

Vite+Netlify에서 ChatGPT API 활용하기

2025.04.07.

AI융합교육의 동향과 이슈

5주차 강의자료

30초 요약

- Vite+Netlify 환경설정 되돌아보기
- ChatGPT API 넣기
- 간단한 챗봇 만들기
- 중간 발표 안내



ChatGPT API가 작동하는 방식

API Application Programming Interface

- 어떤 프로그램이 다른 프로그램과 상호작용할 수 있도록 도와주는 인터페이스
- 한 소프트웨어가 다른 소프트웨어의 기능이나 데이터에 접근할 수 있는 방법과 규칙
- Application Programming Interface
 - 사용되는 프로그램 또는 서비스
 - 개발자가 프로그램을 만들 때 사용하는 코드
 - 프로그램과 프로그램 사이의 명확한 약속/접점
- “앱이나 프로그램끼리 대화할 수 있게 도와주는 통역사 또는 메뉴판”

API의 구성요소

- 엔드포인트 Endpoint
 - 요청을 보내는 대상 URL
- HTTP 메서드 HTTP Method
 - 요청 방식
- 헤더 Header
 - 인증 정보, 데이터 형식 등을 담음
- 본문 Body
 - 실제로 전달할 데이터(JSON 형식)
- 응답 Response
 - API가 반환하는 데이터 (성공 or 실패 정보, 요청 결과 등)

API Key의 역할

- API는 다른 시스템의 기능이나 데이터를 사용할 수 있게 해주는 프로그래밍 인터페이스
- 주로 HTTP 요청 (REST API)을 통해 사용
- 사용자는 API를 호출해 데이터를 보내고, 결과를 수신함
- 보통 인증 토큰(API Key)이 필요하며, 이를 통해 보안을 유지함
- API Key는 고유 키 문자열로 사용자를 식별하는 것
“누가 이 요청을 보냈는지 확인하기 위한 비밀번호 같은 고유 문자열”
- 따라서 API Key의 보안이 굉장히 중요함

API Key의 필요성

- 인증(Authentication): 이 요청이 누구로부터 왔는지 확인
- 사용량 제한(Rate limiting): 한 사람이 너무 많이 쓰지 않도록 제한
- 통계 추적(Analytics): 누가 얼마나 API를 쓰고 있는지 기록

API Key 보안의 중요성

- 누군가 GitHub에 API Key가 포함된 코드를 확인하고 내 API Key를 가져다 사용할 수 있음
- 특히 ChatGPT API는 사용하는 양 만큼 API 사용료가 청구되므로, API Key 보안이 중요함

Home > 전체기사

사이버 범죄자들, 오픈AI API 키 스크랩해서 GPT4를 무허가로 사용한다

무료 서비스 해줬다가 AI 학습에 당한 부산대 "서버비 감당 안 돼"

2023.07.06. 오후 2:46.

□ 가가 ↑ 🖨

API Key와 환경변수

- 환경변수:
실행되는 시스템(서버, 로컬 컴퓨터 등)에 설정된 “외부로 노출되지 않는 설정 값 저장소”
- API Key 같은 민감한 값을 코드에 직접 쓰지 않고, 숨겨진 공간에 보관해두는 방식

Calling the ChatGPT API with Environment Variables



Calling the ChatGPT API with Environment Variables



1. 사용자가 질문을 입력함
2. 브라우저가 서버에게 요청
3. 서버가 환경변수에서 API Key 꺼냄
4. 서버가 ChatGPT API에 요청을 보냄
5. 서버가 안전하게 결과 전달
6. 사용자가 결과를 확인하고 이해

Vite+Netlify 프로젝트에 ChatGPT API 추가하기

모든 내용은
구글 문서에도 정리되어 있습니다.

Vite+Netlify 프로젝트 세팅하기

- 내 문서에 “GitHub” 폴더 만들고 이동 → 이미 되어 있으면 생략
- GitHub 폴더에서 마우스 오른쪽 클릭 후 ‘터미널에서 열기’
- `npm create vite@latest` 입력
 - project name 입력(프로젝트 이름이 폴더 이름이 됨)
package name 입력(프로젝트 명과 동일하게)
framework는 Vanilla, variant는 Javascript 선택

Vite+Netlify 프로젝트 세팅하기(계속)

```
◇ Project name:
  SMWU-static-test
◇ Package name:
  smwu-static-test
◇ Select a framework:
  Vanilla
◇ Select a variant:
  JavaScript
◇ Scaffolding project in C:\Users\eduwa\OneDrive\문서\GitHub\SMWU-static-test...
|
  Done. Now run:

  cd SMWU-static-test
  npm install
  npm run dev

npm notice
npm notice New major version of npm available! 10.7.0 -> 11.2.0
npm notice Changelog: https://github.com/npm/cli/releases/tag/v11.2.0
npm notice To update run: npm install -g npm@11.2.0
npm notice
```

