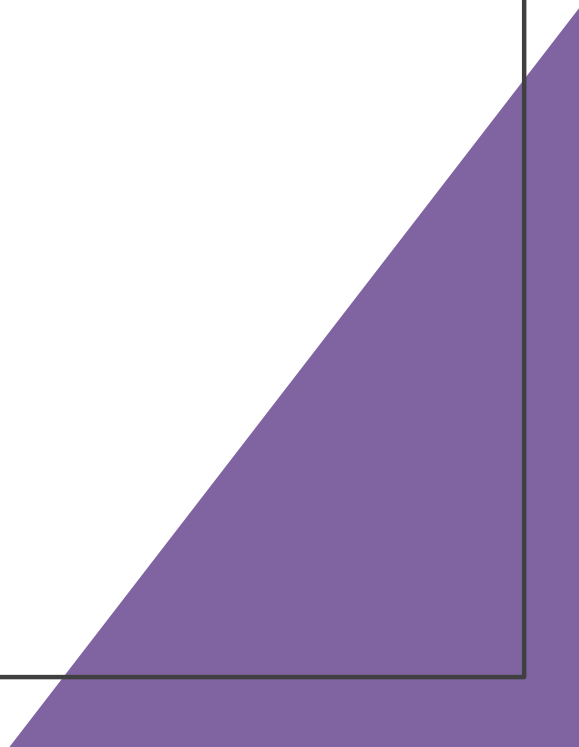


# OpenAI API 챗봇 제작 실습

Cursor AI 단계별 실습



## 1단계: 기본 환경 준비

chatbot.py를 만들어서 OpenAI API를 사용해 1회 질문을 보내고 답을 출력하는 코드를 함수나 환경변수 사용하지 않는 최소 코드만 작성해줘.

```
chatbot.py > ...
1  import openai
2
3  # OpenAI API 키 설정 (실제 키로 교체하세요)
4  openai.api_key = "your-api-key-here"
5
6  # 질문과 답변
7  response = openai.ChatCompletion.create(
8      model="gpt-3.5-turbo",
9      messages=[
10         {"role": "user", "content": "안녕하세요! 오늘 날씨가 어떨까요?"}
11     ]
12 )
13
14 # 답변 출력
15 print(response.choices[0].message.content)
16
```



## OpenAI API 키 설정

OpenAI API 키를 복사하여 아래와 같이 소스코드 안에 “openai.api\_key” 설정 값으로 따옴표 안에 한 줄로 붙여 넣는다.

# OpenAI API 키 설정 (실제 키로 교체하세요)

```
openai.api_key = "sk-proj-D1axXtlwTtEj8Y-wcxEGvS-  
qSb_jtnnZaXqNxH0j0pXY5PsKbsXmwh4szYkl22hhC73V  
...(중략) ... zhPc0LpjhDoLaG0XJku9vTIQA"
```

```
import openai
```

```
# OpenAI API 키 설정 (실제 키로 교체하세요)
```

```
openai.api_key = "sk-proj-D1axXtIwTtEj8Y-wcxEGvS-qSb_jtnnZaXqNxH0j0pXY5PsKbsXmwh4sz
```

## OpenAI 모듈과 dotenv 모듈 설치

chatbot.py를 Editor 로 열고 실행하면 openai 모듈이 설치되어 있지 않으므로 모듈설치가 필요하다. 환경 변수 처리 관련 모듈인 dotenv 모듈도 같이 설치해준다

Cursor의 터미널 창에서 아래 명령을 실행시킨다

**pip install openai python-dotenv**

```
PS C:\Users\storm\바탕 화면\My_Cursor_Lab\Chatbot_Project> pip install openai python-dotenv
Requirement already satisfied: openai in c:\users\storm\appdata\local\programs\python\python313\lib\site-packages (1.105.0)
Requirement already satisfied: python-dotenv in c:\users\storm\appdata\local\programs\python\python313\lib\site-packages (1.1.1)
Requirement already satisfied: anyio<5,>=3.5.0 in c:\users\storm\appdata\local\programs\python\python313\lib\site-packages (from openai) (3.7.1)
```

## OpenAI API 버전에 따른 오류 발생

코드를 실행시키면 OpenAI API 구 버전 으로 생성된 경우 실행 오류가 나므로 **최신 버전 (1.0.0 이상)**으로 소스를 수정한다

```
⊗ Traceback (most recent call last):
  File "c:\Users\storm\바탕 화면\My_Cursor_Lab\Chatbot_Project\chatbot.py", line 7, in <module>
    response = openai.ChatCompletion.create(
        model="gpt-3.5-turbo",
        ...<2 lines>...
    ]
  )
  File "C:\Users\storm\AppData\Local\Programs\Python\Python313\Lib\site-packages\openai\lib\_old_api.py", line 39
    raise APIRemovedInV1(symbol=self._symbol)
  r the API.

You can run `openai migrate` to automatically upgrade your codebase to use the 1.0.0 interface.

Alternatively, you can pin your installation to the old version, e.g. `pip install openai==0.28`

A detailed migration guide is available here: https://github.com/openai/openai-python/discussions/742

PS C:\Users\storm\바탕 화면\My_Cursor_Lab\Chatbot_Project> █
```

