# <오전>

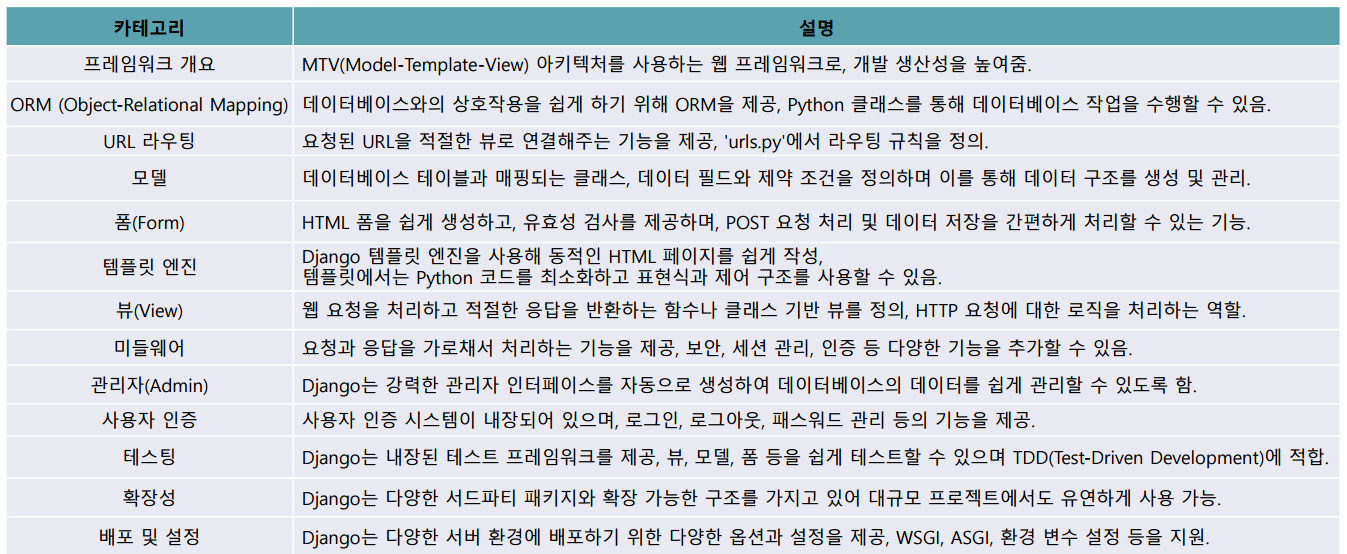
## CRUD 어플리케이션 제작

### Django를 활용한 CRUD 어플리케이션 제작

#### Django란?

Django는 웹 애플리케이션 개발을 빠르고 효율적으로 할 수 있게 도와준다.

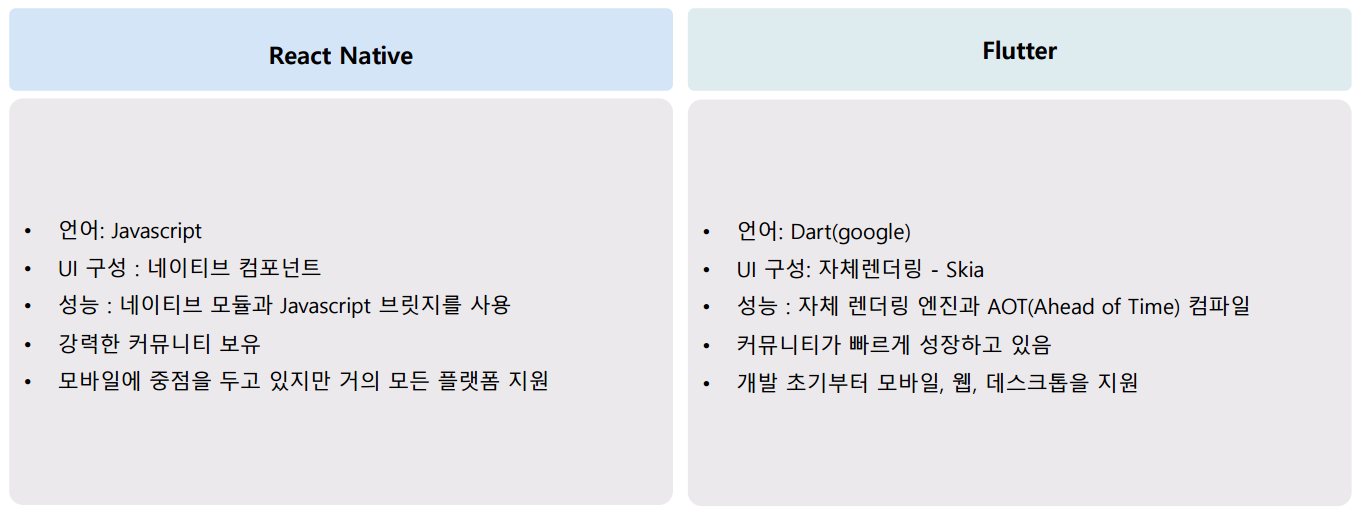
데이터베이스 관리, 사용자 인증, 어드민 화면 등을 쉽게 구현할 수 있는 기능을 내장하고 있다.



