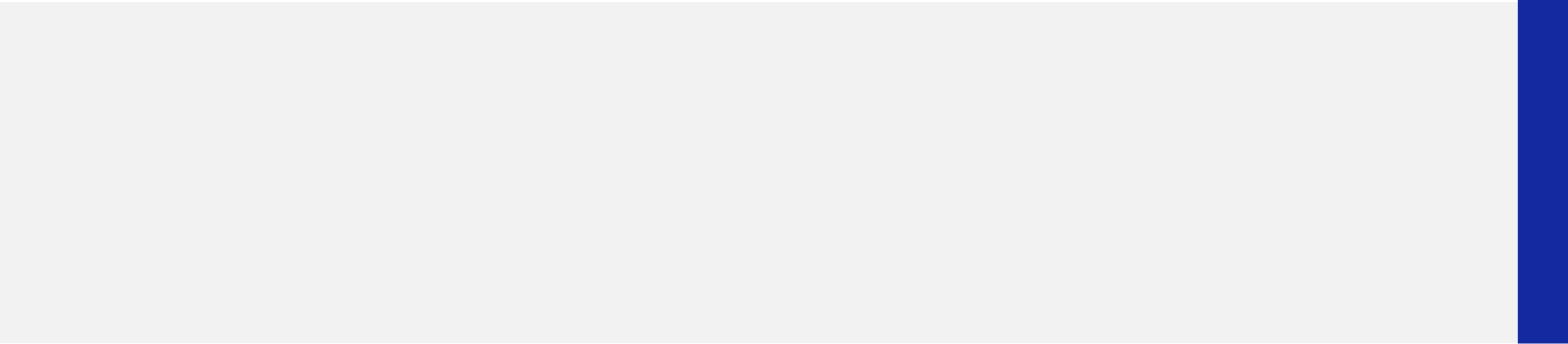
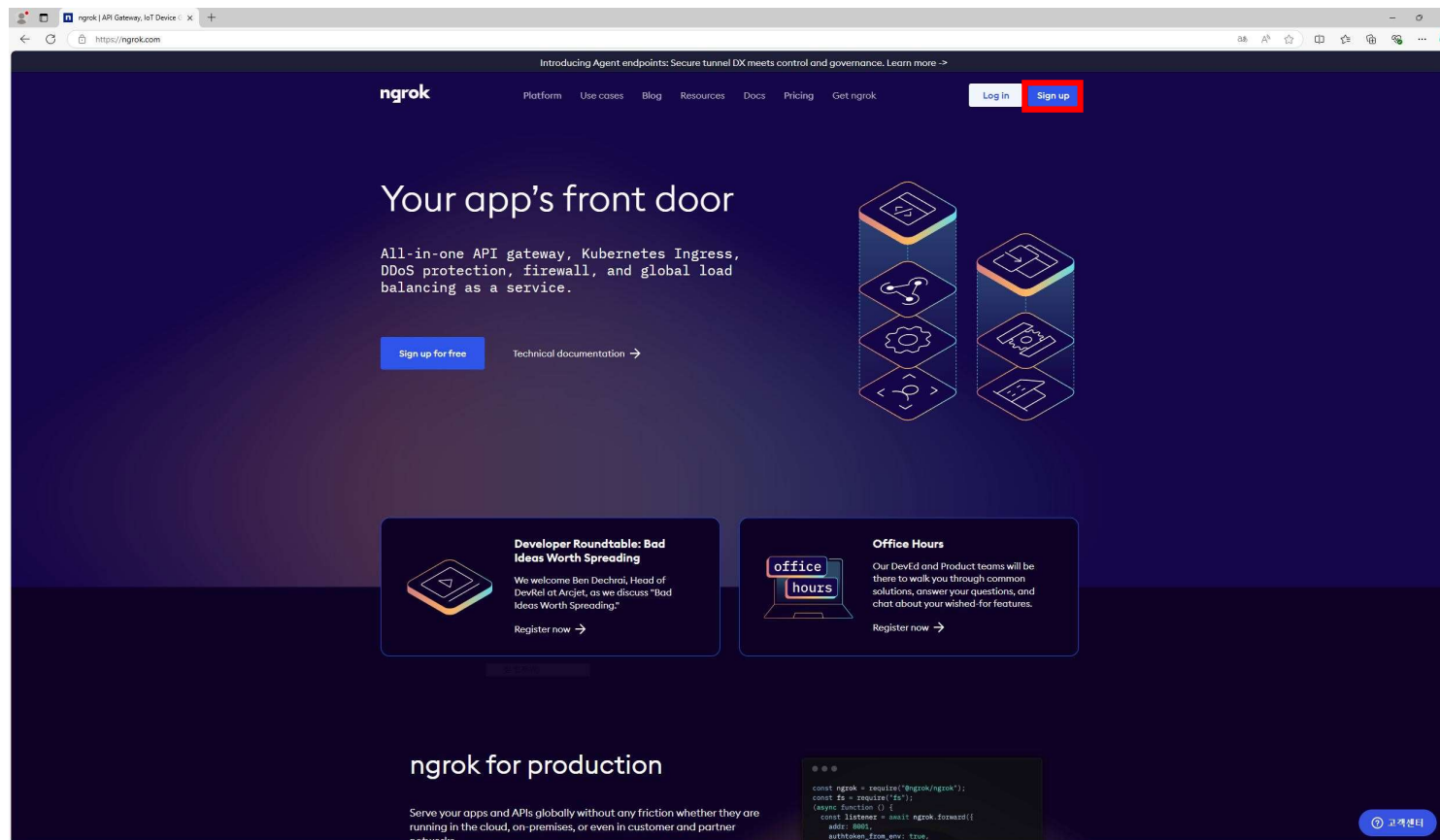


AI Essential

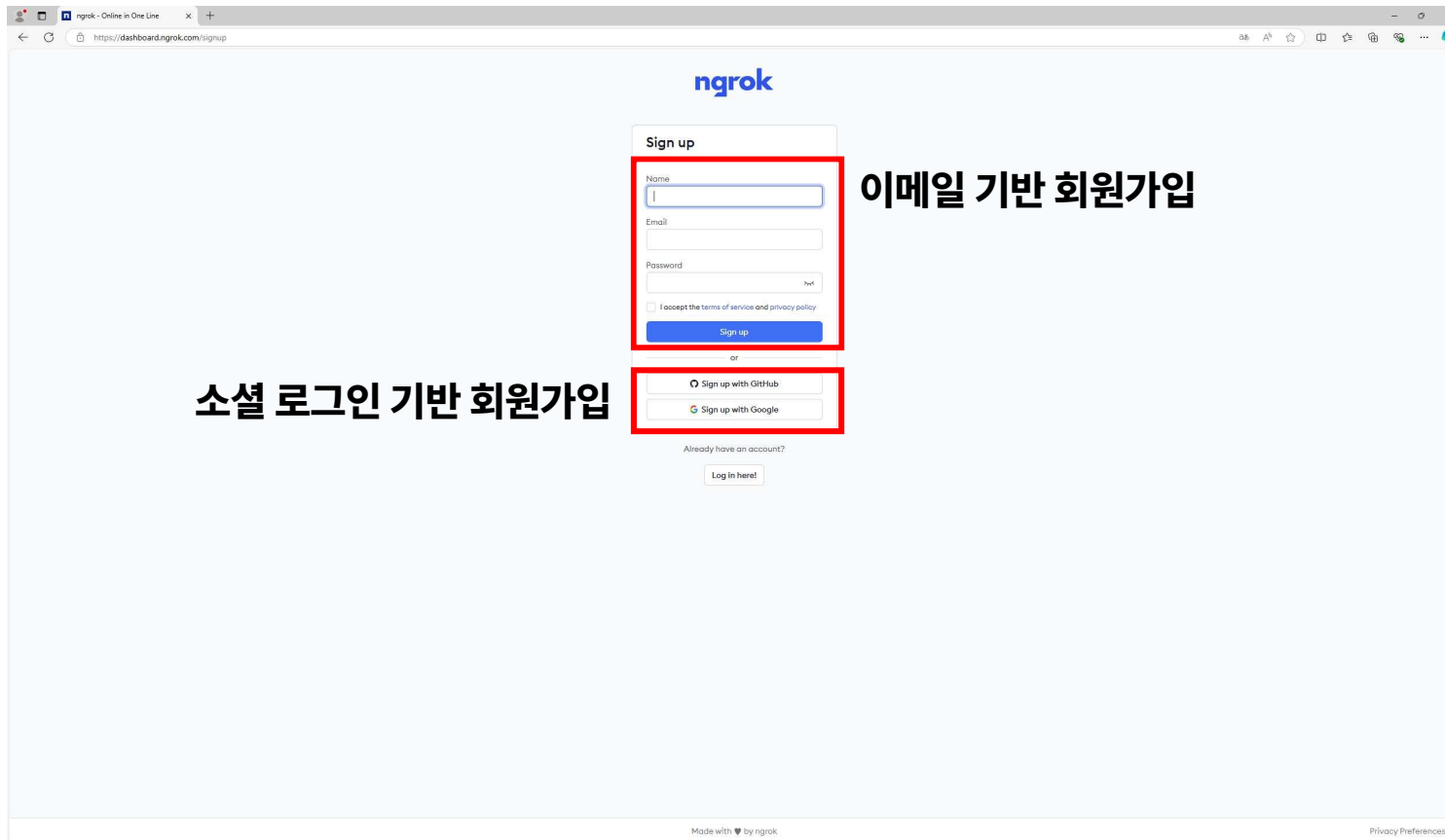
API KEY 발급 가이드 (상세)