Vite+Netlify 프로젝트 세팅하기(계속)

- 아래 명령어를 차례대로 입력(한 줄씩 입력 후 엔터)
 - `cd smwu-static-test`(내 프로젝트 이름)
 - `npm install`
 - `npm run dev`
- localhost 주소 접속하여 사이트 확인

```
PS C:\Users\eduwa\OneDrive\문서\GitHub\smwu-static-test> npm run dev

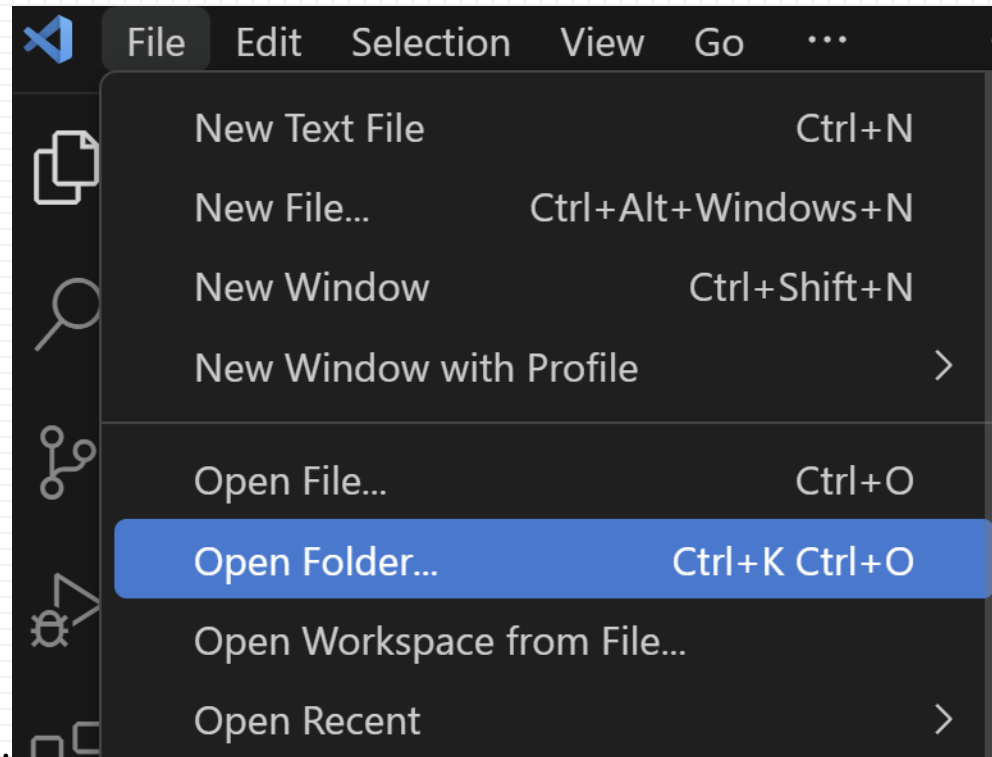
> smwu-static-test@0.0.0 dev
> vite

VITE v6.2.1 ready in 165 ms

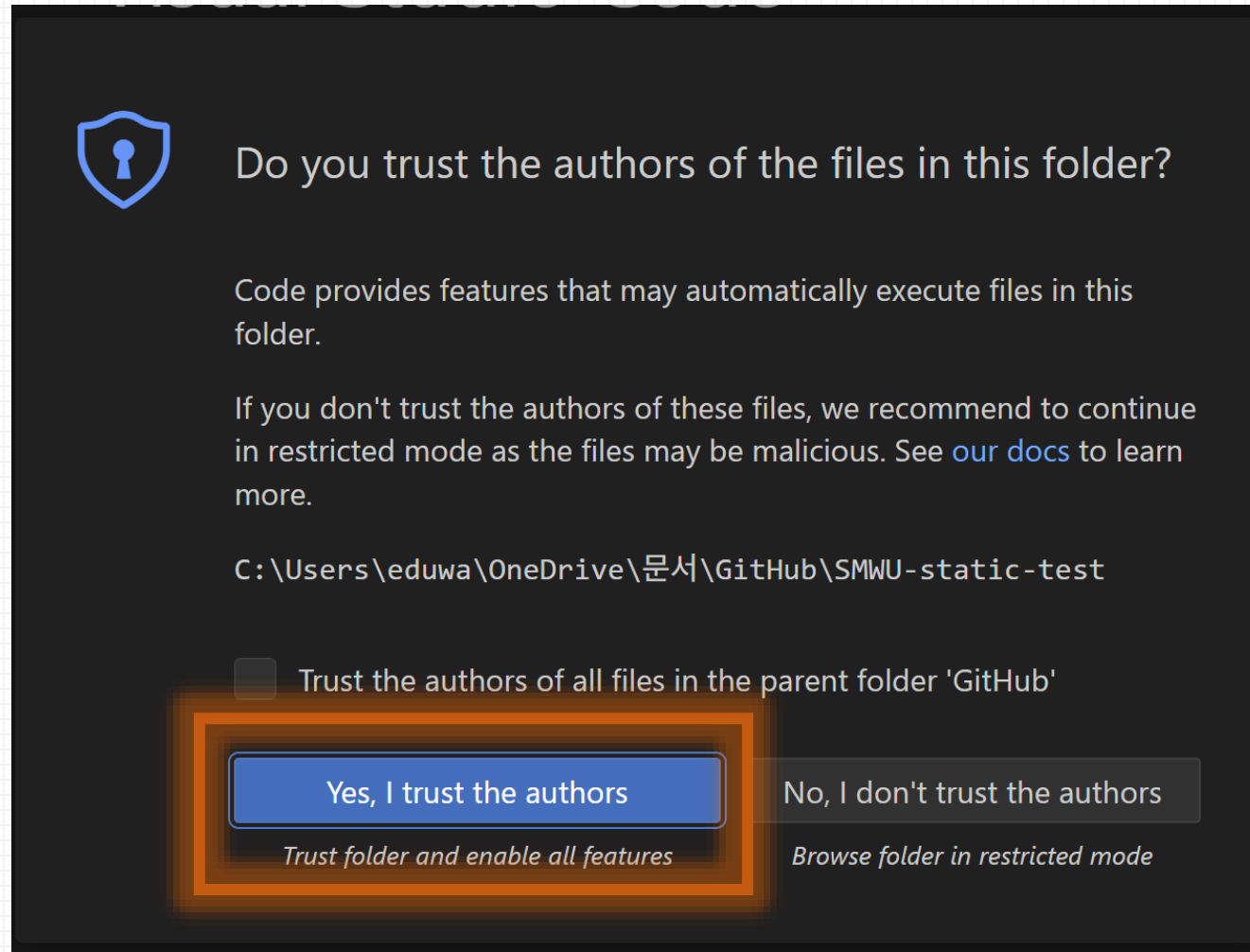
→ Local:   http://localhost:5173/
→ Network: use --host to expose
→ press h + enter to show help
```