## <구버전으로 실행 오류 발생 수정 질의

OpenAI 최신 버전 스타일로 다시 만들어줘.

```
chatbot.py > ...
import openai
1 from openai import OpenAI
2
# OpenAI API 키 설정 (실제 키로 교체하세요)
openai.api_key = "sk-proj-D1axXtIwTtEj8Y-wcxEGvS-qSb_jtnnZaXqNxH0j0pXY5PsKbsXmwh4sz"
3 # OpenAI 클라이언트 생성
4 client = OpenAI(api_key="sk-proj-D1axXtIwTtEj8Y-wcxEGvS-qSb_jtnnZaXqNxH0j0pXY5PsKbsXmwh4sz")
5
6 # 질문과 답변
response = openai.ChatCompletion.create(
7 response = client.chat.completions.create(
8     model="gpt-3.5-turbo",
9     messages=[
10         {"role": "user", "content": "안녕하세요! 오늘 날씨가 어떨까요?"}
11     ]
12 )
13
14 # 답변 출력
15 print(response.choices[0].message.content)
```

수정된 코드를 다시 실행시키면 터미널 출력에서 잘 실행됨을 확인할 수 있다. 만일 오류가 나면 다시 수정 요청한다

Problems Output Debug Console Terminal Ports

Python + - [ ] [X] ... ^ X

```
PS C:\Users\storm\바탕 화면\My_Cursor_Lab\Chatbot_Project> & C:/Users/storm/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/storm/바탕 화면/My_Cursor_Lab/Chatbot_Project/chatbot.py"
```

- 안녕하세요! 저는 인공지능 챗봇이라서 날씨를 확인할 수는 없지만, 여기 있는 곳의 날씨를 미리 알려드릴 수는 있어요. 어느 지역의 날씨를 알고 싶으신가요?

- PS C:\Users\storm\바탕 화면\My\_Cursor\_Lab\Chatbot\_Project>



## 2단계: GPT Model 변경

소스 코드의 model="gpt-3.5-turbo" 부분을 선택하고 Ctrl+K를 누르고 아래 내용으로 질의 한다. (gpt-4o 나 gpt-4o-mini 사용)

GPT 최신 모델로 변경해줘.

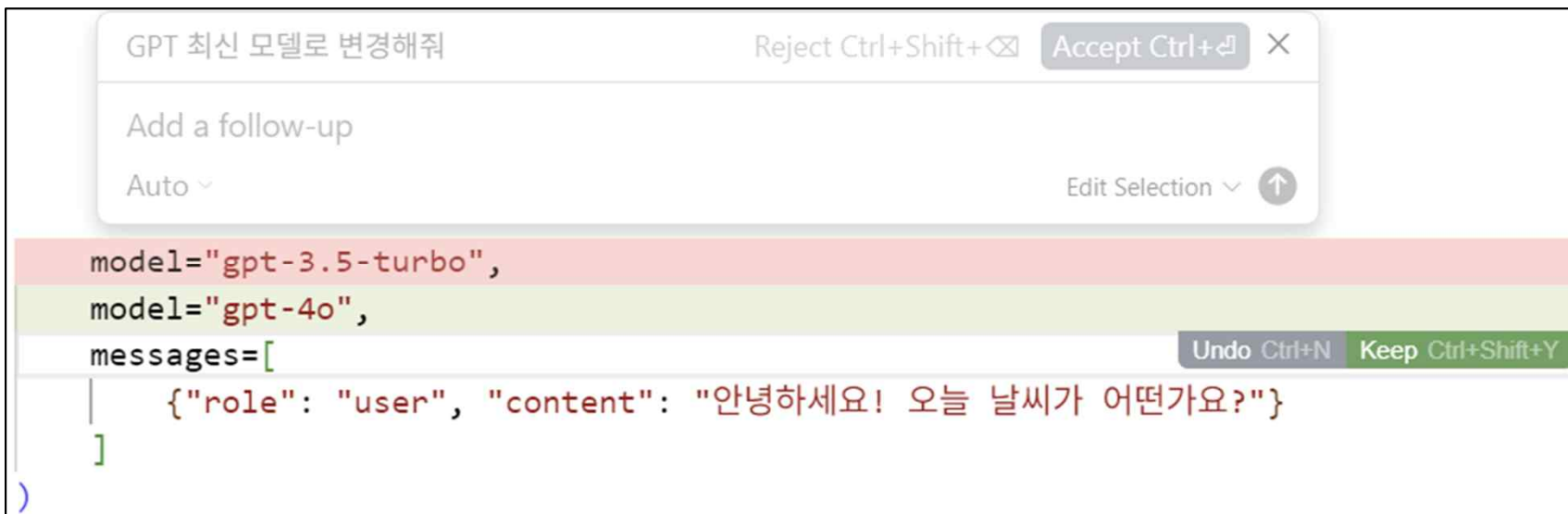


```
GPT 최신 모델로 변경해줘
```

Auto ▾ Edit Selection ▾ ⬆

```
model="gpt-3.5-turbo",  
messages=[  
  {"role": "user", "content": "안녕하세요! 오늘 날씨가 어떨까요?"}  
]
```

Inline Edit  
(Ctrl+K)



```
GPT 최신 모델로 변경해줘
```

Reject Ctrl+Shift+⌘ Accept Ctrl+⌘ X

Add a follow-up

Auto ▾ Edit Selection ▾ ⬆

```
model="gpt-3.5-turbo",  
model="gpt-4o",  
messages=[  
  {"role": "user", "content": "안녕하세요! 오늘 날씨가 어떨까요?"}  
]  
)
```