#### Cross-Platform Framework

대표적인 크로스 플랫폼 프레임워크로는 React Native와 Flutter가 있다.

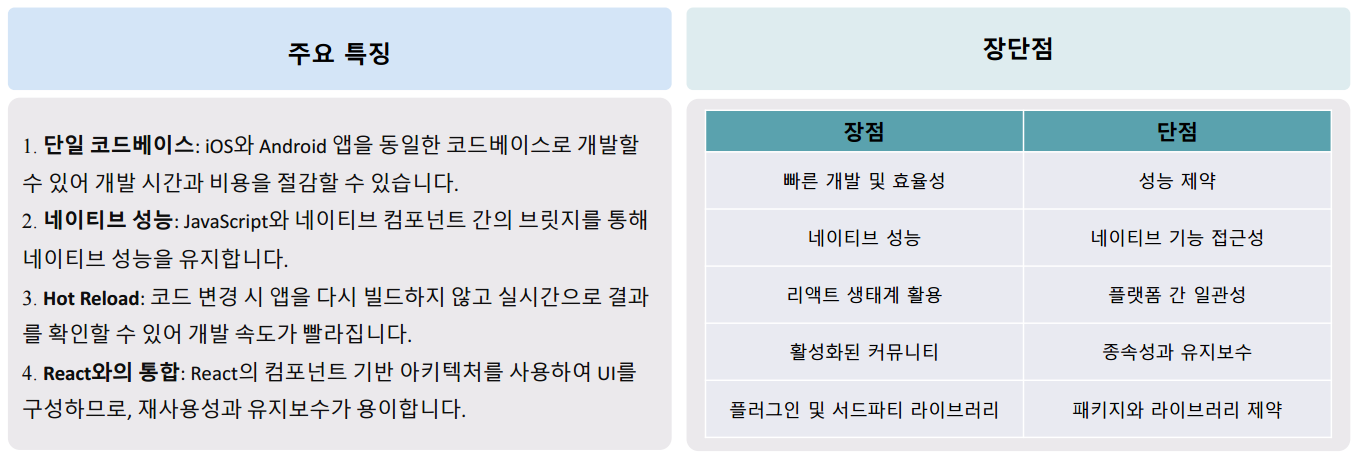
그 외 Xamarin, lonic 등 여러 크로스 플랫폼 프레임워크가 있다.



#### React Native

React Native는 Facebook(현재 Meta)에서 개발한 오픈 소스 프레임워크로, 하나의 코드베이스로 iOS와 Android 앱을 개발할 수 있게 해준다.

React Native는 JavaScript와 React 라이브러리를 기반으로 하며, 모바일 애플리케이션 개발을 보다 효율적이고 간편하게 만들어 준다.



React Native는 expo go 와 react native cli 환경을 제공한다.