➤ ngrok 홈페이지(<https://ngrok.com>) 접속



- 이메일 기반 또는 소셜로그인(gitlab, google) 기반 회원 가입 진행



이메일 기반 회원가입

소셜 로그인 기반 회원가입

ngrok

Sign up

Name

Email

Password

☐ I accept the terms of service and privacy policy

Sign up

or

Sign up with GitHub

Sign up with Google

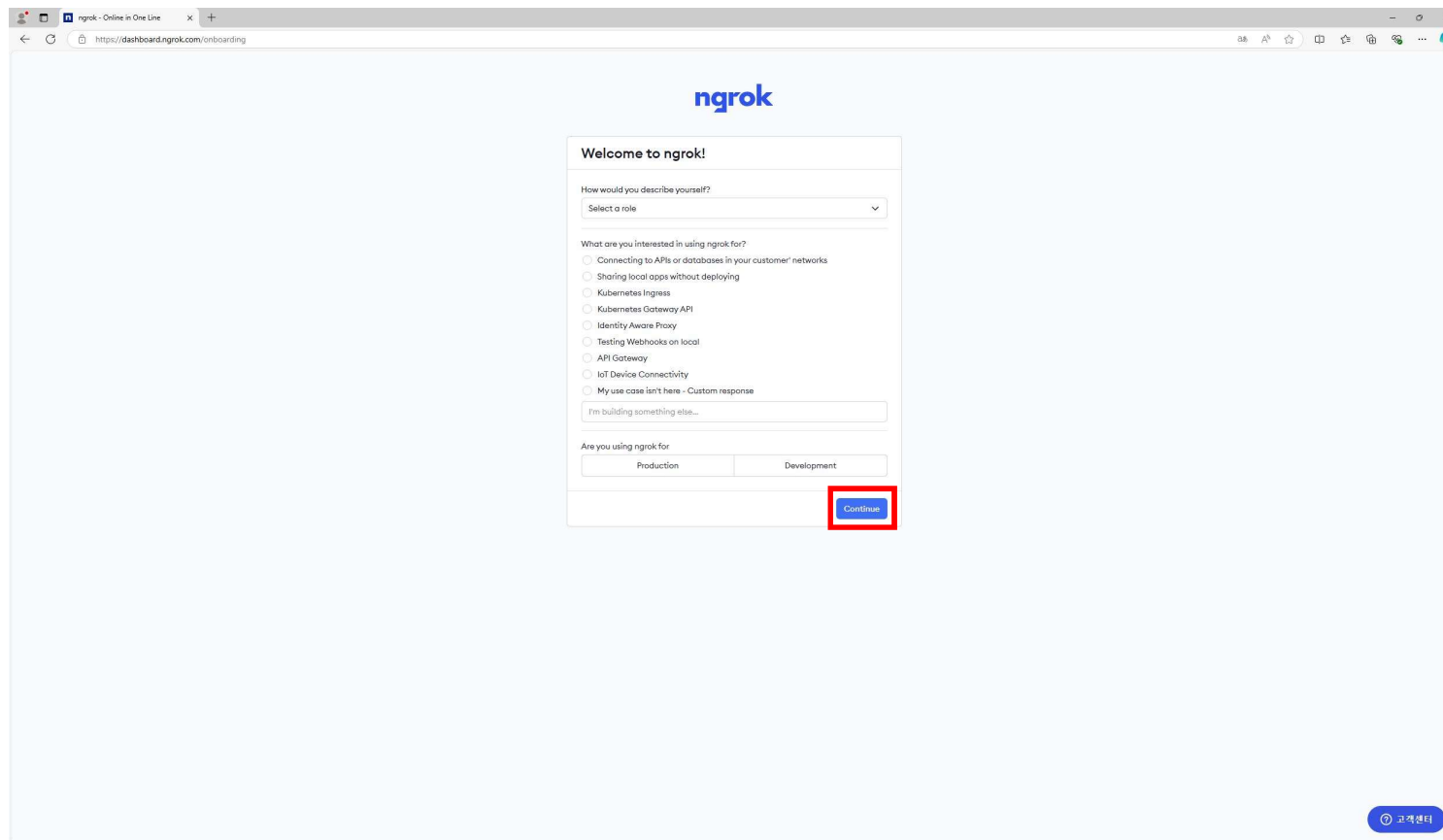
Already have an account?

Log in here!

Made with ♥ by ngrok

Privacy Preferences

- 회원가입 이후 온보딩 설문조사의 경우 Continue를 클릭하여 생략가능

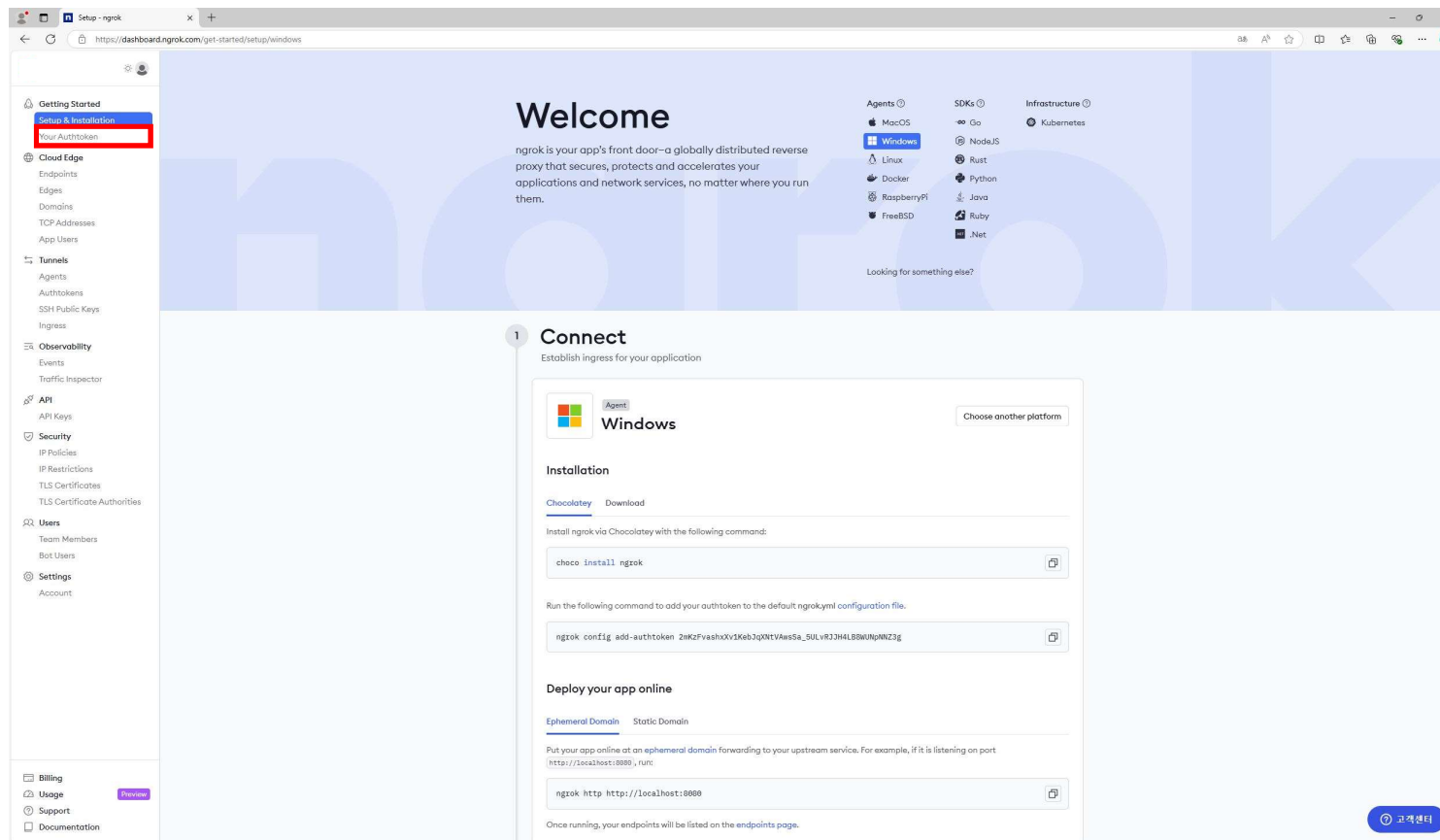


The screenshot shows the ngrok onboarding page in a web browser. The page has a light blue background with the ngrok logo at the top. A white form is centered on the page with the title "Welcome to ngrok!". The form contains the following sections:

- How would you describe yourself?**: A dropdown menu with "Select a role" as the placeholder.
- What are you interested in using ngrok for?**: A list of radio button options:
 - ☐ Connecting to APIs or databases in your customer's networks
 - ☐ Sharing local apps without deploying
 - ☐ Kubernetes Ingress
 - ☐ Kubernetes Gateway API
 - ☐ Identity Aware Proxy
 - ☐ Testing Webhooks on local
 - ☐ API Gateway
 - ☐ IoT Device Connectivity
 - ☐ My use case isn't here - Custom response
- I'm building something else...
- Are you using ngrok for**: Two radio button options: "Production" and "Development".
- Continue**: A blue button at the bottom right of the form, highlighted with a red rectangle.

In the bottom right corner of the page, there is a small blue button with a question mark icon and the text "고객센터" (Customer Center).

- 로그인 화면에서 좌측 **Your Authtoken** 클릭 (API KEY 메뉴가 아님을 주의)

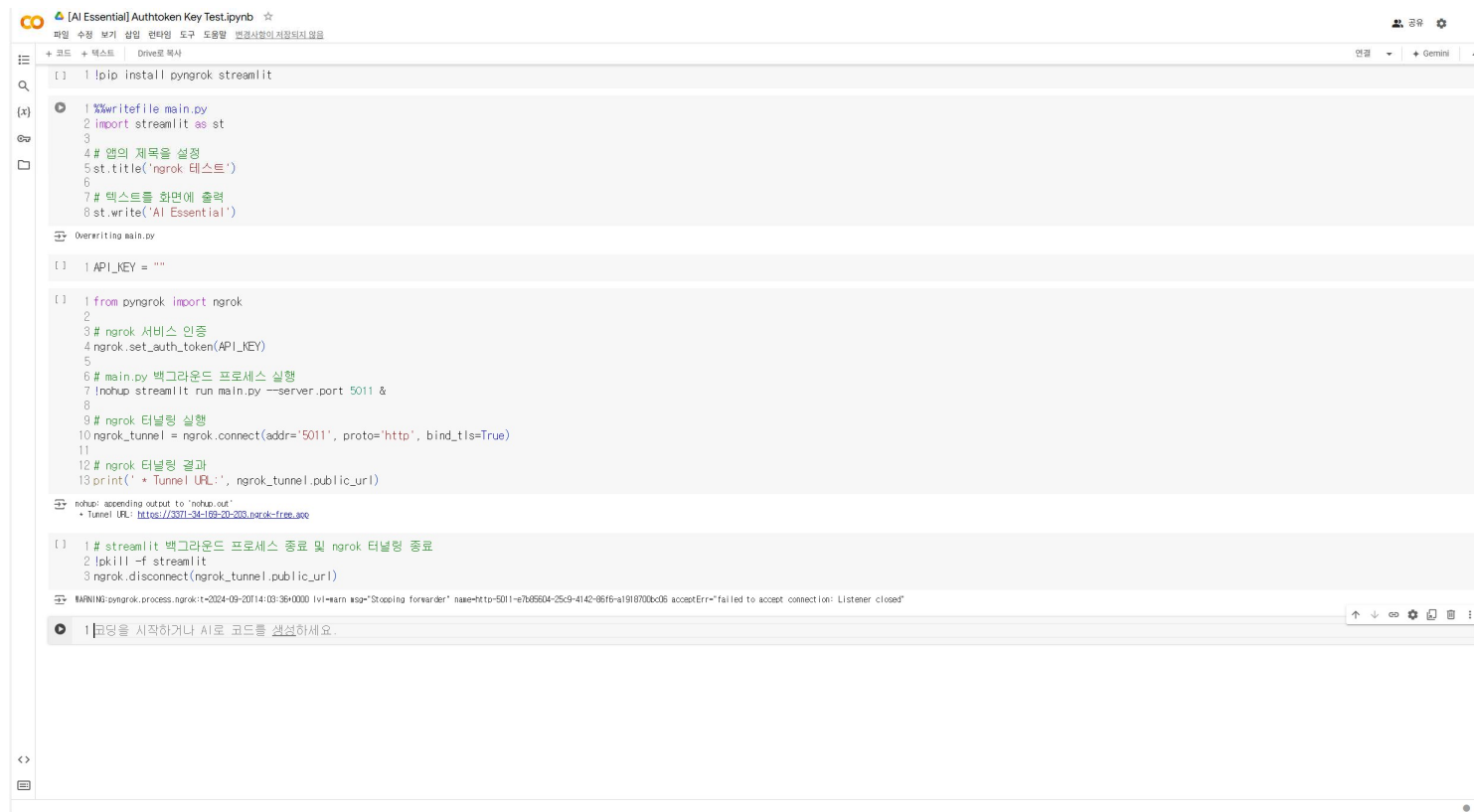


➤ Authtoken Key 복사

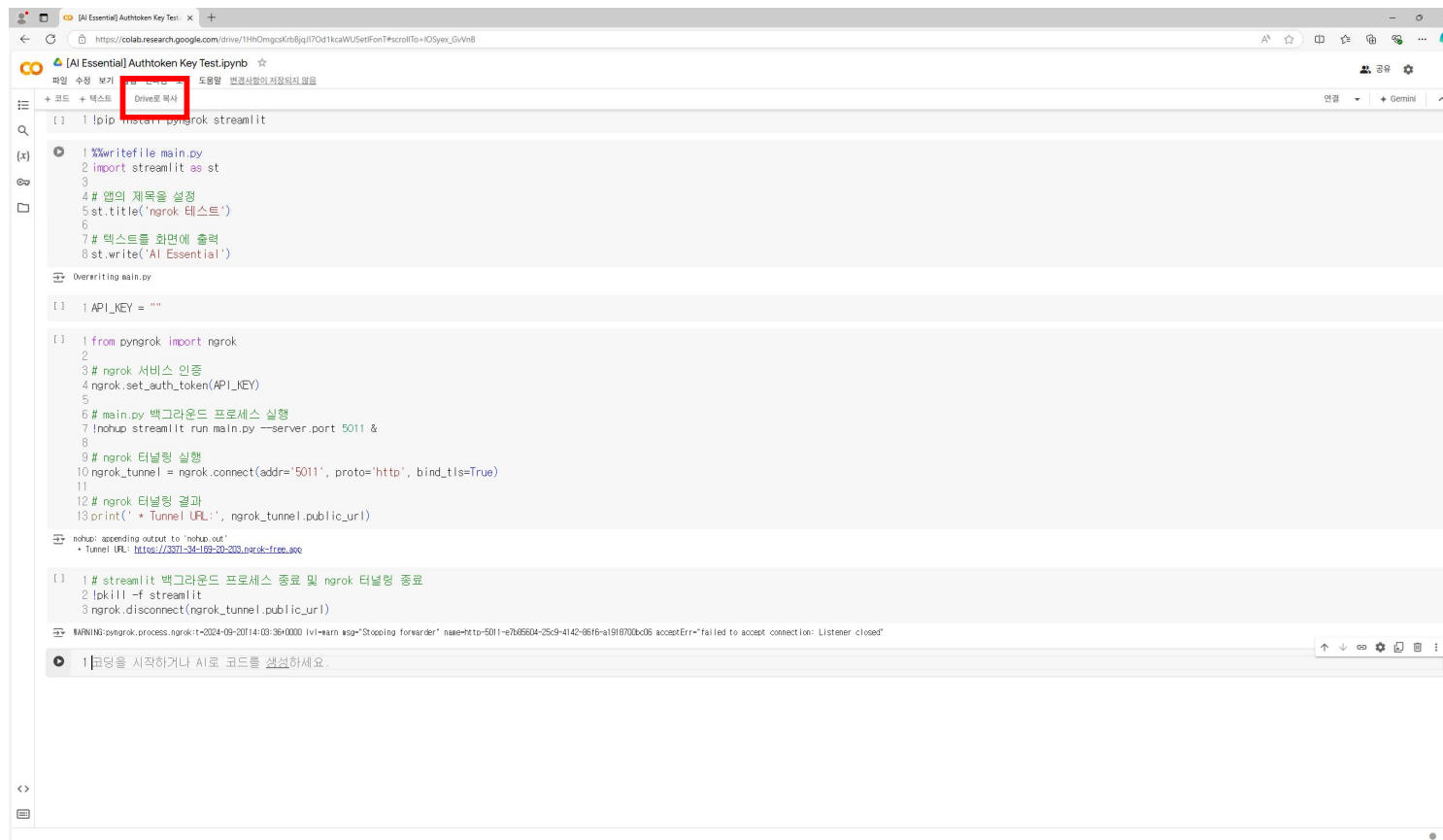
The screenshot displays the 'Your Authtoken' page on the Ngrok dashboard. The page is divided into a sidebar on the left and a main content area. The sidebar contains navigation links for 'Getting Started', 'Cloud Edge', 'Tunnels', 'Observability', 'API', 'Security', 'Users', 'Settings', 'Billing', 'Usage', 'Support', and 'Documentation'. The main content area has a light blue header with the title 'Your Authtoken'. Below the header, a message states: 'This is your personal Authtoken. Use this to authenticate the ngrok agent that you downloaded.' A red box highlights the generated auth token, which is a long string of characters. To the right of the token is a 'Copy' button. Below the token, there are three sections: 'Command Line', 'Configuration File', and 'Reset Your Authtoken'. The 'Command Line' section explains that the token is saved in the default configuration file and provides a terminal snippet showing the command 'ngrok config add-authtoken \$YOUR_AUTHTOKEN'. The 'Configuration File' section explains that the token can be added to a configuration file and provides a terminal snippet showing the command 'ngrok.yml' and the configuration 'authtoken: <your-authtoken>'. The 'Reset Your Authtoken' section states that the token cannot be undone and provides a 'Reset Authtoken' button.