Undo Ctrl+N Keep Ctrl+Shift+Y



### 3단계: API 키 환경 변수 등 관리

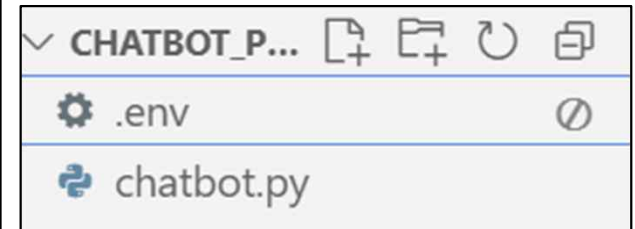
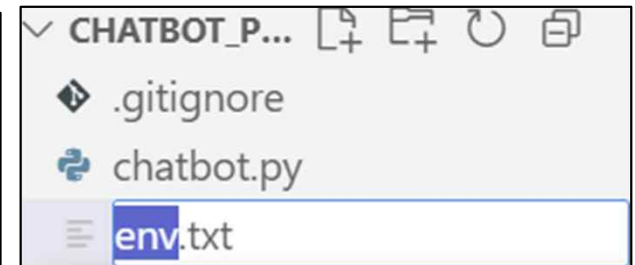
API 키를 코드에 직접 쓰지 말고 os.environ에서 읽어오도록 수정해줘.

```
import os
from openai import OpenAI
from dotenv import load_dotenv

# .env 파일 로드
load_dotenv()

# OpenAI 클라이언트 생성
client = OpenAI(api_key="sk-proj-D1axXtIwTtEj8Y-wcxEGvS-qSb_jtnnZaXqNxH0j0pXY5PsKbs")
client = OpenAI(api_key=os.getenv("OPENAI_API_KEY"))

# 질문과 답변
response = client.chat.completions.create(
    model="gpt-4o",
    messages=[
        {"role": "user", "content": "안녕하세요! 오늘 날씨가 어떨까요?"}
    ]
)
```



환경 변수 설정 파일이 생성된다. 만일 환경 변수 파일(.env)이 만들어지지 않거나 다른 이름으로 만들어지면 .env 파일을 직접 생성하거나 파일명을 .env로 직접 변경해준다

## 4단계: 사용자 인터페이스 개선

질문 입력 시 'You:', 응답 출력 시 'Bot:' 이 앞에 붙도록 수정해줘.

```
# 질문과 답변
question = "안녕하세요! 오늘 날씨가 어떨까요?"
print(f"You: {question}")

response = client.chat.completions.create(
    model="gpt-4o",
    messages=[
        {"role": "user", "content": "안녕하세요! 오늘 날씨가 어떨까요?"},
        {"role": "user", "content": question}
    ]
)

# 답변 출력
print(response.choices[0].message.content)
print(f"Bot: {response.choices[0].message.content}")
```

```
chatbot.py
You: 안녕하세요! 오늘 날씨가 어떨까요?
Bot: 안녕하세요! 죄송하지만 실시간 날씨 정보를 제공할 수는 없습니다. 대신, 스마트폰의 날씨 애플리케이션이나 인터넷의 날씨 사이트를 통해 오늘 날씨를 확인하실 수 있습니다. 도움이 필요하시면 언제든지 말씀해 주세요!
○ PS C:\Users\storm\바탕 화면\My_Cursor_Lab\Chatbot_Project>
```

## 5단계: 함수 구조화 및 루프 추가

사용자 입력을 받아 OpenAI API에 전달하고, 응답을 리턴하는 함수(chat\_with\_gpt)를 만들어줘.  
사용자가 'quit'을 입력하기 전까지 반복적으로 실행하도록 구현해줘.

chatbot.py > ...

```
11 def chat_with_gpt(user_input):
11 def chat_with_gpt(user_input):
12     """
13     사용자 입력을 받아 OpenAI API에 전달하고 응답을 반환하는 함수
14
15     Args:
16     |     user_input (str): 사용자가 입력한 질문
17
18     Returns:
19     |     str: GPT의 응답
20     """
21     try:
22         response = client.chat.completions.create(
23             model="gpt-4o",
24             messages=[
25                 {"role": "user", "content": user_input}
26             ]
27         )
28         return response.choices[0].message.content
29     except Exception as e:
30         return f"오류가 발생했습니다: {str(e)}"
31
```

채팅봇에 오신 것을 환영합니다! 'quit'을 입력하면

You: 하늘은 왜 파란가요?

Bot: 하늘이 파란 이유는 주로 빛의 산란 현상 때문이며, 이들 각각은 서로 다른 파장을 갖고 있어 태양빛을 통과할 때, 그 빛은 산란됩니다. 파장이 짧기 때문에, 하늘을 볼 때 우리는 파란색을 더 많이 보게 됩니다. 이 산란 효과 때문에 하늘이 파랗게 보이는 것입니다.

You: 노란이유는?

Bot: "노란"이라는 단어와 관련된 질문은 여러 가지 있습니다. 노란색은 다양한 이유로 사용되거나 나타날 수 있습니다. 예를 들어:

1. **\*\*자연\*\***: 노란색은 해바라기, 단풍 등의 자연에서 자주 발견됩니다.
2. **\*\*상징성\*\***: 문화나 문맥에 따라 다양하게 해석될 수 있습니다. 서양에서는 경고나 주의의 의미로 사용되기도 합니다.