Vite+Netlify 프로젝트 세팅하기(계속)

- VS Code 실행한 후
[File] - [Open Folder] 클릭하여 만든 폴더 열기

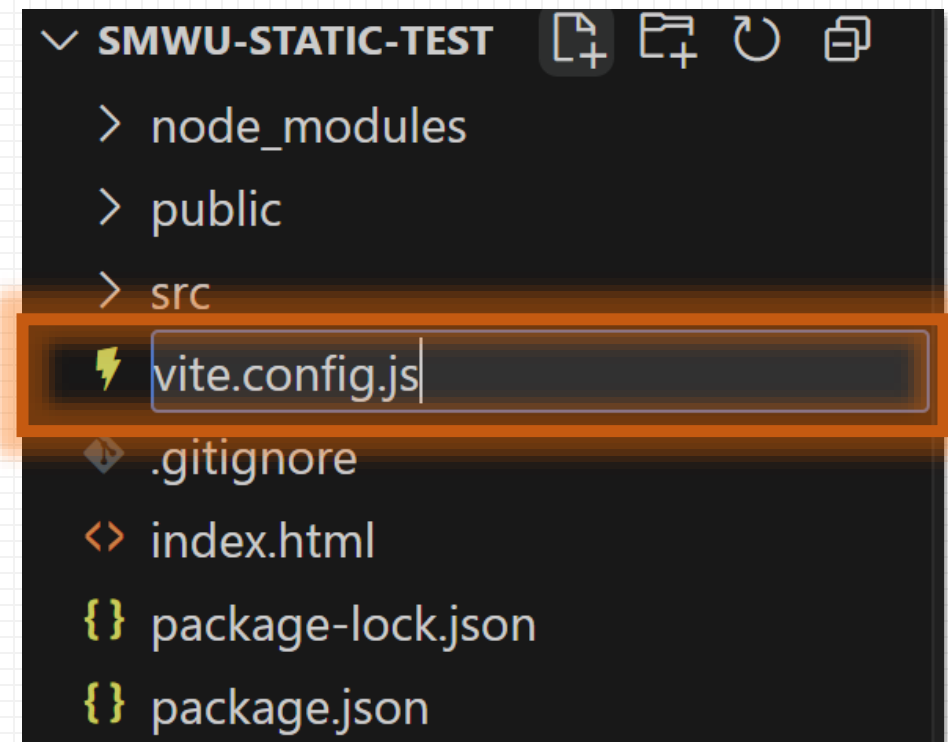
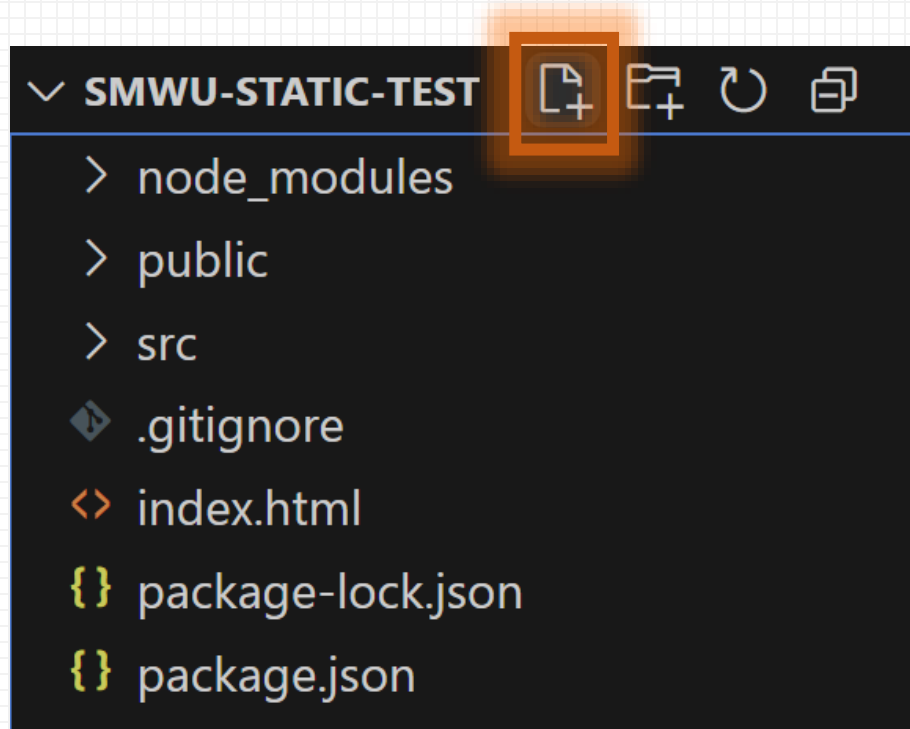


