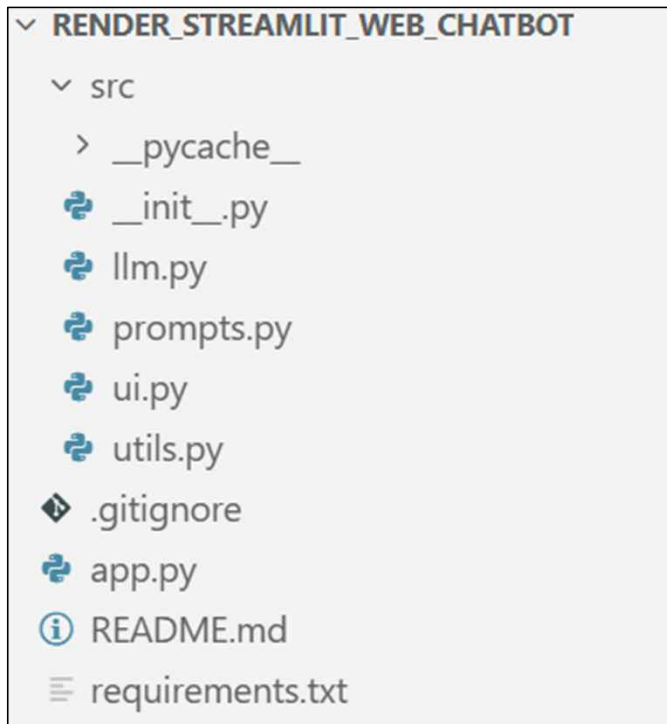
# Virtual env
.venv/
venv/

# OS / IDE
.DS_Store
.vscode/
.idea/

# Logs
*.log
```

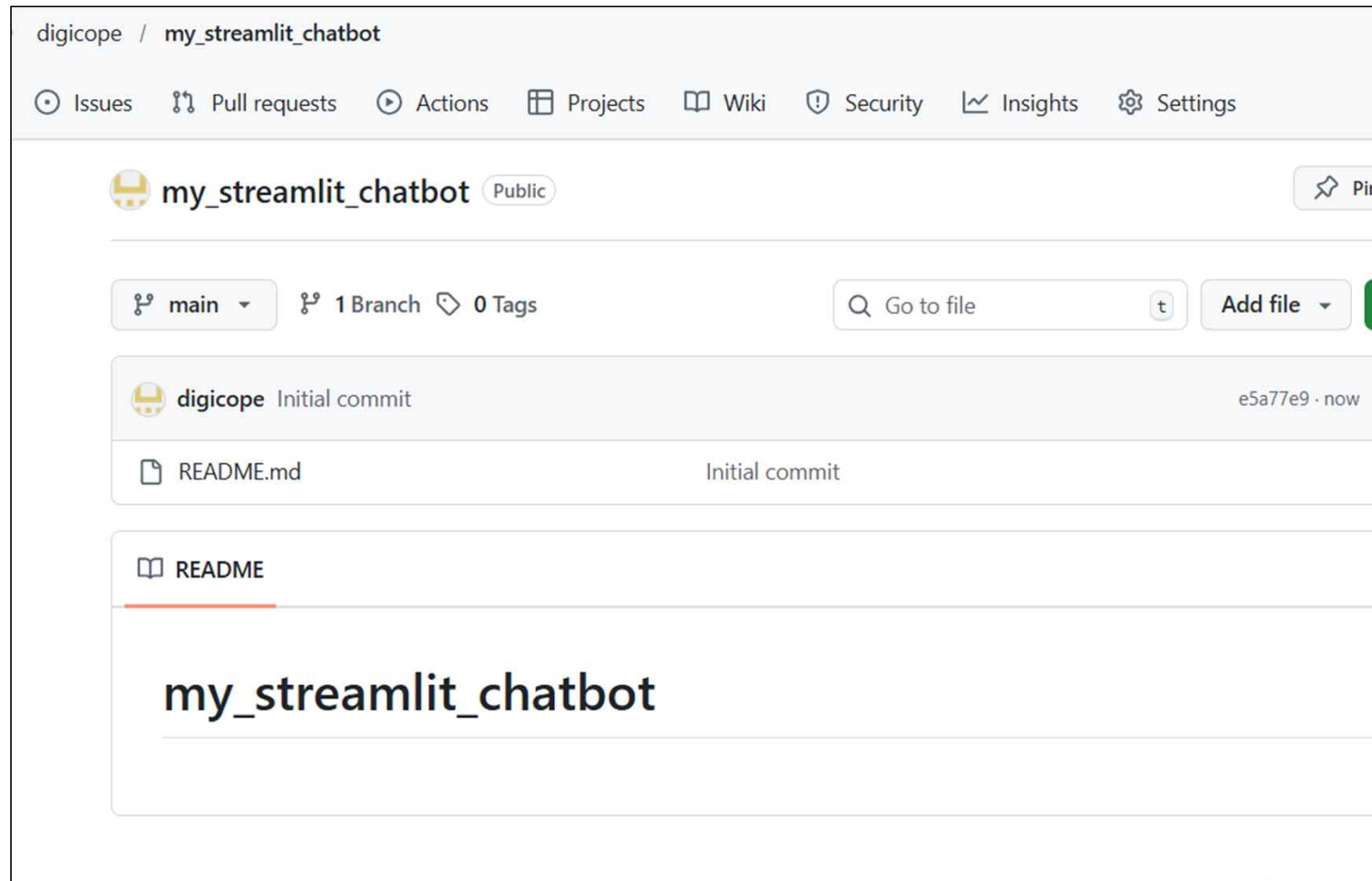
- .env 파일은 Git 저장소에 올리면 보안상 위험 하므로 삭제한다
Render에서 수동으로 환경변수에 OpenAI API 키 값을 지정해야 하므로 다른 곳에 사본
을 복사해놓고 프로젝트내에서 삭제한다

변경된 프로젝트 내의 소스 파일 구조

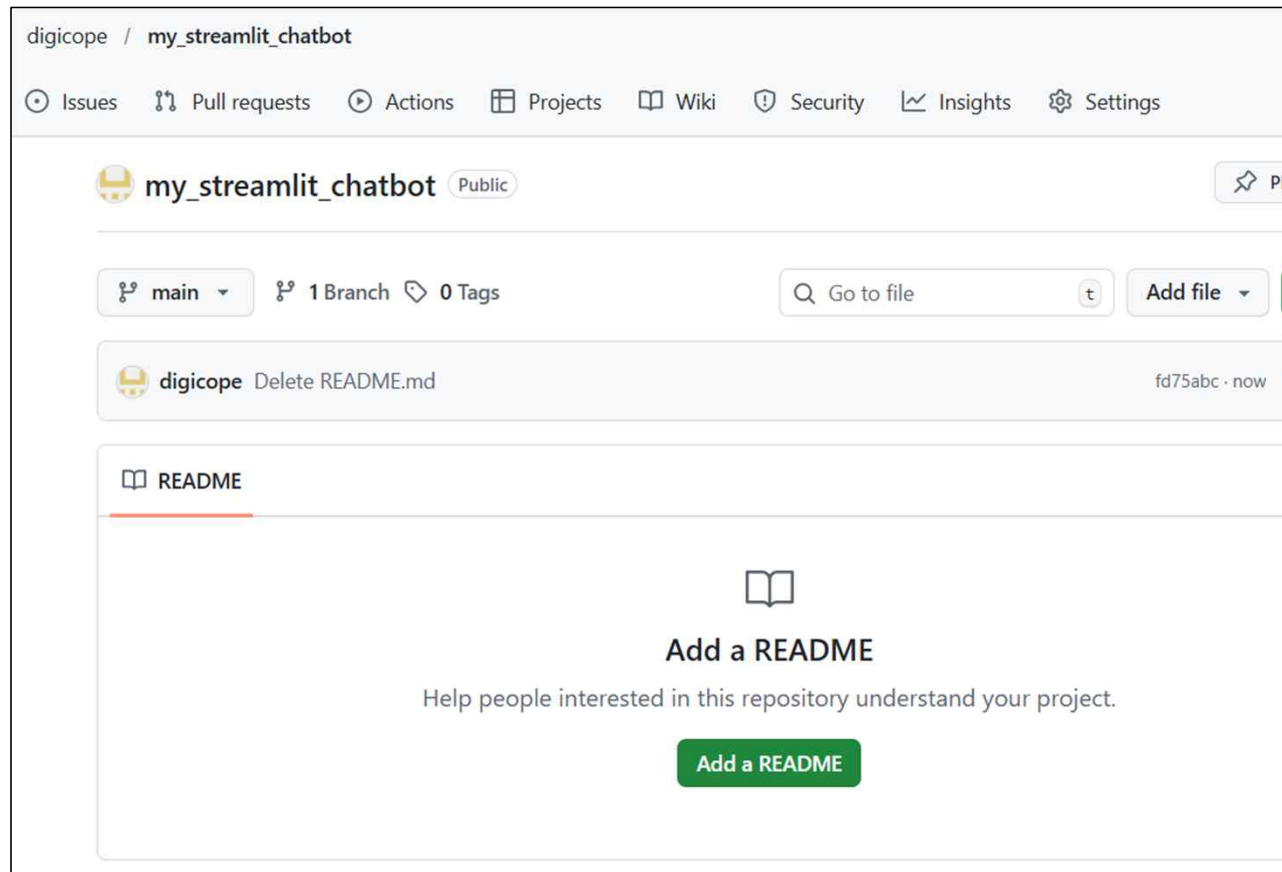


2. Git 저장소에 배포 소스 업로드