## 6단계: 간단한 메모리 기능

대화 히스토리를 리스트에 저장해서, 이전 질문과 답변을 맥락으로 포함해주는 챗봇으로 수정해줘.

```
# 대화 히스토리를 저장할 리스트
```

```
conversation_history = []
```

```
while True:
```

```
    # 사용자 입력 받기
```

```
    user_input = input("You: ").strip()
```

```
    # 'quit' 입력 시 종료
```

```
    if user_input.lower() == 'quit':
```

```
        print("Bot: 안녕히 가세요!")
```

```
        break
```

```
    # 빈 입력 처리
```

```
    if not user_input:
```

```
        print("Bot: 질문을 입력해주세요.")
```

```
        continue
```

```
    # GPT와 채팅
```

```
    bot_response = chat_with_gpt(user_input)
```

```
    # GPT와 채팅 (히스토리 포함)
```

```
    bot_response = chat_with_gpt(user_input, conversation_history)
```

```
    print(f"Bot: {bot_response}\n")
```

```
    # 대화 히스토리에 현재 대화 추가
```

≡ text

채팅봇에 오신 것을 환영합니다! 'quit'을 입력하면 종료

You: 내 이름은 김철수야

Bot: 안녕하세요, 김철수님! 만나서 반가워요. 무엇을 도

You: 내 나이는 25살이야

Bot: 김철수님, 25세시군요! 젊고 활기찬 나이네요. 어땐

You: 내 직업이 뭐라고 했지?

Bot: 죄송하지만 김철수님의 직업에 대해서는 아직 말씀하

이제 챗봇이 이전 대화 내용을 기억하여 더 자연스럽게 맥락적인 대화를 할 수 있습니다!



## 대화 내용을 기억하는 대화 내용

```
PS C:\Users\storm\바탕 화면\My_Cursor_Lab\Chatbot_Project> & C:/Users/storm/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/storm/바탕 화면/My_Cursor_Lab/Chatbot_Project/chatbot.py"
```

- 채팅봇에 오신 것을 환영합니다! 'quit'을 입력하면 종료됩니다.

You: 하늘이 파란 이유는?

Bot: 하늘이 파란 이유는 주로 빛의 산란 때문입니다. 태양빛은 여러 파장의 빛으로 구성되어 있으며, 지구의 대기층을 통과할 때 이 빛들이 산란됩니다. 이 중에서 파장이 짧은 파란색 빛이 다른 색보다 더 많이 산란되는데, 이는 '레이리 산란'이라고 불리는 현상 때문입니다. 레일리 산란은 빛의 파장과 산란 정도가 반비례한다는 원리에 기반하고 있습니다. 이러한 이유로 우리가 보는 하늘은 주로 파란색으로 보이는 것입니다.

You: 노란 이유는?

Bot: 하늘이 노랗게 보이는 상황은 주로 태양이 지평선에 가까워질 때 발생합니다, 즉 일출이나 일몰 시에 주로 관찰됩니다. 이때 태양빛은 대기를 통과하는 경로가 길어지면서 더 많은 공기 분자와 입자들에 의해 산란됩니다. 파란색과 초록색과 같은 짧은 파장의 빛은 산란되어 사라지고, 상대적으로 파장이 긴 빨간색, 주황색,

You: quit

Bot: 안녕히 가세요!

## 7단계: 에러 처리

API 호출이 실패했을 때 예외 처리(try-except)를 추가하고, 오류 메시지를 출력하도록 해줘.

```
chatbot.py ×
chatbot.py > main
1  import os
2  from openai import OpenAI
3  from openai import APIError, RateLimitError, APITimeoutError, APIConnectionError
4  from dotenv import load_dotenv
5
6  # .env 파일 로드
7  load_dotenv()
8
9  # API 키 확인
10 api_key = os.getenv("OPENAI_API_KEY")
11 if not api_key:
12     print("오류: OPENAI_API_KEY가 설정되지 않았습니다.")
13     print(".env 파일에 OPENAI_API_KEY를 설정해주세요.")
14     exit(1)
15
16 # OpenAI 클라이언트 생성
17 client = OpenAI(api_key=os.getenv("OPENAI_API_KEY"))
18 try:
19     client = OpenAI(api_key=api_key)
20     print("OpenAI 클라이언트가 성공적으로 초기화되었습니다.")
21 except Exception as e:
22     print(f"OpenAI 클라이언트 초기화 실패: {str(e)}")
23     exit(1)
```