➤ Authtoken Key 테스트를 위한 Google Colab Notebook 접속

<https://colab.research.google.com/drive/1HhOmgcsKrbBjqJI7Od1kcaWU5etIFonT?usp=sharing>



➤ 본인 계정의 Drive로 복사 (로그인 필요)



```
[ ] 1 !pip install pyngrok streamlit

[ ] 2 %writefile main.py
3 2 import streamlit as st
4 3
5 4 # 앱의 제목을 설정
6 5 st.title('ngrok 테스트')
7 6
8 7 # 텍스트를 화면에 출력
9 8 st.write('AI Essential')

[ ] 9 Overwriting main.py

[ ] 10 API_KEY = ""

[ ] 11 from pyngrok import ngrok
12 2
13 3 # ngrok 서비스 인증
14 4 ngrok.set_auth_token(API_KEY)
15 5
16 6 # main.py 백그라운드 프로세스 실행
17 7 !nohup streamlit run main.py --server.port 5011 &
18 8
19 9 # ngrok 터널링 실행
20 10 ngrok_tunnel = ngrok.connect(addr='5011', proto='http', bind_tls=True)
21 11
22 12 # ngrok 터널링 결과
23 13 print(' * Tunnel URL: ', ngrok_tunnel.public_url)

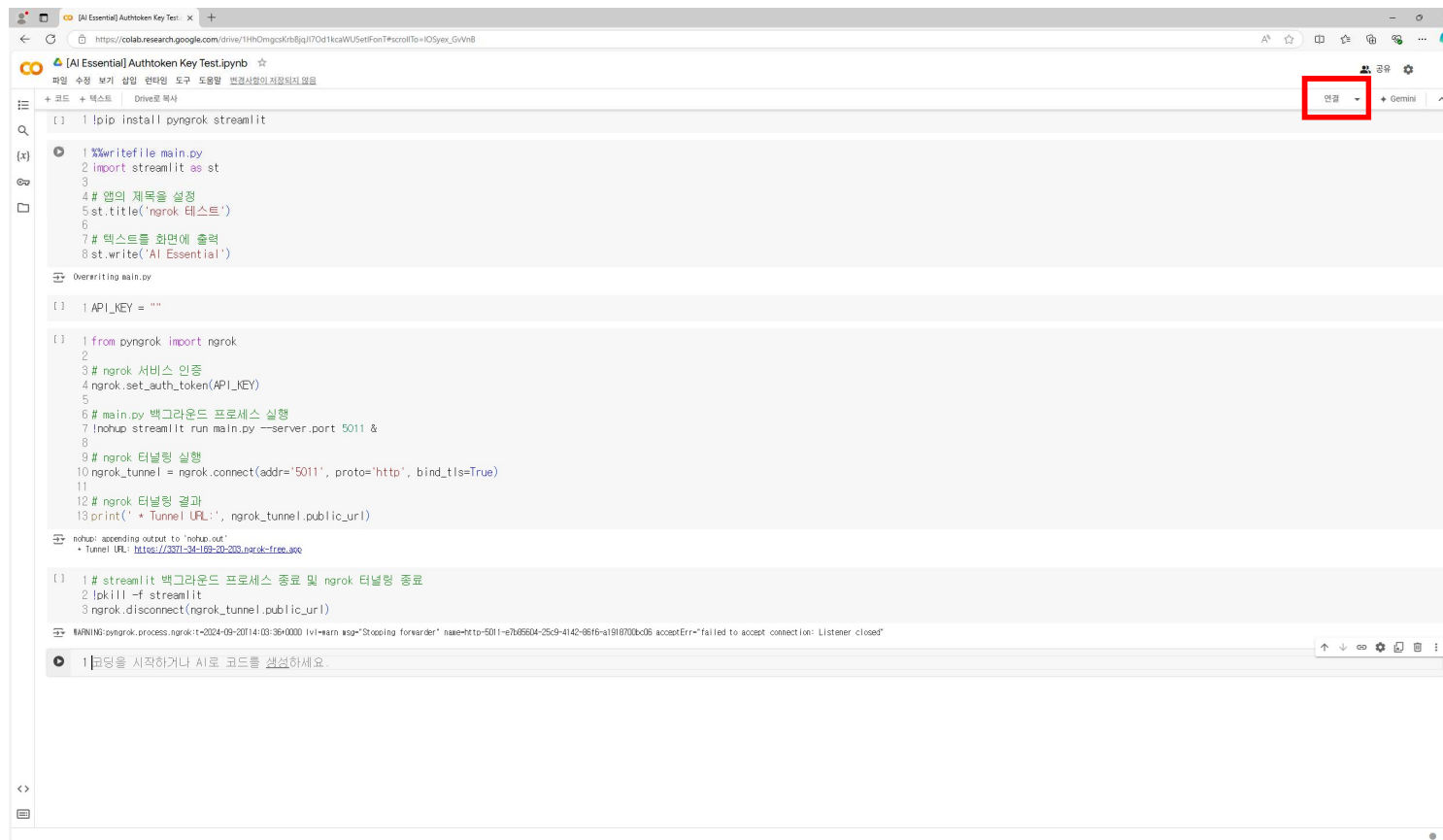
[ ] 24 nohup: appending output to 'nohup.out'
 * Tunnel URL: https://3071-34-169-20-203.ngrok-free.app

[ ] 25 1 # streamlit 백그라운드 프로세스 종료 및 ngrok 터널링 종료
2 2 !kill -f streamlit
3 3 ngrok.disconnect(ngrok_tunnel.public_url)

[ ] 26 WARNING:pyngrok.process.ngrok:t-2024-09-20T14:03:36+0000 [v1=warn,msg="Stopping forwarder" name=http-5011-e766504-25d9-4142-95f6-a1918700bc06 acceptErr="failed to accept connection: Listener closed"]

[ ] 27 1 코딩을 시작하거나 AI로 코드를 생성하세요.
```