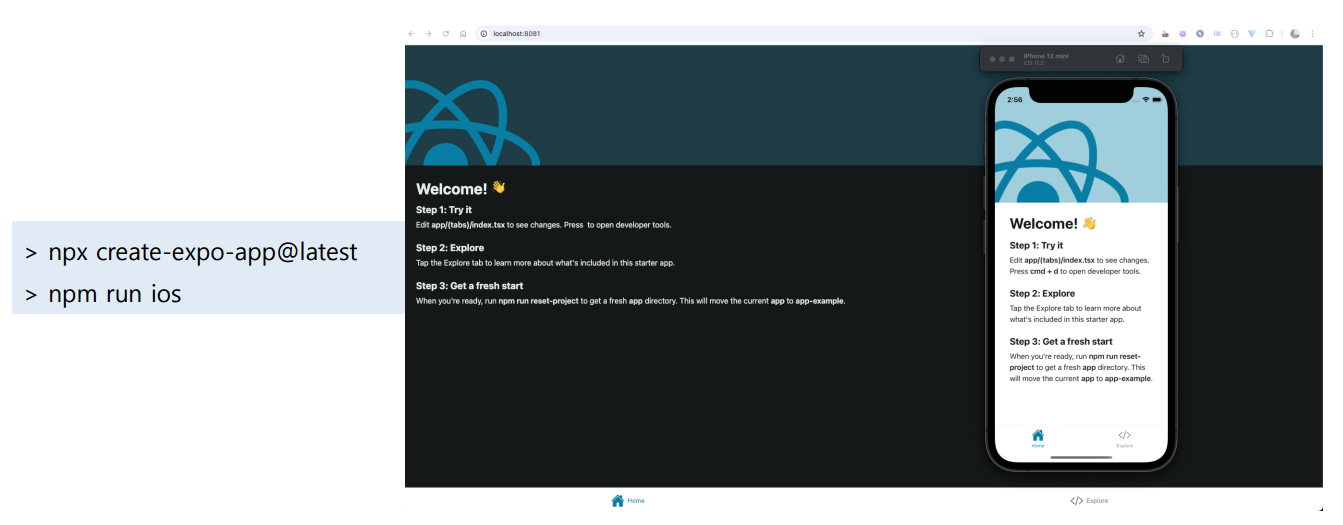
0.74 버전부터는 Expo를 권장하고 있지만, 아직은 자유도 문제가 있다.



##### React Native(Expo Go)

Expo를 사용해서 프로젝트를 만들게 되면 간단한 명령어로 설치하고 실행할 수 있다.

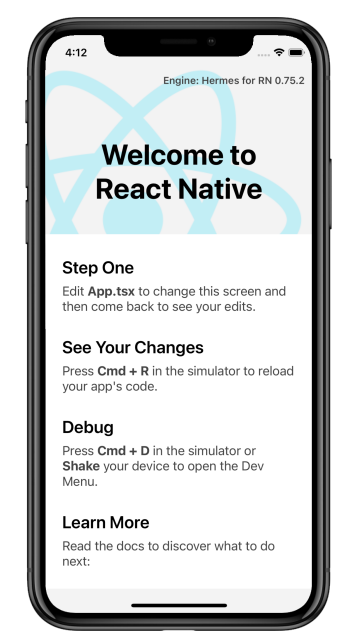
현재 Expo는 IOS, Android, Web, TV를 지원한다.



##### React Native(cli)

CLI를 이용해서 React Native 프로젝트를 만들 수도 있다.

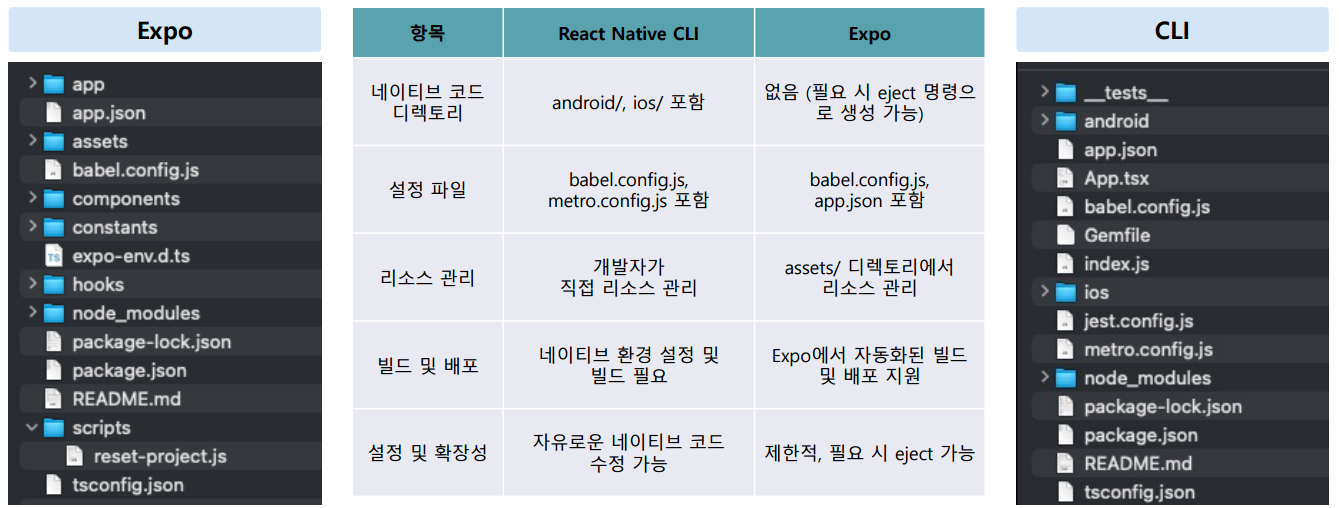
Android, IOS, macOS, TV, watchOS, Web, Windows, visionOS를 지원한다.