Vite+Netlify 프로젝트 세팅하기(계속)



Vite+Netlify 프로젝트 세팅하기(계속)

- 폴더에 'new file' 클릭하여 vite.config.js 생성
 - vite.config.js는 추후에 페이지를 추가할 경우에 필요함



Vite+Netlify 프로젝트 세팅하기(계속)

- vite.config.js에 아래 코드 입력

```
// vite.config.js
import { resolve } from 'path'
import { defineConfig } from 'vite'

export default defineConfig({
  build: {
    rollupOptions: {
      input: {
        main: resolve(__dirname, 'index.html'),
      },
    },
  },
})
```

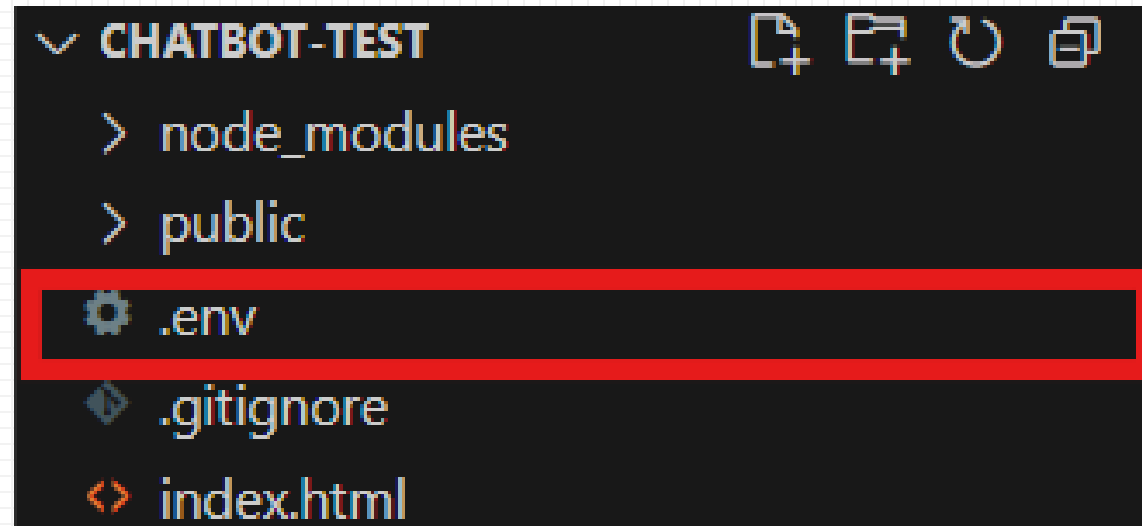
Vite+Netlify 프로젝트 세팅하기(계속)

- 기본으로 만들어지는 src 폴더는 삭제해도 됨

지금부터는 2주차 강의와 조금 달라집니다.