- Git에 새로운 저장소를 **my_streamlit_chatbot** 이름으로 생성해 놓는다












- Git 저장소에 파일이 있으면 저장소로 가서 파일을 모두 삭제해 놓는다
(README.md 파일 반드시 삭제)



- 윈도우 탐색기에서 test_git 폴더 아래 배포할 앱을 복사해 놓는다
(강사 배포 파일 사용시 : Render_Streamlit_Web_Chatbot.zip)

실제 경로: C:\Users\storm\test_git\Render_Streamlit_Web_Chatbot

> test_git > Render_Streamlit_Web_Chatbot >				
  정렬  보기  ...				
<input type="checkbox"/>	이름	수정한 날짜	유형	크기
	src	2026-01-26 오후 2:40	파일 폴더	
	.gitignore	2026-01-25 오후 2:47	txtfile	1KB
	app.py	2026-01-26 오전 10:12	PY 파일	4KB
	README.md	2026-01-26 오전 10:12	MD 파일	5KB
	requirements.txt	2026-01-26 오후 2:29	텍스트 문서	1KB

- Windows에서 Git Bash를 실행하고 아래 경로로 이동한다

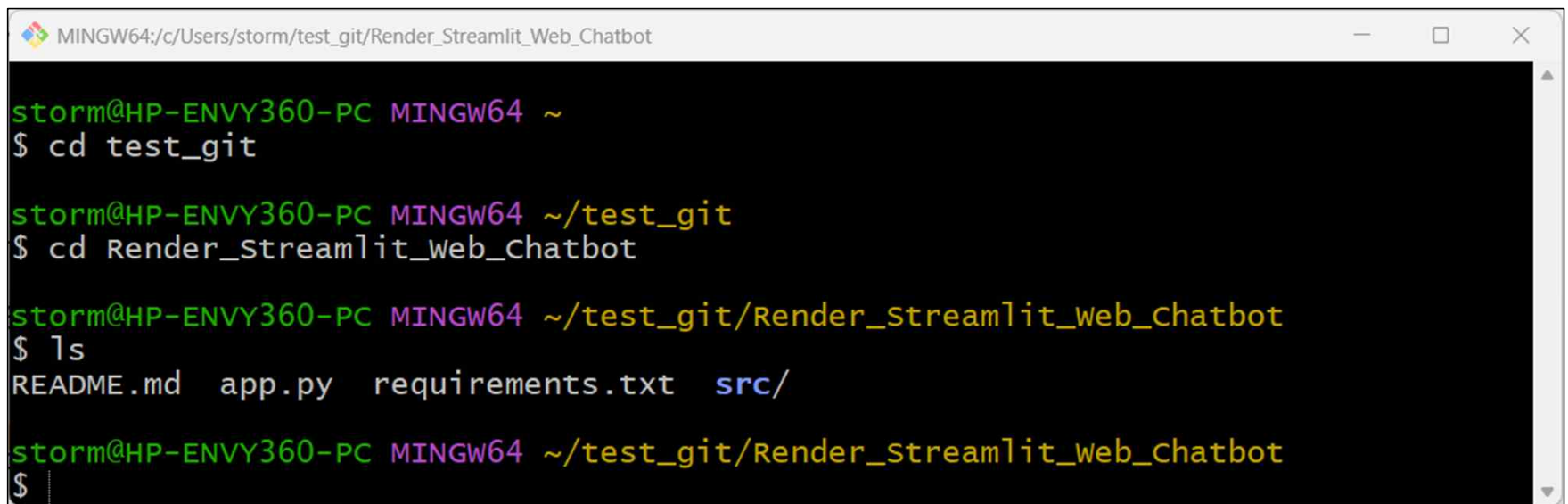
cd test_git

- 소스 경로로 이동한다

cd Render_Streamlit_Web_Chatbot

- 파일을 확인해 본다

ls