Npm run ios가 아닌 react-native run-android를 사용해야 한다.

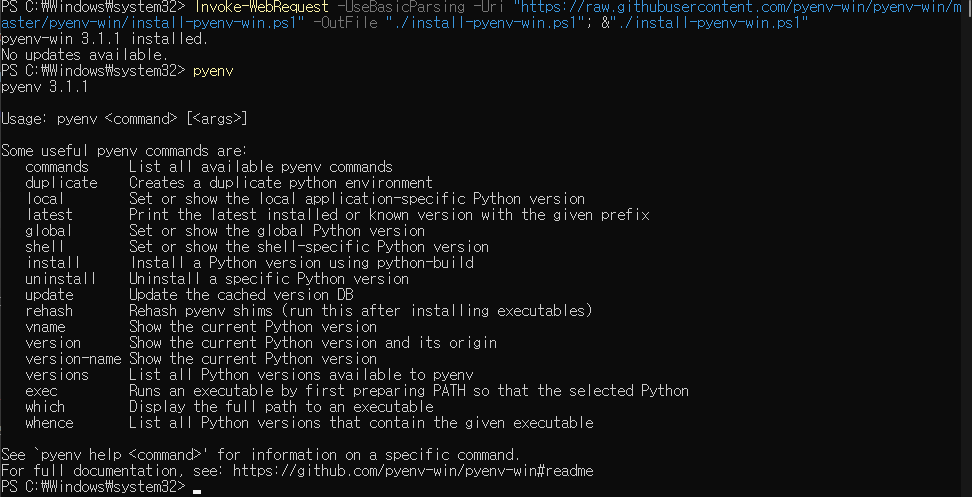
CLI를 통해 만들어진 프로젝트는 Android, IOS 네이티브 코드가 모두 포함되어 있는 반면, Expo로 생성된 프로젝트는 네이티브 코드가 포함되어 있지 않다.

단, 필요한 경우에 expo eject 명령어로 네이티브 코드를 얻을 수 있다.