로컬에서 환경변수 설정하기

- 프로젝트 최상단에서 'New File...' 클릭하여 .env 파일 생성
 - 파일 이름이 '.env'
 - 이 파일이 환경변수를 저장할 장소



로컬에서 환경변수 설정하기(계속)

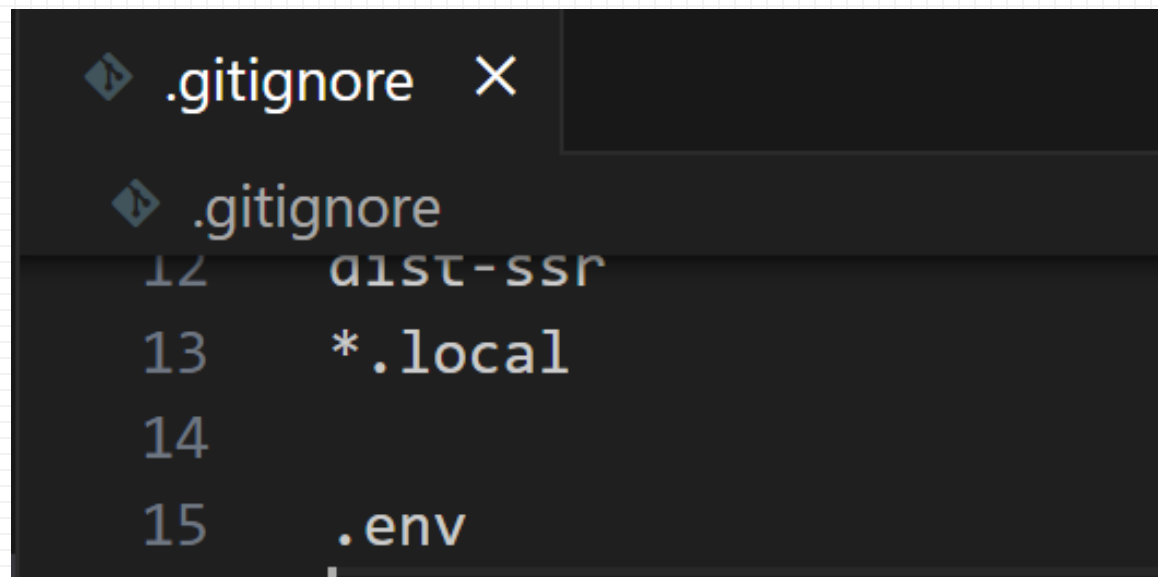
- .env 파일에 다음과 같이 본인의 API Key 입력
 - VITE_OPENAI_API_KEY = sk-xxxxxxxxxxxxxxxxxxxxxxxx
 - 따옴표 필요 없음

⚙ .env

```
1 VITE_OPENAI_API_KEY=sk-xxxxxxxxxxxxxxxxxxxxxxxx
```


로컬에서 환경변수 설정하기(계속)

- .gitignore에 .env 파일 추가(보안을 위해, 위치는 상관없음)
 - .gitignore는 Git이 추적하거나 업로드하지 말아야 할 파일이나 폴더를 적어두는 목록