```
MINGW64:/c:/Users/storm/test_git/Render_Streamlit_Web_Chatbot

storm@HP-ENVY360-PC MINGW64 ~
$ cd test_git

storm@HP-ENVY360-PC MINGW64 ~/test_git
$ cd Render_Streamlit_Web_Chatbot

storm@HP-ENVY360-PC MINGW64 ~/test_git/Render_Streamlit_Web_Chatbot
$ ls
README.md  app.py  requirements.txt  src/

storm@HP-ENVY360-PC MINGW64 ~/test_git/Render_Streamlit_Web_Chatbot
$
```

Git 저장소에 소스 파일 업로드

아래 명령을 차례로 수행한다

(**user name**과 **email**, **저장소 경로**는 본인의 이름으로 모두 수정한 다음 실행한다)

```
git init
```

```
git config --global user.name digicope
```

```
git config --global user.email digicope@aicore.co.kr
```

```
git add .
```

```
git commit -m "initial streamlit chatbot for render"
```


```
git branch -M main
```

```
git remote add origin https://github.com/digicope/my_streamlit_chatbot.git
```


```
git pull origin main --rebase
```

```
git push -u origin main
```






Git 저장소에서 업로드 된 파일들을 확인한다

 **my_streamlit_chatbot** PublicPinWatch 0

main 1 Branch 0 Tags t Add file <> Code

 **digicope**

initial streamlit chatbot for render aa357da · now 3 Commits

 src	initial streamlit chatbot for render	now
 .gitignore	initial streamlit chatbot for render	now
 README.md	initial streamlit chatbot for render	now
 app.py	initial streamlit chatbot for render	now
 requirements.txt	initial streamlit chatbot for render	now

README✎☰

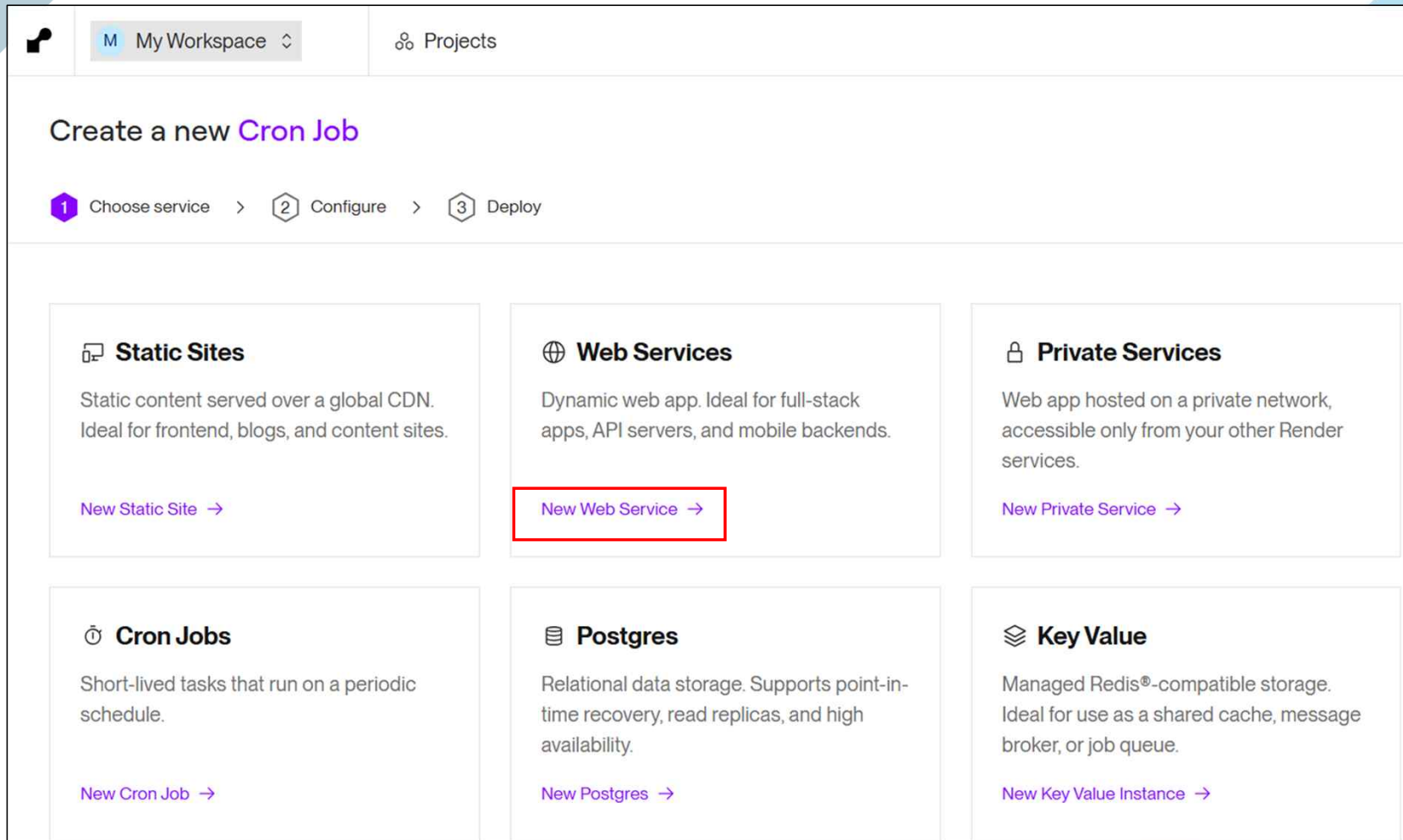
Streamlit 웹 채팅봇

OpenAI API를 사용한 대화형 AI 채팅봇 웹 애플리케이션입니다.

3. Render 에서 서비스 배포하기