## 8단계: 파라미터 선택기능

temperature 값을 0.2와 1.0 두 가지 모드로 선택할 수 있게 수정해줘.

```
def main():
    """
    메인 함수 - 사용자가 'quit'을 입력할 때까지 반복적으로 채팅
    """
    print("채팅봇에 오신 것을 환영합니다! 'quit'을 입력하면 종료됩니다.\n")
    print("채팅봇에 오신 것을 환영합니다!")
    print("Temperature 모드를 선택해주세요:")
    print("1. 정확한 모드 (0.2) - 일관되고 정확한 응답")
    print("2. 창의적 모드 (1.0) - 다양하고 창의적인 응답")

    while True:
        mode_choice = input("모드를 선택하세요 (1 또는 2): ").strip()
        if mode_choice == "1":
            temperature = 0.2
            print("정확한 모드가 선택되었습니다.\n")
            break
        elif mode_choice == "2":
            temperature = 1.0
            print("창의적 모드가 선택되었습니다.\n")
            break
        else:
            print("1 또는 2를 입력해주세요.")

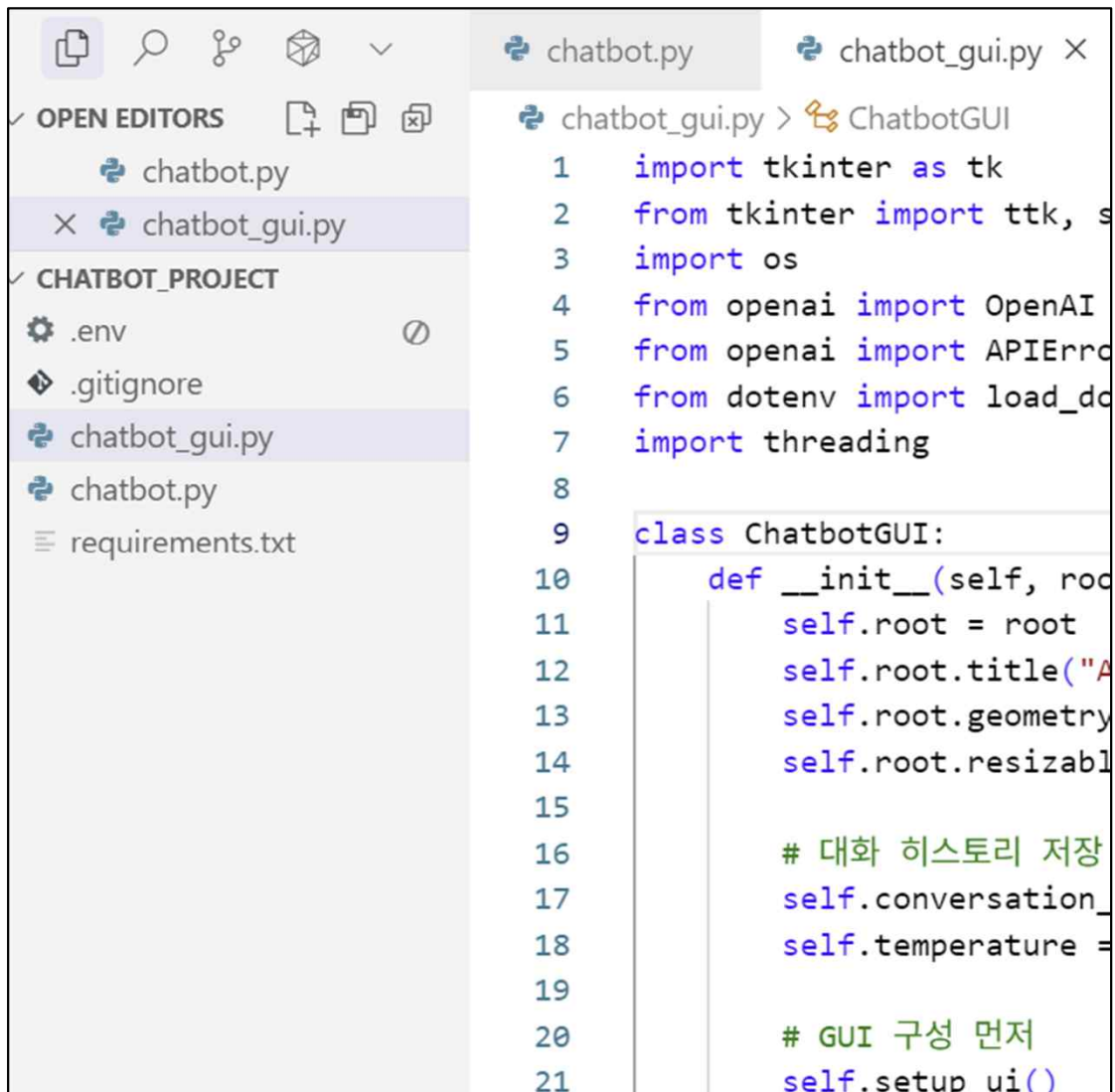
    print("'quit'을 입력하면 종료됩니다.\n")
```

OpenAI 클라이언트가 성공적으로 초기화되었습니다.  
채팅봇에 오신 것을 환영합니다!  
Temperature 모드를 선택해주세요:  
1. 정확한 모드 (0.2) - 일관되고 정확한 응답  
2. 창의적 모드 (1.0) - 다양하고 창의적인 응답  
모드를 선택하세요 (1 또는 2):



## 9단계: GUI 버전 구현

지금까지 만든 예제를 Tkinter GUI로 새로운 파일로 작성해줘.



```
chatbot.py chatbot_gui.py X
chatbot_gui.py > ChatbotGUI
1 import tkinter as tk
2 from tkinter import ttk, s
3 import os
4 from openai import OpenAI
5 from openai import APIError
6 from dotenv import load_dotenv
7 import threading
8
9 class ChatbotGUI:
10     def __init__(self, root):
11         self.root = root
12         self.root.title("A
13         self.root.geometry(
14         self.root.resizabl
15
16         # 대화 히스토리 저장
17         self.conversation_
18         self.temperature =
19
20         # GUI 구성 먼저
21         self.setup_ui()
```

지금까지 만든 예제를 Tkinter GUI로 새로운 파일로 작성해줘.