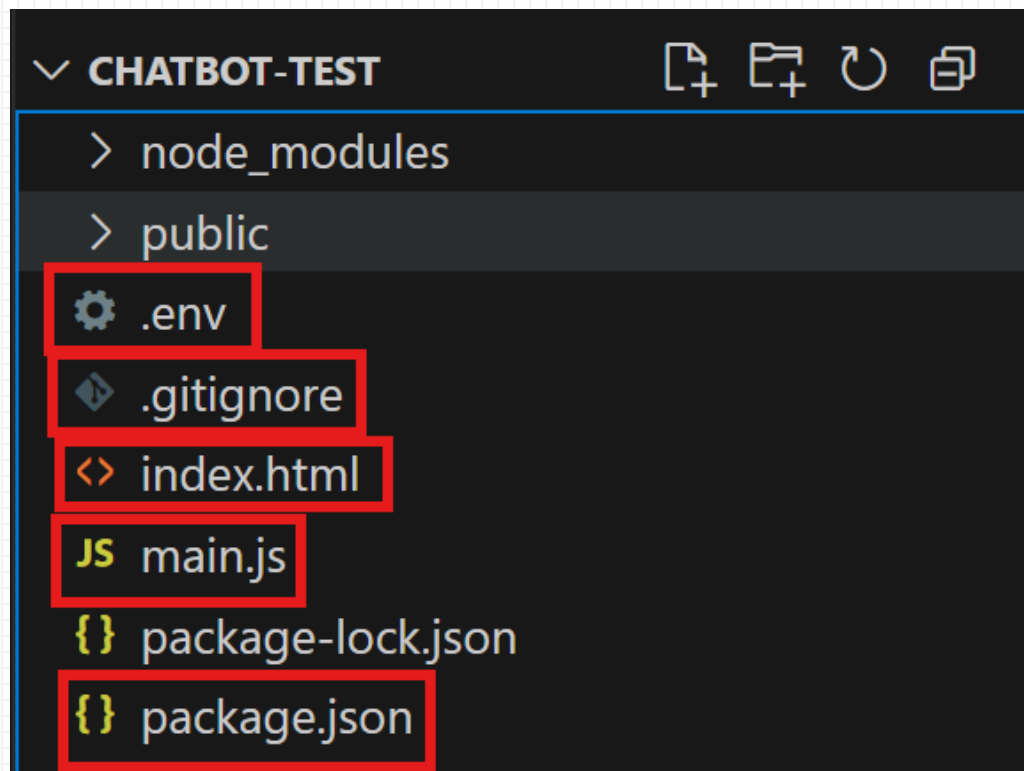


The screenshot shows a code editor with a dark theme. At the top, there's a tab labeled '.gitignore' with a close button (X). Below the tab, the file content is displayed with line numbers on the left. The content of the file is:

```
12 dist-ssr
13 *.local
14
15 .env
```

로컬에서 환경변수 설정하기(계속)

- 지금까지 잘 따라오셨다면, 프로젝트 폴더는 다음의 파일을 다음의 구조로 가지고 있어야 함
- chatbot-app(프로젝트 이름)
 - ├── index.html
 - ├── .env
 - ├── .gitignore
 - ├── package.json
 - └── main.js



간단한 Chatbot 만들기

- index.html 파일과 main.js 파일을 첨부한 파일의 코드로 수정
- 파일 자체를 교체해도 됨

간단한 Chatbot 만들기(계속)

- index.html

```
<!DOCTYPE html>
<html lang="ko">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>간단한 GPT 챗봇</title>
  <script type="module" src="/main.js"></script>
  <link href="https://cdn.jsdelivr.net/npm/tailwindcss@2.2.19/dist/tailwind.min.css" rel="stylesheet">
</head>
<body class="bg-gray-100 p-4 flex flex-col items-center justify-center h-screen">
  <div class="bg-white shadow-lg rounded-lg w-full max-w-lg p-4">
    <h1 class="text-xl font-bold mb-4">GPT 간단 챗봇</h1>
    <div id="chatbox" class="h-64 overflow-auto bg-gray-50 p-2 rounded mb-2"></div>
    <input type="text" id="userInput" placeholder="질문을 입력하세요..." class="border p-2 w-full rounded mb-2">
    <button id="sendBtn" class="bg-blue-500 text-white rounded p-2 w-full">질문하기</button>
  </div>
</body>
</html>
```

간단한 Chatbot 만들기(계속)

- main.js

```
const apiKey = import.meta.env.VITE_OPENAI_API_KEY;
const chatbox = document.getElementById('chatbox');
const userInput = document.getElementById('userInput');
const sendBtn = document.getElementById('sendBtn');

async function fetchGPTResponse(prompt) {
  const response = await fetch("https://api.openai.com/v1/chat/completions", {
    method: "POST",
    headers: {
      "Content-Type": "application/json",
      "Authorization": `Bearer ${apiKey}`
    },
    body: JSON.stringify({
      model: "gpt-3.5-turbo",
      messages: [{"role": "user", "content": prompt}],
      temperature: 0.7,
    })
  });
  const data = await response.json();
  return data.choices[0].message.content;
}

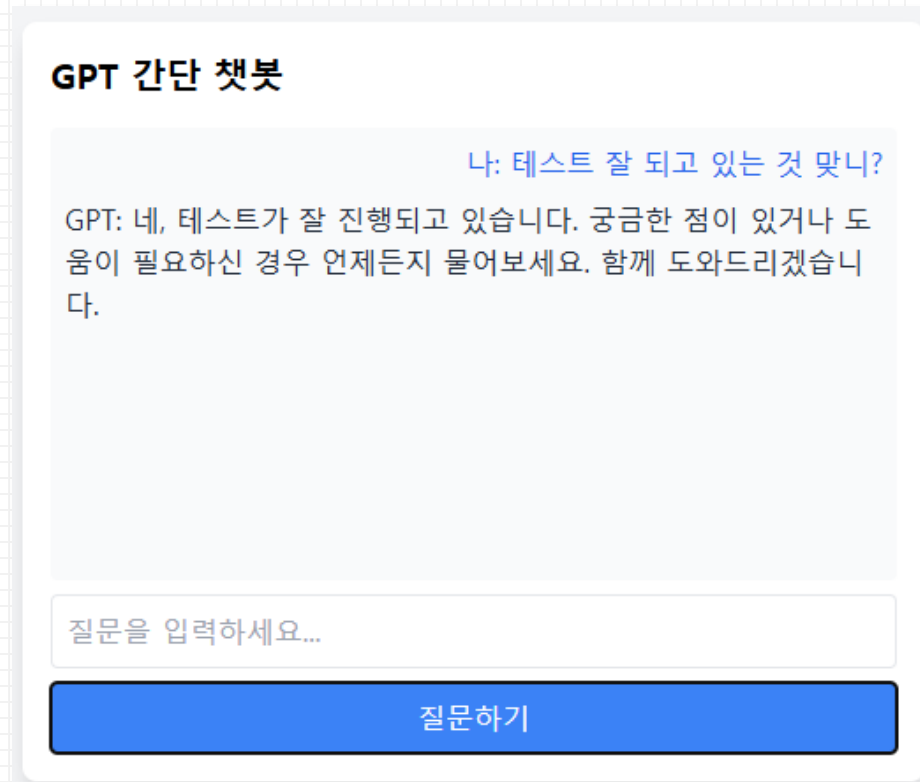
sendBtn.addEventListener('click', async () => {
  const prompt = userInput.value;
  if (!prompt) return;

  chatbox.innerHTML += `<div class="text-right mb-2 text-blue-600">나: ${prompt}</div>`;
  userInput.value = '';
  chatbox.scrollTop = chatbox.scrollHeight;

  const reply = await fetchGPTResponse(prompt);
  chatbox.innerHTML += `<div class="text-left mb-2 text-gray-800">GPT: ${reply}</div>`;
  chatbox.scrollTop = chatbox.scrollHeight;
});
```