➤ 연결 버튼을 클릭하여 Python 런타임 연결



```
[ ] 1 !pip install pyngrok streamlit

[ ] 2 %writefile main.py
3 2 import streamlit as st
4 3
5 4 # 앱의 제목을 설정
6 5 st.title('ngrok 테스트')
7 6
8 7 # 텍스트를 화면에 출력
9 8 st.write('AI Essential')

Overriding main.py

[ ] 1 API_KEY = ""

[ ] 1 from pyngrok import ngrok
2 2
3 3 # ngrok 서비스 인증
4 4 ngrok.set_auth_token(API_KEY)
5 5
6 6 # main.py 백그라운드 프로세스 실행
7 7 !nohup streamlit run main.py --server.port 5011 &
8 8
9 9 # ngrok 터널링 실행
10 10 ngrok_tunnel = ngrok.connect(addr='5011', proto='http', bind_tls=True)
11 11
12 12 # ngrok 터널링 결과
13 13 print(' * Tunnel URL: ', ngrok_tunnel.public_url)

nohup: appending output to 'nohup.out'
 * Tunnel URL: 'https://3071-34-169-20-203.ngrok-free.app'

[ ] 1 # streamlit 백그라운드 프로세스 종료 및 ngrok 터널링 종료
2 2 !kill -f streamlit
3 3 ngrok.disconnect(ngrok_tunnel.public_url)

WARNING:pyngrok.process.ngrok:t-2024-09-20T14:03:36+0000 [vi=main nsg="Stopping forwarder" name=http-5011-e766504-25d9-4142-95f6-a1918700bc06 acceptErr="failed to accept connection: Listener closed"]

[ ] 1 [요청을 시작하거나 AI로 코드를 생성하세요.]
```

➤ pyngrok, streamlit 패키지 설치 코드 실행

```
[1] !pip install pyngrok streamlit

[2] %writefile main.py
1 import streamlit as st
2
3
4 # 앱의 제목을 설정
5 st.title('ngrok 테스트')
6
7 # 텍스트를 화면에 출력
8 st.write('AI Essential')

Overriding main.py

[1] API_KEY = ""

[2] 1 from pyngrok import ngrok
2
3 # ngrok 서비스 인증
4 ngrok.set_auth_token(API_KEY)
5
6 # main.py 백그라운드 프로세스 실행
7 !nohup streamlit run main.py --server.port 5011 &
8
9 # ngrok 터널링 실행
10 ngrok_tunnel = ngrok.connect(addr='5011', proto='http', bind_tls=True)
11
12 # ngrok 터널링 결과
13 print(' * Tunnel URL: ', ngrok_tunnel.public_url)

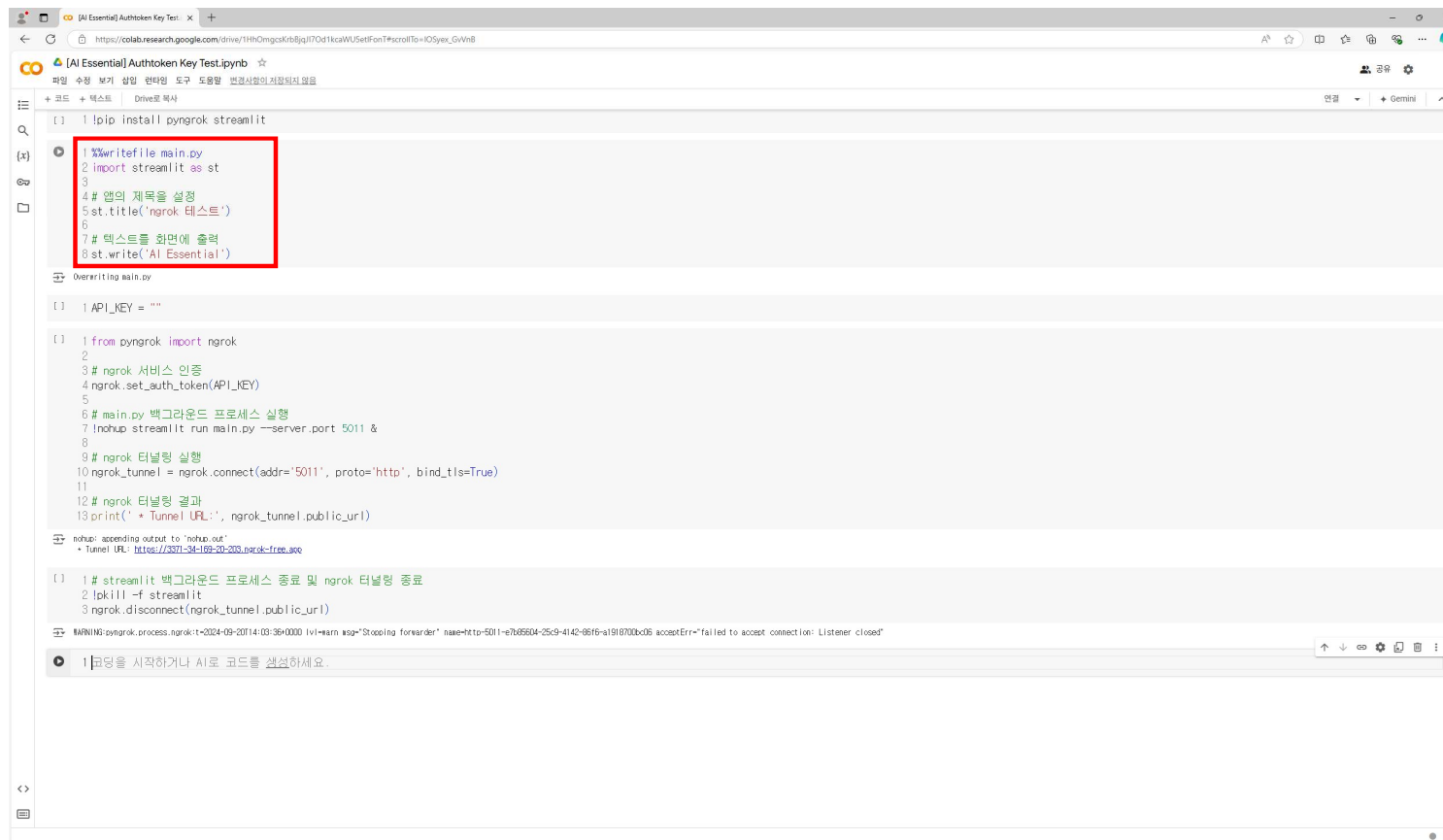
nohup: appending output to 'nohup.out'
 * Tunnel URL: 'https://3071-34-169-20-203.ngrok-free.app'

[3] 1 # streamlit 백그라운드 프로세스 종료 및 ngrok 터널링 종료
2 !kill -f streamlit
3 ngrok.disconnect(ngrok_tunnel.public_url)

WARNING: pyngrok.process.ngrok:1-2024-09-20T14:03:36+0000 [vi=main asp="Stopping forwarder" name=http-5011-e766504-25d9-4142-95f6-a1918700c06 acceptErr="failed to accept connection: Listener closed"]

[4] 1 코딩을 시작하거나 AI로 코드를 생성하세요.
```

➤ Streamlit 서비스 파일 생성 코드 실행



```
[ ] 1 !pip install pyngrok streamlit

[ ] 2 %writefile main.py
3 import streamlit as st
4
5 # 앱의 제목을 설정
6 st.title('ngrok 테스트')
7
8 # 텍스트를 화면에 출력
9 st.write('AI Essential')

Overwriting main.py

[ ] 1 API_KEY = ""

[ ] 1 from pyngrok import ngrok
2
3 # ngrok 서비스 인증
4 ngrok.set_auth_token(API_KEY)
5
6 # main.py 백그라운드 프로세스 실행
7 !nohup streamlit run main.py --server.port 5011 &
8
9 # ngrok 터널링 실행
10 ngrok_tunnel = ngrok.connect(addr='5011', proto='http', bind_tls=True)
11
12 # ngrok 터널링 결과
13 print(' * Tunnel URL: ', ngrok_tunnel.public_url)

nohup: appending output to 'nohup.out'
 * Tunnel URL: 'https://3071-34-199-20-203.ngrok-free.app'

[ ] 1 # streamlit 백그라운드 프로세스 종료 및 ngrok 터널링 종료
2 !kill -f streamlit
3 ngrok.disconnect(ngrok_tunnel.public_url)

WARNING: pyngrok process ngrok-t-2024-09-20T14:09:36+0000 [vi=main] asp="Stopping forwarder" name=http-5011-e766504-25d9-4142-95f6-a1918700bc06 acceptErr="failed to accept connection: Listener closed"

[ ] 1 [요청을 시작하거나 AI로 코드를 생성하세요.]
```