Render 홈페이지 대시보드로 가서 **Web Services**의 **New Web Service** 를 누른다

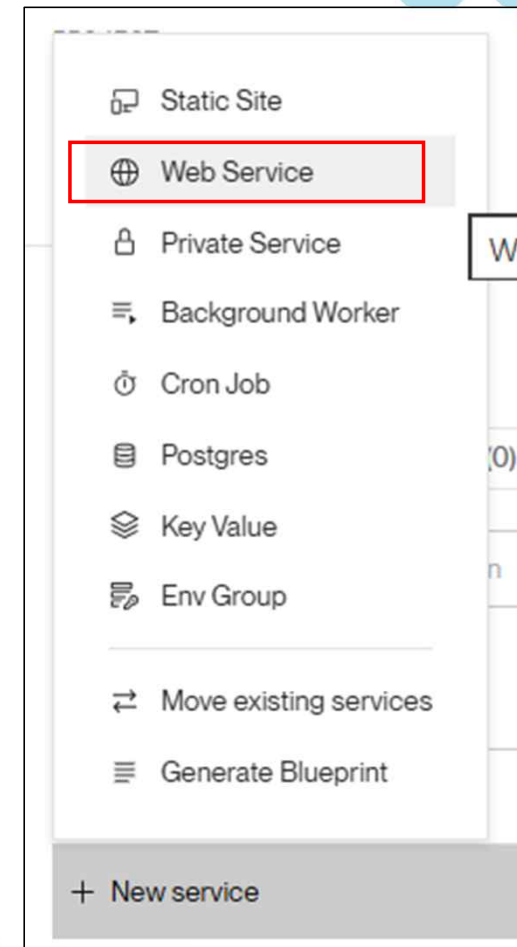
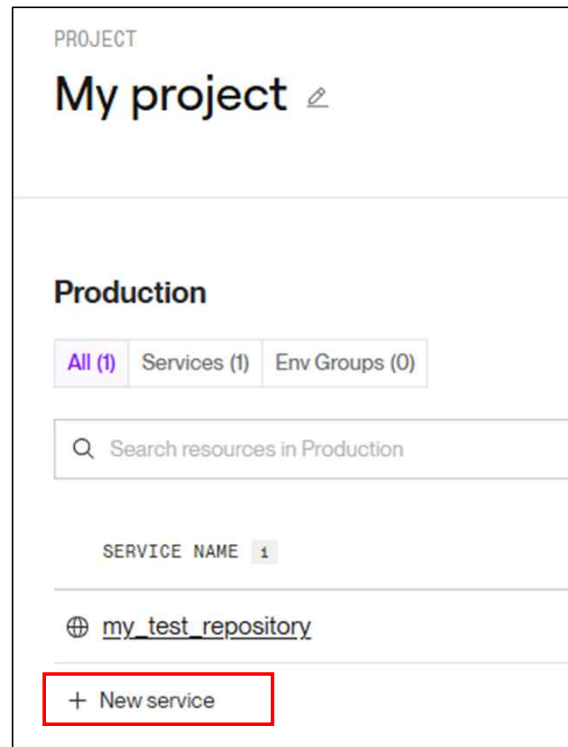
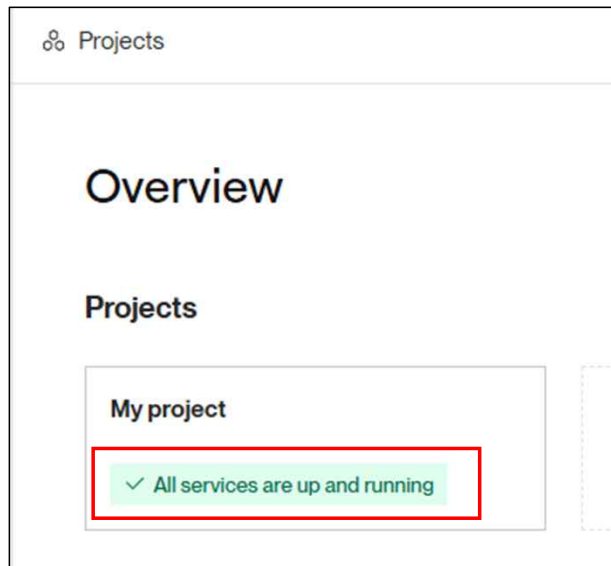


The screenshot shows the Render dashboard interface. At the top, there's a navigation bar with 'My Workspace' and 'Projects'. Below this, a section titled 'Create a new Cron Job' features a progress indicator with three steps: '1 Choose service', '2 Configure', and '3 Deploy'. The main area displays six service options in a grid:

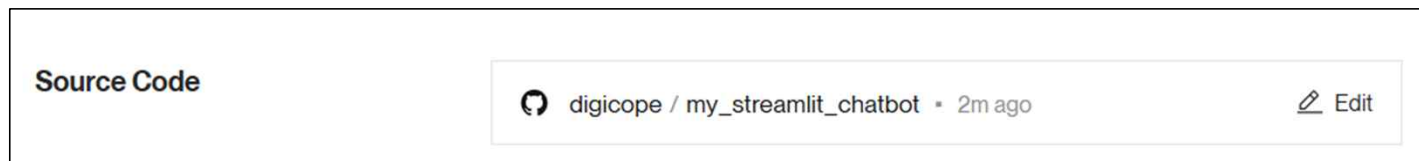
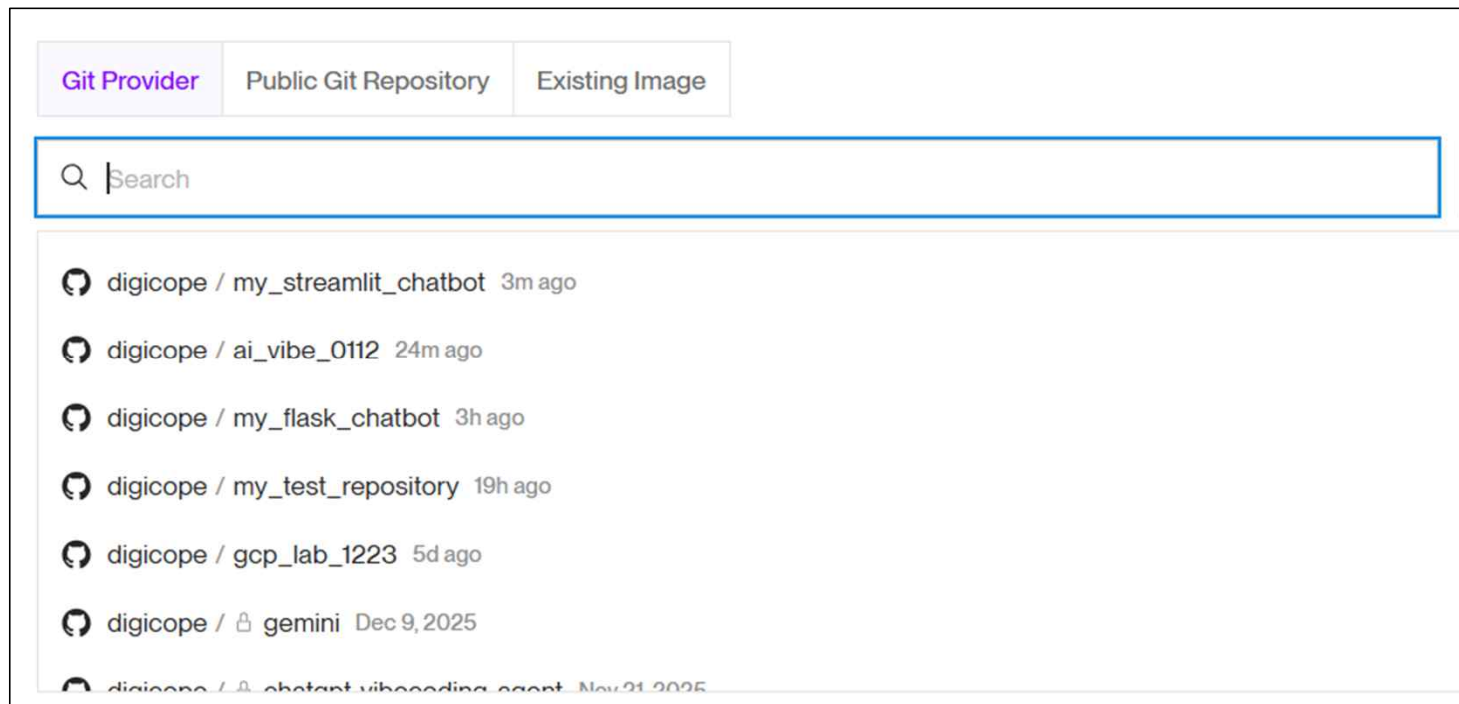
- Static Sites**: Static content served over a global CDN. Ideal for frontend, blogs, and content sites. Link: [New Static Site →](#)
- Web Services**: Dynamic web app. Ideal for full-stack apps, API servers, and mobile backends. Link: [New Web Service →](#) (highlighted with a red box)
- Private Services**: Web app hosted on a private network, accessible only from your other Render services. Link: [New Private Service →](#)
- Cron Jobs**: Short-lived tasks that run on a periodic schedule. Link: [New Cron Job →](#)
- Postgres**: Relational data storage. Supports point-in-time recovery, read replicas, and high availability. Link: [New Postgres →](#)
- Key Value**: Managed Redis®-compatible storage. Ideal for use as a shared cache, message broker, or job queue. Link: [New Key Value Instance →](#)

<https://dashboard.render.com/>