간단한 Chatbot 만들기(계속)

- 터미널 열어 `npm run dev` 실행하여 테스트 해보기



Netlify 배포 및 환경변수 추가

- npm run build를 하여 이 프로젝트를 빌드하고, 이 프로젝트 폴더를 GitHub에 Publish함
- Netlify에서 배포하는 과정은 이전과 동일하지만, 마지막 환경변수 설정 부분이 달라짐

Netlify 배포 및 환경변수 추가(계속)

- 마지막 [Deploy...] 전에 Environment variables에서 [New variable] 클릭

Environment variables

Define environment variables for more control and flexibility over your build.

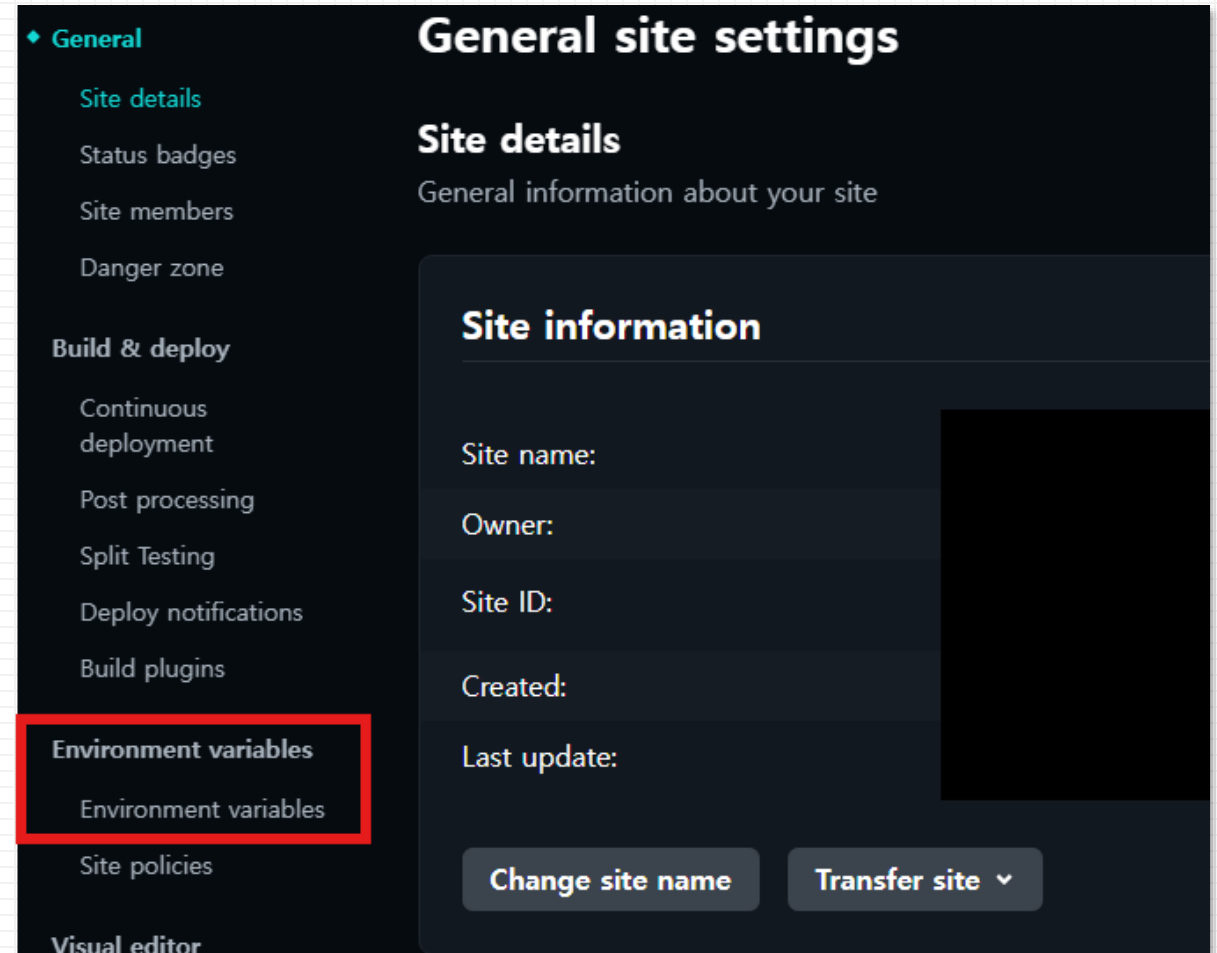
New variable

Netlify 배포 및 환경변수 추가(계속)

- Key → VITE_OPENAI_API_KEY
- Value → 내 API Key 입력(sk-proj-xxxxxx...)
- 위의 정보 입력 후 [Deploy...] 클릭
- 간단한 Chatbot 페이지 완성!

Netlify 배포 및 환경변수 추가(계속)

- 이미 배포한 Web-app이라면?
- Netlify 대시보드에서 내가 배포한 Web-app 클릭하여 Site configuration 클릭
- 좌측 메뉴에서 Environment variables 클릭



Netlify 배포 및 환경변수 추가(계속)

- Add a variable 클릭하여
- Key: → VITE_OPENAI_API_KEY
- Values → 내 API Key 입력(sk-proj-xxxxx...)
- 이때 Contains secret values는
- 해당 값이 내 대시보드에서도 보이지 않게 만들고, 다른 팀원들도 볼 수 없게 만드는 역할을 함

Netlify 배포 및 환경변수 추가(계속)

New environment variable

Key:

EXAMPLE_KEY

Secret:

☐ **Contains secret values**
Secret values are only readable by code running on Netlify's systems. With secrets, only the local development context values are readable and unmasked on Netlify's UI, API, and CLI.

Scopes:

☒ **All scopes**

☐ **Specific scopes**
Limit this environment variable to specific scopes, such as builds, functions, or post processing