지금까지 만든 챗봇을 Tkinter GUI로 새로운 파일로 작성하겠습니다.

➔ Tkinter GUI 챗봇 파일 생성

5 to-dos

chatbot\_gui.py +237 ✓

5 of 5 Done

완료되었습니다! Tkinter GUI 버전의 챗봇을 `chatbot_gui.py` 파일로 생성했습니다.

### 주요 기능:

#### 1. GUI 구성 요소

- 채팅 영역: 스크롤 가능한 텍스트 위젯으로 대화 내용 표시
- 입력 필드: 사용자 메시지 입력
- 전송 버튼: 메시지 전송

## AI 챗봇

대화

System: OpenAI 클라이언트가 성공적으로 초기화되었습니다.

System: 안녕하세요! AI 챗봇입니다. 무엇을 도와드릴까요?

You: 안녕?

Bot: 안녕하세요! 어떻게 도와드릴까요?

You: 오늘 날씨 알려줘

Bot: 죄송하지만, 현재 실시간 정보를 제공할 수 없습니다. 대신, 스마트폰의 날씨 앱이나 인터넷의 날씨 웹사이트를 통해 오늘의 날씨를 확인하실 수 있습니다. 다른 질문이 있으면 말씀해 주세요!

You: 하늘이 파란 이유는?

Bot: 하늘이 파란 이유는 주로 빛의 산란 현상 때문입니다. 태양빛은 여러 가지 색의 빛으로 구성되어 있으며, 이 빛들은 각각 다른 파장을 가지고 있습니다. 지구의 대기는 주로 질소와 산소로 이루어져 있으며, 이 분자들은 태양빛을 산란시킵니다.

빛의 산란은 파장이 짧을수록 더 강하게 일어납니다. 파란색 빛은 상대적으로 짧은 파장을 가지고 있기 때문에 대기에서 더 많이 산란됩니다. 그래서 태양빛이 대기를 통과할 때, 파란색 빛이 사방으로 퍼지게 되어 하늘이 파랗게 보이게 됩니다. 이 현상을 레일리 산란이라고 합니다.

반면, 해질녘이나 해돋이에는 태양빛이 대기를 더 두껍게 통과해야 하기 때문에 파란색 빛이 더 많이 산란되고, 상대적으로 파장이 긴 빨간색과 주황색 빛이 더 많이 남아 하늘이 붉게 보이게 됩니다.

응답 모드

☒ 정확한 모드 (0.2)☐ 창의적 모드 (1.0)

전송

준비됨

# 10단계: GUI 화면 디자인 개선

GUI화면과 대화 텍스트 폰트 및 색상을 좀더 예쁘게 만들어줘.

```
def setup_theme(self):  
    """테마 및 색상 설정"""  
    # 색상 팔레트  
    self.colors = {  
        'bg_primary': '#f8f9fa',      # 메인 배경색  
        'bg_secondary': '#ffffff',    # 카드/프레임 배경색  
        'bg_chat': '#ffffff',         # 채팅 영역 배경색  
        'text_primary': '#2c3e50',    # 주요 텍스트 색상  
        'text_secondary': '#6c757d',  # 보조 텍스트 색상  
        'accent_blue': '#007bff',     # 사용자 메시지 색상  
        'accent_green': '#28a745',    # 봇 메시지 색상  
        'accent_orange': '#fd7e14',   # 시스템 메시지 색상  
        'border': '#dee2e6',          # 테두리 색상  
        'button_bg': '#007bff',       # 버튼 배경색  
        'button_hover': '#0056b3',    # 버튼 호버 색상  
    }  
  
    # 폰트 설정  
    self.fonts = {  
        'title': ('Segoe UI', 18, 'bold'),  
        'subtitle': ('Segoe UI', 12, 'bold'),  
        'body': ('Segoe UI', 11),  
        'chat': ('Segoe UI', 10),  
        'small': ('Segoe UI', 9)  
    }
```

~ 3 / 21 ~

Undo all Ctrl+Shift+⌘

Keep all Ctrl+⌘

현재 GUI 화면의 폰트와 색상을 더 예쁘게 만들어드리겠습니다. 먼저 현재 코드를 확인해보겠습니다.

Read chatbot\_gui.py

현재 코드를 확인했습니다. GUI의 폰트와 색상을 더 예쁘게 개선해보겠습니다. 다음과 같은 개선사항을 적용하겠습니다:

1. 더 모던한 폰트 사용 (Segoe UI, Noto Sans 등)
2. 색상 팔레트 개선 (더 부드럽고 현대적인 색상)
3. 배경색과 텍스트 색상 조화
4. 더 나은 시각적 계층 구조