Render에서 이미 배포를 위한 프로젝트가 생성 되어 있을 경우에는 **Projects**에 있는 **My project**를 아래를 클릭하고 하단의 **New service**를 클릭한다음 **[Web Service]**를 클릭한다



New Web Service에서 배포에 사용할 저장소 my_streamlit_chatbot을 선택해준다



저장소를 선택 후 다음 항목을 설정한다.
대부분 기본 값으로 사용하면 된다

- **Name** : 서비스 이름 (기본값인 **저장소 이름**을 사용한다)
- **Environment** : **Python 3**
- **Region** : 기본값 사용 (**Oregon (US West)**)
- **Branch** : **main** (또는 master)
- **Build Command** : **pip install -r requirements.txt**
- **Start Command** : **streamlit run app.py --server.port \$PORT --server.address 0.0.0.0**
(파일명이 app.py인 경우)

Source Code

 digicope / my_streamlit_chatbot • 4m ago

 Edit

Name

A unique name for your web service.

my_streamlit_chatbot

Project

Optional

Add this web service to a [project](#) once it's created.

 My project

/

 Production

Language

Choose the [runtime environment](#) for this service.

Python 3

Branch

The Git branch to build and deploy.

main

Region

Your services in the same [region](#) can communicate over a [private network](#). You currently have services running in **Oregon**.

 Oregon (US West)

2 existing services

Deploy in a new region +



Start Command 설정에 유의한다

streamlit run app.py --server.port \$PORT --server.address 0.0.0.0

Root Directory Optional

If set, Render runs commands from this directory instead of the repository root. Additionally, code changes outside of this directory do not trigger an auto-deploy. Most commonly used with a [monorepo](#).