[Upgrade to unlock](#)

Values:

☒ **Same value for all deploy contexts**

☐ **Different value for each deploy context**
Use different environment variable values for production, Deploy Previews, branch deploys, and local development. Optionally override these values on specific branches.

Create variable

Cancel

API Key 완벽하게 보호하고 싶다면...

- [구글 문서](#)에서 프록시 설정 부분 확인
- 다만, 이 방법은 Netlify 응답 시간 제한으로 인해 모델 사용에 제한이 있음

중간 발표 안내

중간 발표 안내

- 중간 발표 일정: 4월 28일(8주차) 온라인 비대면
- 1인당 10분 이내
- 발표 내용
 - 계획 중인 Web-app의 기본 틀
 - 기술 명세서

중간 발표 안내

- 계획 중인 Web-app의 기본 틀
 - Web-app의 페이지 기본 구성의 초안을 완성해 주시면 됩니다.
 - 구체적으로 index.html 파일과 style.css 파일을 작성하셔서 버튼이나 입력창 등을 배치한 형태를 만들어 주시면 됩니다.
 - 발표 시에는 web-app 주소에 접속하여 저를 비롯하여 다른 분들이 확인할 수 있게 만들어 주시면 됩니다.

중간 발표 안내

- 기술 명세서

- 본격적인 개발에 앞서 기술 명세서를 상세하게 작성해 주시면 됩니다.
- 기술 명세서는 다음의 순서와 내용을 따라 작성해 주시면 됩니다.

1. Web-app 이름 / 수업 주제 → 어떤 수업을 위한 앱인가?
2. Web-app의 목적 → 수업에서 이 앱이 하는 역할은 무엇인가?
3. Web-app의 기능 → 앱에 포함될 주요 기능은 무엇인가?
4. Web-app 사용 흐름도 → 수업 중 앱 사용이 이루어지는 과정은?
5. 사용할 도구 → 어떤 도구를 사용할 것인가?

중간 발표 안내

1. Web-app 이름 / 수업 주제
 - 예) 도형의 전개도 그리기
2. Web-app의 목적
 - 예) 학생의 주도적인 학습 목적
3. Web-app의 기능
 - 예) 3차원 모형 제시, 학생의 입력 시각화 등
4. Web-app 사용 흐름도
 - 예) 시작 화면 → 학생 입력 → 시각화 → 학생 분석 입력 → 분석 결과 제시
5. 사용할 도구
 - 예) Vite, Netlify, ChatGPT API, Sigma JS 등

고생하셨습니다.