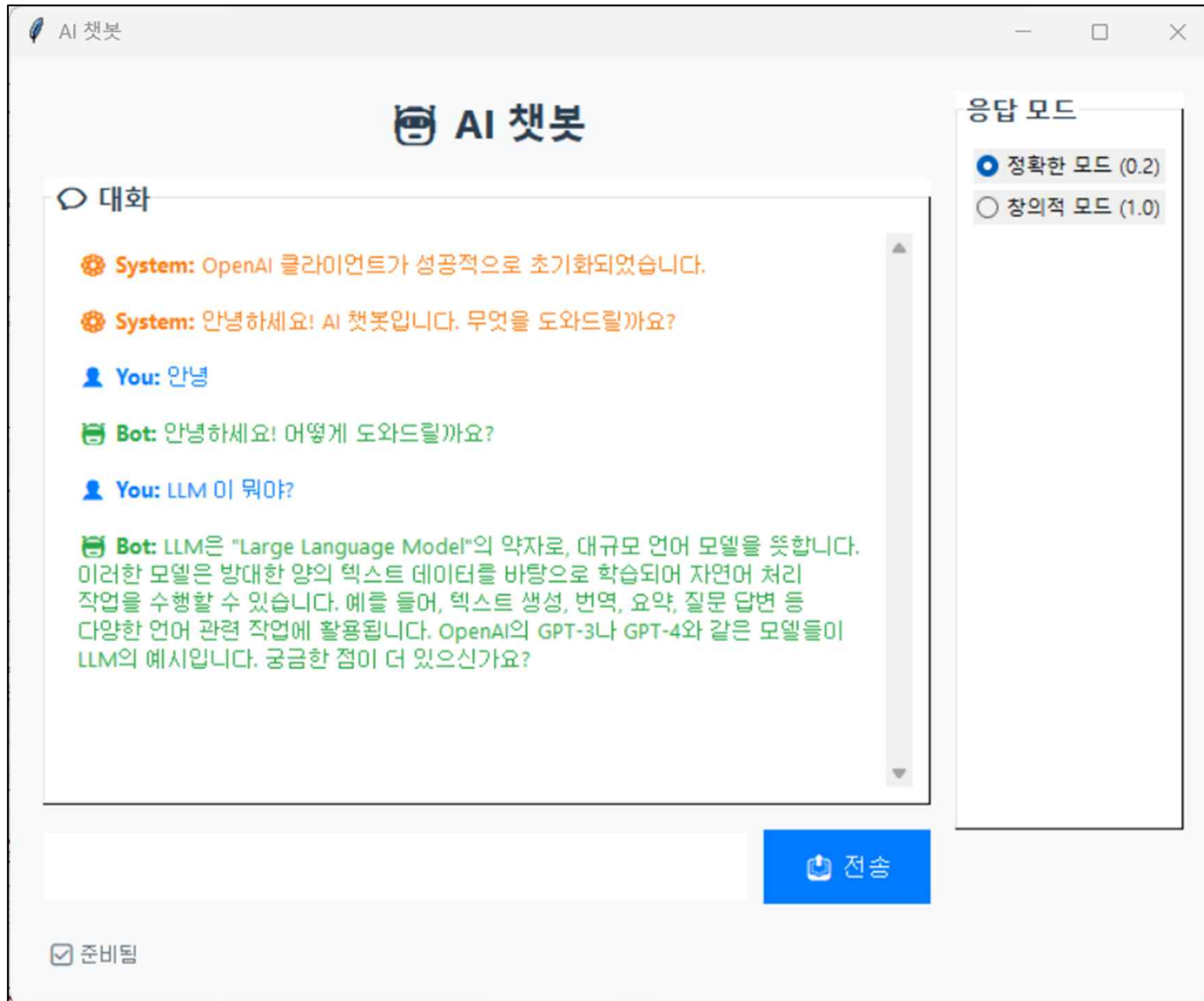
➔ GUI 폰트를 더 모던하고 읽기 좋은 ... 4 to-dos ↕

🔗 chatbot\_gui.py +33 -1 ✓ ↕

⚙️ 1 of 4 Done ↕

➔ 색상 팔레트를 더 현대적이고 부드... 1 of 4 ↕

# 개선된 GUI 화면



## 11단계: 대화 내용 저장 및 불러오기 구현

대화 내용을 현재 시간 정보와 함께 DB로 항상 저장하고  
[History] 버튼을 누르면 대화 내용 전체를 목록으로 볼수 있는  
기능을 추가해줘.

```
chatbot_gui.py ×  
chatbot_gui.py > ChatbotGUI  
1 import tkinter as tk  
2 from tkinter import ttk, scrolledtext, messagebox  
3 import os  
4 import sqlite3  
5 from datetime import datetime  
6 from openai import OpenAI  
7 from openai import APIError, RateLimitError,  
8 from dotenv import load_dotenv  
9 import threading  
10  
11 class ChatbotGUI:  
12     def __init__(self, root):  
13         self.root = root
```

대화 내용을 현재 시간 정보와 함께 DB로 항상 저장  
하고

[History] 버튼을 누르면 대화 내용 전체를 목록으로  
볼수 있는 기능을 추가해줘

대화 내용을 DB에 저장하고 히스토리 기능을 추가해드리겠습니다.  
SQLite를 사용하여 간단하고 효율적인 데이터베이스  
구조를 만들어보겠습니다.