➤ API_KEY 변수에 Authtoken Key 대입 코드 실행

```
[ ] 1 !pip install pyngrok streamlit

[ ] 2 1 %writefile main.py
3 2 import streamlit as st
4 3
5 4 # 앱의 제목을 설정
6 5 st.title('ngrok 테스트')
7 6
8 7 # 텍스트를 화면에 출력
9 8 st.write('AI Essential')

[ ] 9 Overwriting main.py

[ ] 10 1 API_KEY = ""

[ ] 11 1 from pyngrok import ngrok
2
3 2 # ngrok 서비스 인증
4 3 ngrok.set_auth_token(API_KEY)
5 4
6 5 # main.py 백그라운드 프로세스 실행
7 6 !nohup streamlit run main.py --server.port 5011 &
8 7
9 8 # ngrok 터널링 실행
10 9 ngrok_tunnel = ngrok.connect(addr='5011', proto='http', bind_tls=True)
11 10
12 11 # ngrok 터널링 결과
13 12 print(' * Tunnel URL: ', ngrok_tunnel.public_url)

[ ] 14 nohup: appending output to 'nohup.out'
 * Tunnel URL: 'https://3071-34-169-20-203.ngrok-free.app'

[ ] 15 1 # streamlit 백그라운드 프로세스 종료 및 ngrok 터널링 종료
2 !kill -f streamlit
3 ngrok.disconnect(ngrok_tunnel.public_url)

[ ] 16 WARNING:pyngrok.process.ngrok:t-2024-09-20T14:03:36+0000 [v1=warn,msg="Stopping forwarder" name=http-5011-e766504-25d9-4142-95f6-a1918700c06 acceptErr="failed to accept connection: Listener closed"]

[ ] 17 1 코딩을 시작하거나 AI로 코드를 생성하세요.
```

- pyngrok 서비스 및 터널링 코드 실행 후 출력결과에 나오는 Tunnel URL을 클릭

```
[ ] 1 !pip install pyngrok streamlit

[ ] 2
3
4 # 앱의 제목을 설정
5 st.title('ngrok 테스트')
6
7 # 텍스트를 화면에 출력
8 st.write('AI Essential')

Overriding main.py

[ ] 1 API_KEY = ""

[ ] 1 from pyngrok import ngrok
2
3 # ngrok 서비스 인증
4 ngrok.set_auth_token(API_KEY)
5
6 # main.py 백그라운드 프로세스 실행
7 !nohup streamlit run main.py --server.port 5011 &
8
9 # ngrok 터널링 실행
10 ngrok_tunnel = ngrok.connect(addr='5011', proto='http', bind_tls=True)
11
12 # ngrok 터널링 결과
13 print(' * Tunnel URL: ', ngrok_tunnel.public_url)

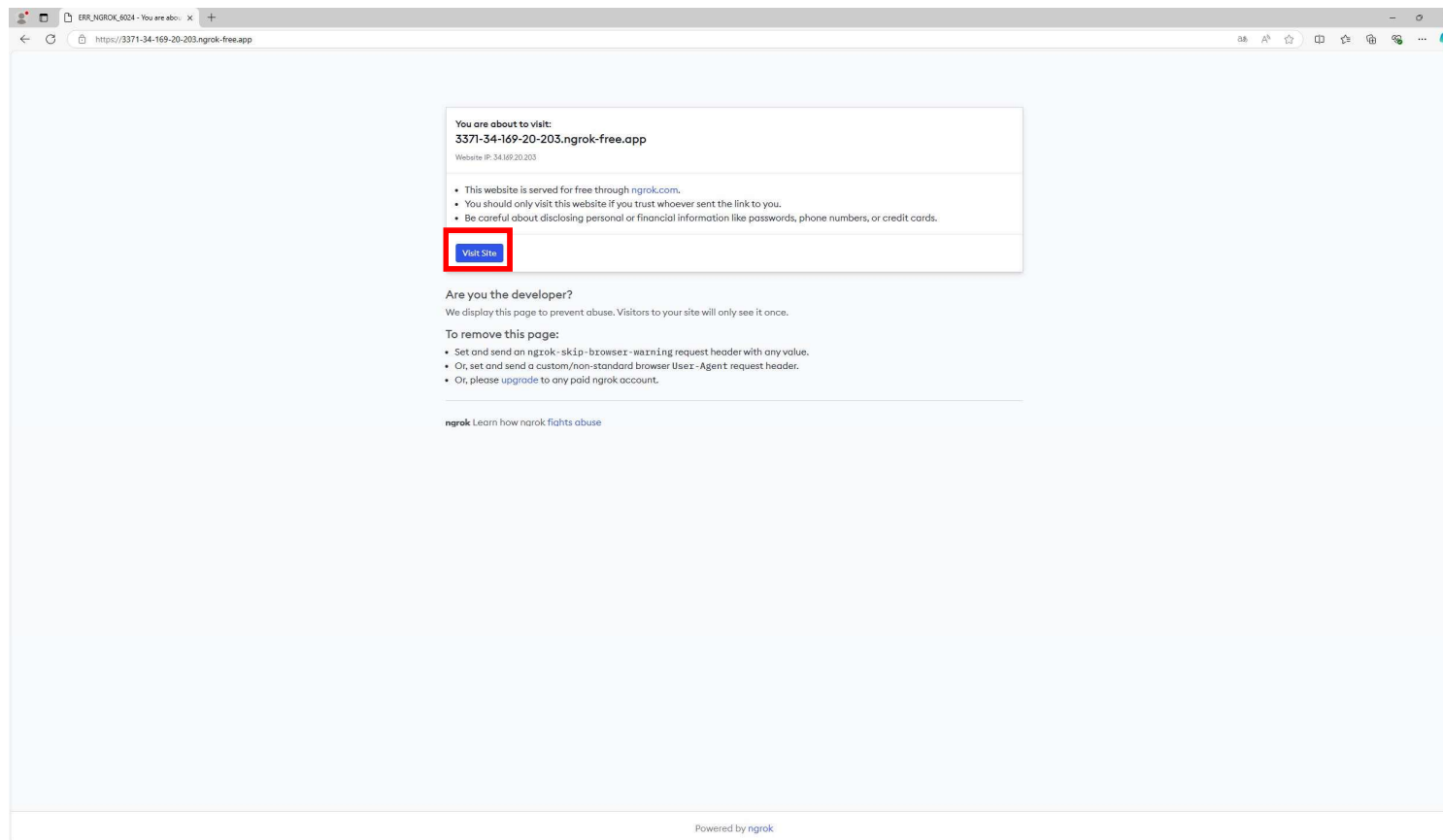
whup: appending output to 'whup.out'
* Tunnel URL: https://391-34-192-20-203.ngrok-free.app

[ ] 1 # streamlit 백그라운드 프로세스 종료 및 ngrok 터널링 종료
2 !kill -f streamlit
3 ngrok.disconnect(ngrok_tunnel.public_url)

WARNING: pyngrok.process.ngrok:1-2024-09-20T14:03:36+0000 [v1=warn,msg="Stopping forwarder" name=http-5011-e766504-25d9-4142-96f6-a1918700bc06 acceptErr="failed to accept connection: Listener closed"]

1 [요청을 시작하거나 AI로 코드를 생성하세요.]
```