```
e.g. src
```

Build Command

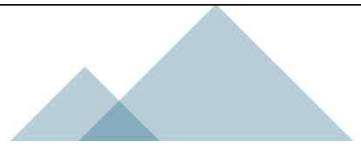
Render runs this command to build your app before each deploy.

```
$ pip install -r requirements.txt
```

Start Command

Render runs this command to start your app with each deploy.

```
$ streamlit run app.py --server.port $PORT --server.address 0.0.0.0
```



Instance Type을 Free 로 선택한다

Instance Type

For hobby projects

Free	512 MB (RAM)
\$0 / month	0.1 CPU

⚠ Upgrade to enable more features

Free instances spin down after periods of inactivity. They do not support SSH access, scaling, one-off jobs, or persistent disks. Select any paid instance type to enable these features.

For professional use

For more power and to get the most out of Render, we recommend using one of our paid instance types. All paid instances support:

- Zero Downtime
- SSH Access
- Scaling
- One-off jobs
- Support for persistent disks

Starter	512 MB (RAM)
\$7 / month	0.5 CPU

Standard	2 GB (RAM)
\$25 / month	1 CPU

Pro	4 GB (RAM)
\$85 / month	2 CPU

Pro Plus	8 GB (RAM)
\$175 / month	4 CPU

Pro Max	16 GB (RAM)
\$225 / month	4 CPU

Pro Ultra	32 GB (RAM)
\$450 / month	8 CPU

Need a [custom instance type](#)? We support up to 512 GB RAM and 64 CPUs.

[illegible]

Environment Variables

Set environment-specific config and secrets (such as API keys), then read those values from your code. [Learn more.](#)

OPENAI_API_KEY

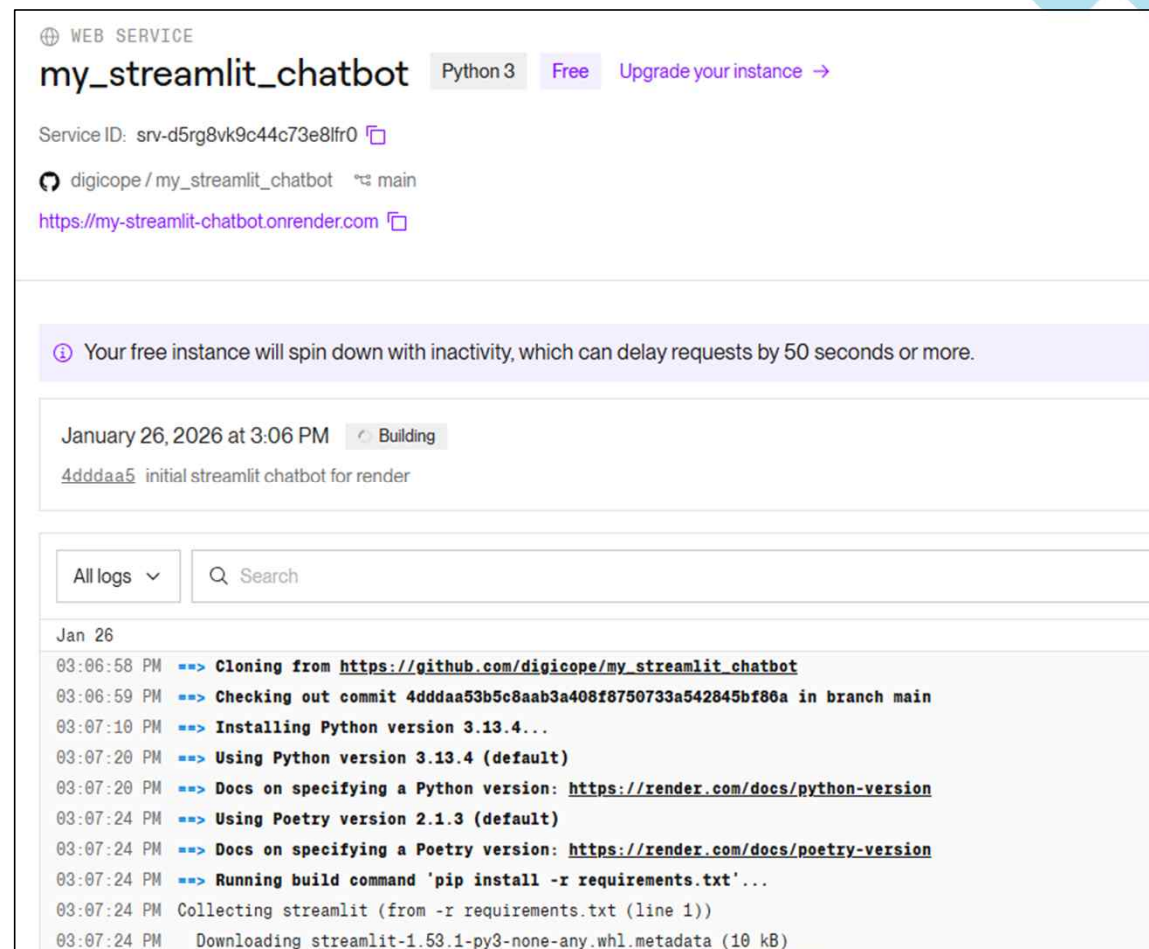
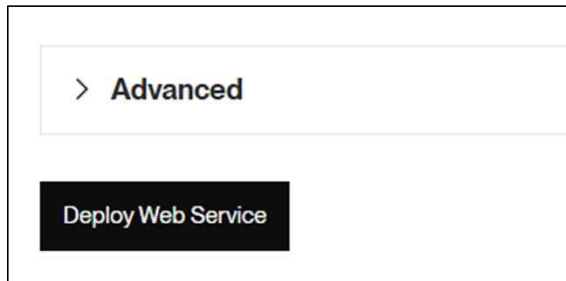
sk-proj-HMDVg2KP2e3paFyox5siqt0ezqDn10jFJZpMgK1x3Br8RXM-
g3mkyHAdo5EOx1_90CSK10ubuyWT3R1bkE1o40Db7cMkP3w0qD7Y

+ Add Environment Variable

 Add from .env



Web Service 배포를 위한 설정이 완료되면 좌측 하단의 [Deploy Web Service]를 누르면 아래와 같이 배포가 진행 된다 (배포 완료 까지 수분 소요)



잠시 기다리면 아래와 같이 접속 가능한 URL이 보여진다
URL을 클릭하면 배포된 Streamlit 챗봇 웹 서버로 접속된다