➔ SQLite 데이터베이스 설정 및 대화 ... 4 to-dos ↕

먼저 필요한 라이브러리를 requirements.txt에 추가하겠습니다:

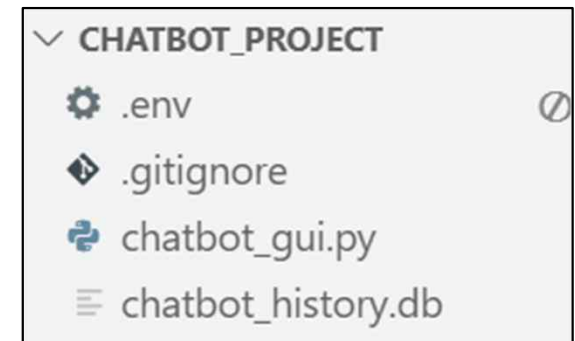
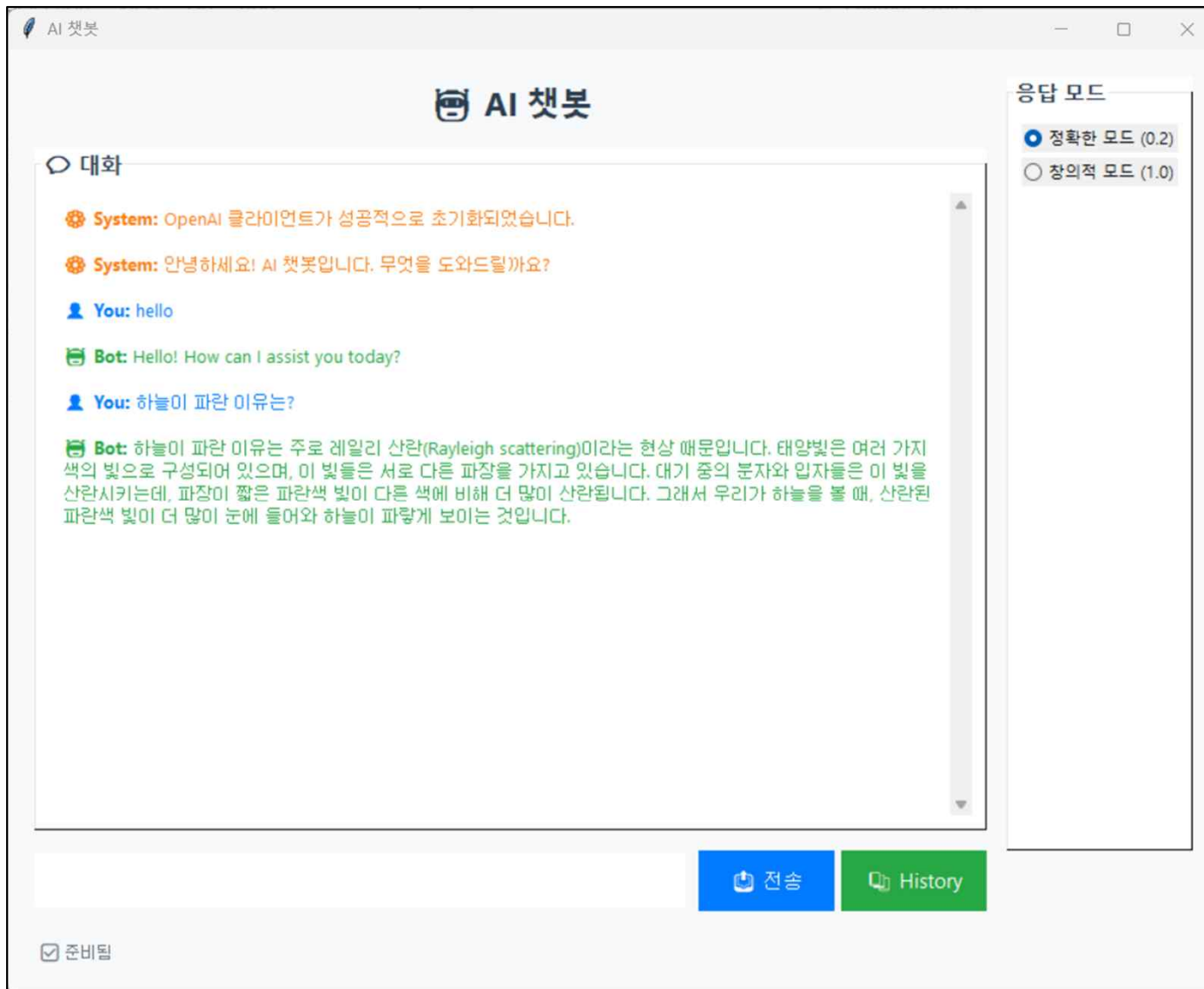
Read requirements.txt

☰ requirements.txt +1 • ✕ ✓ ↕


이제 chatbot\_gui.py에 데이터베이스 기능을 추가하겠습니다:




만일 대화 내용이 안보이면 여러 번 반복해서 오류 수정 요청을 해서 소스를 계속 수정한다  
예시 : 대화 내용이 안보여 다시 수정해줘.




# 대화 히스토리 출력 화면


 대화 히스토리


 대화 히스토리

테스트 데이터 추가


새로고침

 2025-09-05 15:16:45


 Temperature: 0.2


 You:


하늘이 파란 이유는?

 Bot:


하늘이 파란 이유는 주로 레일리 산란(Rayleigh scattering)이라는 현상 때문입니다. 태양빛은 여러 가지 색의 빛으로 구성되어 있으며, 이 빛들은 서로 다른 파장을 가지고 있습니다. 대기 중의 분자와 입자들은 이 빛을 산란시키는데, 파장이 짧은 파란색 빛이 다른 색에 비해 더 많이 산란됩니다. 그래서 우리가 하늘을 볼 때, 산란된 파란색 빛이 더 많이

 2025-09-05 15:16:25


 Temperature: 0.2


 You:


hello

 Bot:

Hello! How can I assist you today?

 2025-09-05 15:12:05

 Temperature: 0.7

 You:

...



# Thank You!!