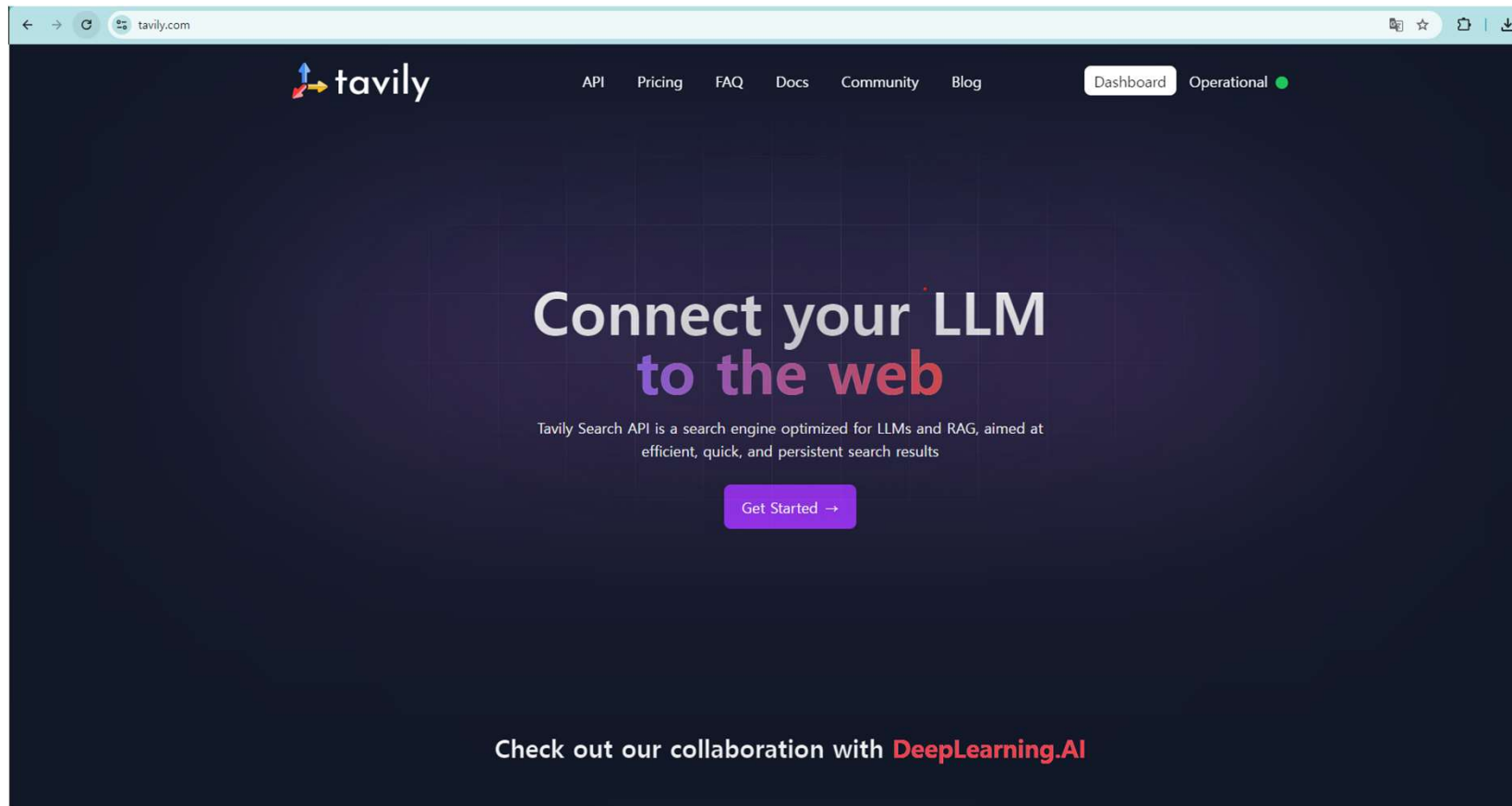
➤ Tunnel URL 접속 화면에서 Visit Site 버튼 클릭



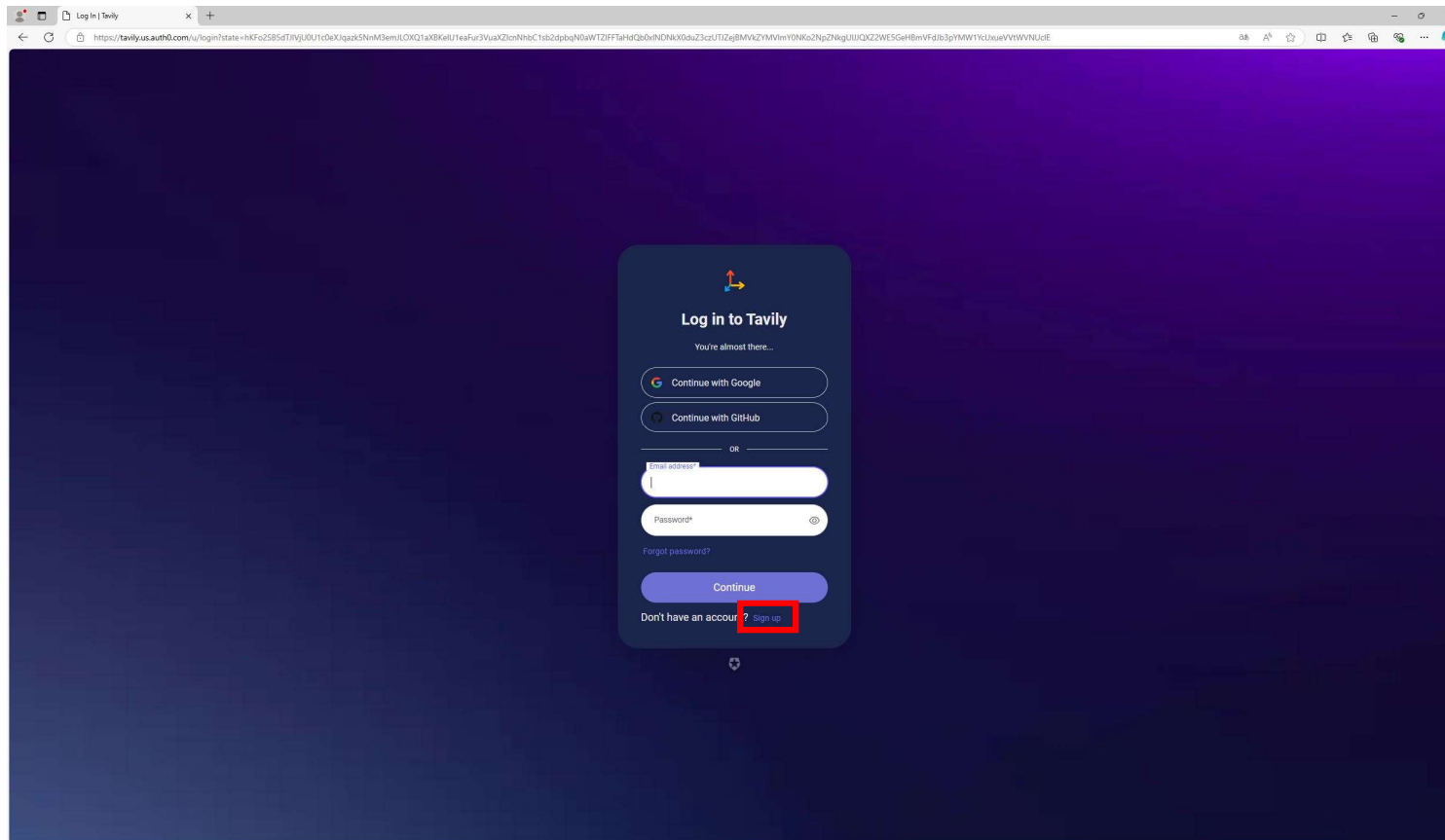
➤ streamlit 서비스 결과 확인



➤ Tavily 홈페이지 <https://tavily.com> 접속



- 하단 Sing up 버튼 클릭하여 회원가입 화면으로 이동



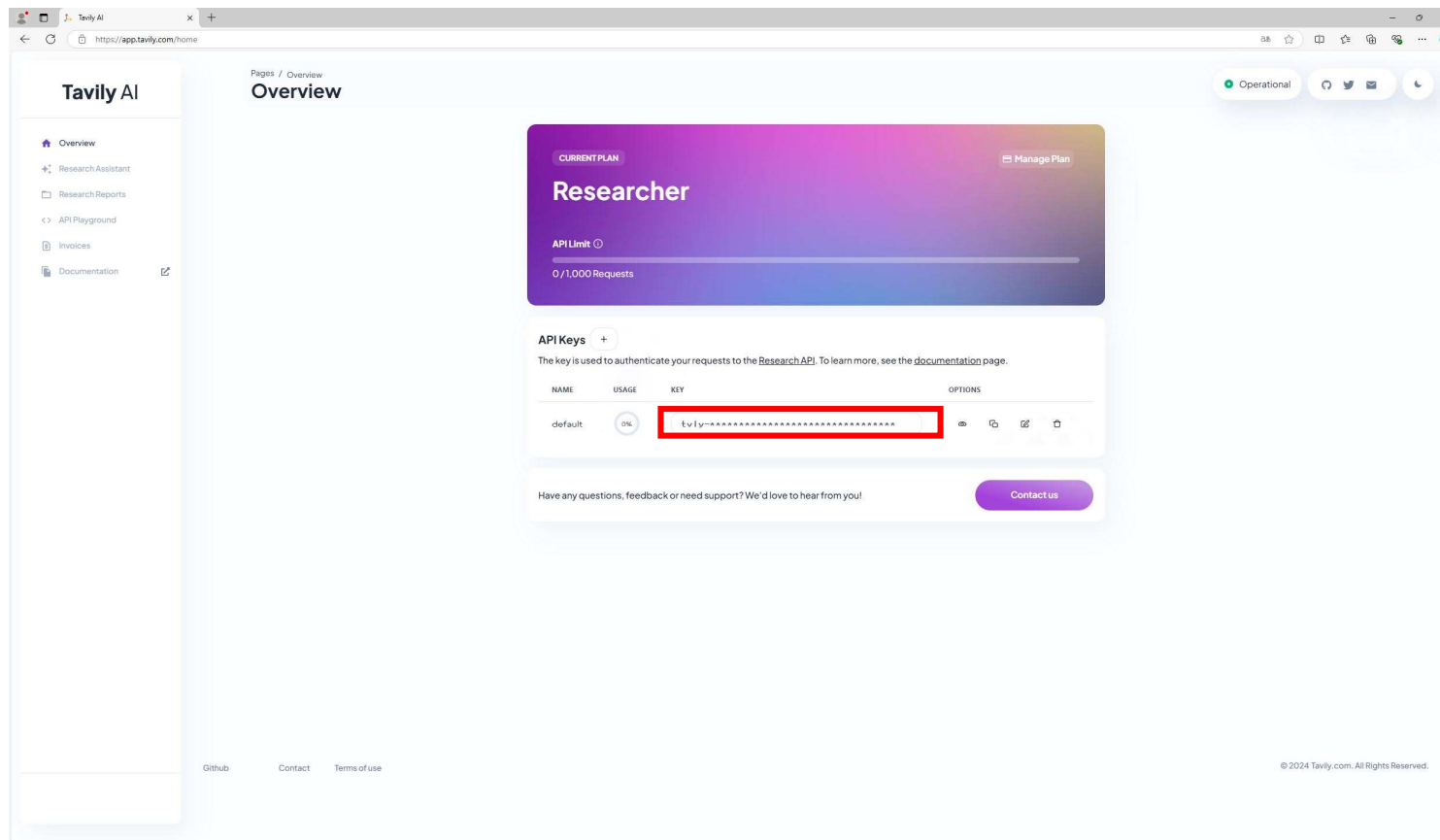
- 하단 Sing up 버튼 클릭하여 회원가입 화면으로 이동

The screenshot shows a web browser window with the URL <https://tavily.us.auth0.com/u/signup?state=HKFz25BuRjctX29K6GRlXh1ZEvyOE0bVhpRVayWGS5lTnmeaFur3YuaX2cnHbcC1sb2dpbqN0aWtZGR0VvVNTNsbENKlXWU5pa1R2UVBldpRnRvV3Rlc2NpZnkgLUQKZ2WESGaH8mVfdB3pYMW1YdJxueVYVWVWNUdE>. The page has a dark blue background with a large white text overlay on the left that reads "소셜 로그인 기반 회원가입" (Social login based membership registration). In the center, there is a white sign-up form titled "Welcome" with the subtitle "Sign Up to Tavily to continue to Tavily." The form contains two rows of options, each enclosed in a red rectangular box. The first row contains "Continue with Google" and "Continue with GitHub". The second row contains "Email address*" and "Password*" with a "Continue" button below them. To the right of the form, the text "이메일 기반 회원가입" (Email based membership registration) is displayed. At the bottom of the form, there is a link "Already have an account? Log in".

소셜 로그인 기반 회원가입

이메일 기반 회원가입

➤ Key 복사



The screenshot shows the Tavily AI Overview page. The left sidebar contains navigation links: Overview, Research Assistant, Research Reports, API Playground, Invoices, and Documentation. The main content area is titled 'Overview' and features a 'CURRENT PLAN' section for 'Researcher' with an 'API Limit' of 0/1,000 Requests. Below this is the 'API Keys' section, which includes a table with columns for NAME, USAGE, KEY, and OPTIONS. The table lists a 'default' key with a usage of 0%. The key value is partially visible and highlighted with a red box. At the bottom of the page, there is a 'Contact us' button and a copyright notice: '© 2024 JAS All rights reserved'.

| NAME | USAGE | KEY | OPTIONS |
|---------|-------|------------|---------|
| default | 0% | tvly-***** | |

➤ Key 테스트를 위한 Google Colab Notebook 접속

<https://colab.research.google.com/drive/12yFBbBNwvStV318ArgP1c-fkf4dKE1Uh?usp=sharing>