```
03:28:49 PM      ==> Your service is live 🎉
03:28:49 PM      ==>
03:28:49 PM      ==> //////////////////////////////////////
03:28:49 PM      ==>
03:28:49 PM      ==> Available at your primary URL https://my-streamlit-chatbot.onrender.com
03:28:49 PM      ==>
03:28:49 PM      ==> //////////////////////////////////////
03:29:42 PM [xj6mg] 2026-01-26 06:29:42,235 - httpx - INFO - HTTP Request: POST https://api.openai.com/v1/chat/completions "HTTP/1.1 200 OK"
```

배포 서비스 접속 실행 화면

The screenshot displays a web browser window with the URL `my-streamlit-chatbot.onrender.com`. The browser's address bar shows several tabs and bookmarks, including `digicope/ai_vibe_01...`, `digicope/gcp_lab_1...`, `gknu-digicope - G...`, and `Google Gemini`. The page title is `AI 채팅봇`.

Left Sidebar (Settings):

- 설정** (Settings)
- 모델 선택** (Model Selection): `gpt-4o-mini`
- Temperature**: A slider set to `0.70`.
- 대화 초기화** (Reset Conversation)
- 시스템 프롬프트** (System Prompt):
시스템 프롬프트를 수정할 수 있습니다.
당신은 친절하고 도움이 되는 AI 어시스턴트입니다.
사용자의 질문에 정확하고 유용한 답변을 제공하세요.

Right Chat Area:

- AI 채팅봇** (AI Chatbot)
OpenAI API를 사용한 대화형 AI 어시스턴트
- User Messages (Red icons):**
 - 안녕 나는 홍길동이야
 - 내 이름이 뭐지?
- Bot Responses (Yellow icons):**
 - 안녕하세요, 홍길동님! 반갑습니다. 어떻게 도와드릴까요?
 - 당신의 이름은 홍길동입니다. 다른 질문이나 도움이 필요한 부분이 있으면 말씀해 주세요!
- Bot Response (Yellow icon):**
하늘이 파란 이유는 대기 중의 산란 현상 때문입니다. 태양빛은 여러 색의 빛으로 이루어져 있으며, 이 중 파란색 빛은 다른 색에 비해 파장이 짧습니다. 대기 중의 분자와 미세한 입자들이 태양빛을 산란시키는 과정에서 파란색 빛이 더 많이 산란되어 하늘이 파랗게 보이게 됩니다. 이 현상을 레일리 산란이라고 하며, 이 때문에 해가 뜨거나 저를 볼 때 하늘이 주황색을 띠게 됩니다.

4. Render 에서 배포 서비스 업데이트 및 삭제

배포된 소스 업데이트 하기

- app.py 소스를 수정하고 Git Bash 아래 명령을 수행하면 자동 배포 업데이트 된다

git add .

git commit -m "update streamlit chatbot for render"

git push origin main

배포된 웹 서비스 삭제하기

render.com → Dashboard
→ Project : My project
→ Production : my_streamlit_chatbot
→ 좌측 메뉴의 Settings 클릭
→ 맨 아래의 [Delete Web Service]
를 클릭한다

← Environment

🌐 my_streamlit_chatbot

☰ Events

⚙️ Settings

MONITOR

🔍 Logs

📊 Metrics

MANAGE

📁 Environment

📄 Shell ⚡

📈 Scaling ⚡

📁 Previews

📖 Changelog

👤 Invite a friend

📞 Contact support

Maintenance Mode

Maintenance Mode ☐ Maintenance Mode D

Temporarily disable public access to your service. While enabled, Render serves a static maintenance page for all incoming requests. [Learn more.](#)

Custom Maintenance Page Optional

If provided, Render uses the specified URL for your maintenance page instead of serving the [default page](#).

⚡ PAID Maintenance mode is only available for paid instances.

🗑️ Delete Web Service

🛑 Suspend Web Service

Git hub의 모든 파일 삭제하기 (삭제후 복구 불가능)

- Git Bash에서 작업 디렉토리에 아래 명령을 수행한다

```
git checkout --orphan main  
git rm -rf .
```

```
echo "# clean repo" > README.md  
git add README.md  
git commit -m "initial commit"  
git push -f origin main
```

(업로드 된 파일이 복잡해서 삭제가 잘 안될 때는 저장소 자체를 삭제하고 다시 만든다)



감사합니다