The image shows a Jupyter Notebook interface with a single cell containing a Python script. The script uses the Tavily API to search for information about the LangChain Community. The script includes comments in Korean explaining the search process and the results.

```
[AI Essential] Tavily Key Test.py ☆
파일 수정 보기 삽입 단편 도구 도움말 변수와함께 저장된 셀 열람
+ 코드 + 텍스트 + DIVE 킷
1 pip install langchain_community

[ ] 1 import os
2 os.environ["TAVILY_API_KEY"] = ""

[ ] 1 from langchain_community.tools.tavily_search import TavilySearchResults
2 search = TavilySearchResults(k=5) # 검색 결과를 5개까지 가져온다는 의미

[ ] 1 search.invoke("삼성전자에 대해서 알려줘")

[ ] [{"url": "https://www.kbstar.com",
      "content": "삼성전자는 한국 최대의 전자제품 제조업체로, 스마트폰, TV, 가전제품 등 다양한 제품을 생산하고 있습니다. 삼성전자는 1983년 설립되었으며, 현재는 전 세계적으로 33개 연구개발(R&D) 센터를 두고 고연혁의 제품 개발을 진행하고 있습니다. 삼성전자는 1983년 설립되었으며, 현재는 전 세계적으로 33개 연구개발(R&D) 센터를 두고 고연혁의 제품 개발을 진행하고 있습니다."},
     {"url": "https://www.samsung.com/kr/about-samsung/home",
      "content": "삼성전자는 1983년 설립되었으며, 현재는 전 세계적으로 33개 연구개발(R&D) 센터를 두고 고연혁의 제품 개발을 진행하고 있습니다. 삼성전자는 1983년 설립되었으며, 현재는 전 세계적으로 33개 연구개발(R&D) 센터를 두고 고연혁의 제품 개발을 진행하고 있습니다."},
     {"url": "https://www.samsung.com/kr/about-samsung/home",
      "content": "삼성전자는 1983년 설립되었으며, 현재는 전 세계적으로 33개 연구개발(R&D) 센터를 두고 고연혁의 제품 개발을 진행하고 있습니다. 삼성전자는 1983년 설립되었으며, 현재는 전 세계적으로 33개 연구개발(R&D) 센터를 두고 고연혁의 제품 개발을 진행하고 있습니다."},
     {"url": "https://www.samsung.com/kr/about-samsung/home",
      "content": "삼성전자는 1983년 설립되었으며, 현재는 전 세계적으로 33개 연구개발(R&D) 센터를 두고 고연혁의 제품 개발을 진행하고 있습니다. 삼성전자는 1983년 설립되었으며, 현재는 전 세계적으로 33개 연구개발(R&D) 센터를 두고 고연혁의 제품 개발을 진행하고 있습니다."},
     {"url": "https://www.samsung.com/kr/about-samsung/home",
      "content": "삼성전자는 1983년 설립되었으며, 현재는 전 세계적으로 33개 연구개발(R&D) 센터를 두고 고연혁의 제품 개발을 진행하고 있습니다. 삼성전자는 1983년 설립되었으며, 현재는 전 세계적으로 33개 연구개발(R&D) 센터를 두고 고연혁의 제품 개발을 진행하고 있습니다."}]]
```

➤ 본인 계정의 Drive로 복사 (로그인 필요)



➤ 연결 버튼을 클릭하여 Python 런타임 연결



- langchain_community 패키지 설치 코드 실행



➤ Tavily KEY 환경 변수 설정 (본인 Key 대입)



```
1 !pip install langchain_community

2 import os
3 os.environ['TAVILY_API_KEY'] = ""

4 from langchain_community.tools.tavily_search import TavilySearchResults
5 search = TavilySearchResults(k=5) # 검색 결과를 5개까지 가져오겠다는 의미

6 search.invoke("삼성전자에 대해서 알려줘")
```

[Content: 삼성전자 관계자는 "다양한 디바이스와 다양한 애플리케이션을 지원하는 정보기술(IT) 기업에 있어 ... 전 세계에 걸쳐 고객들의 삶에 새로운 가치를 부여할 혁신 ..."]

➤ 검색 도구 생성



➤ 검색