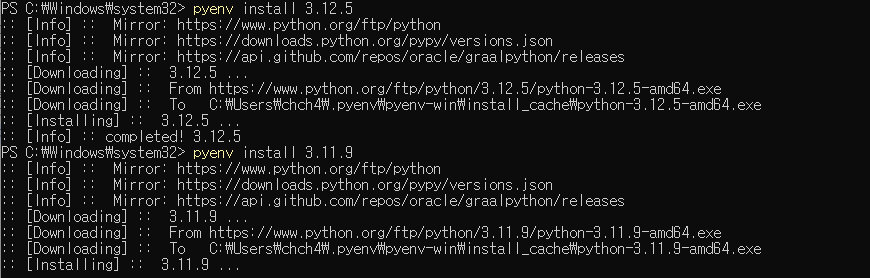
### 가상 환경 설정

#### 가상 환경 설치 및 확인



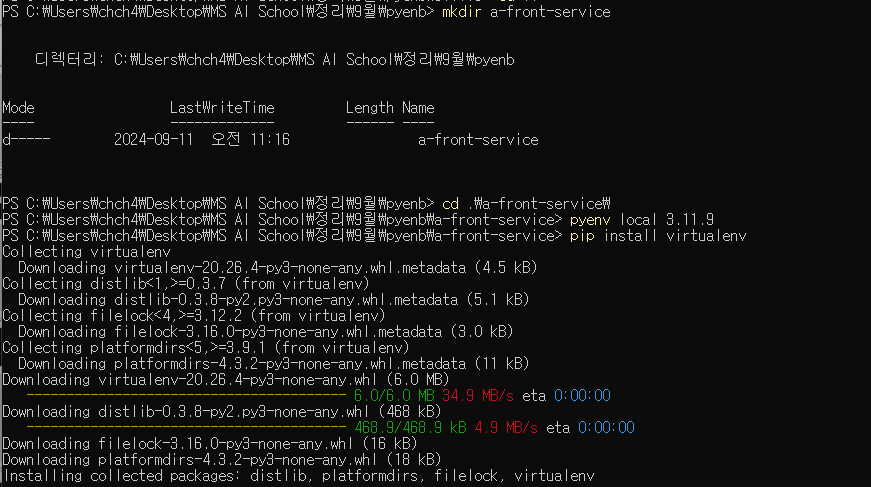
#### 가상 환경 python 설치 및 구성

pyenv install –list를 사용하면 설치 가능한 것들의 list를 확인할 수 있다.

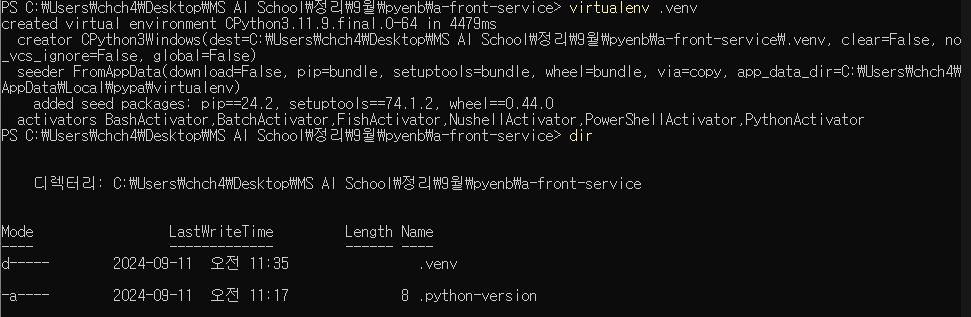




디렉터리 생성 후 python 설정 및 virtualenv 설치



.venv 폴더 생성



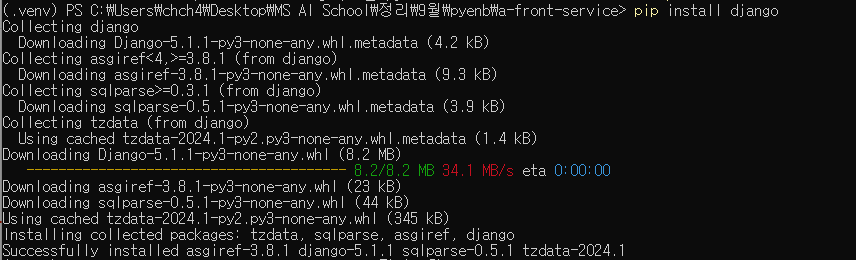
activate를 실행시켜 가상환경에서 셋팅이 될 수 있도록 설정.

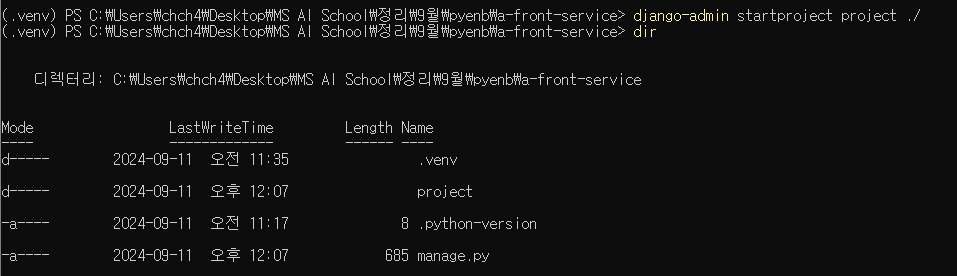
(.venv)가 앞에 생긴 것을 확인할 수 있으며, 가상환경 세팅 후 pip를 통해 패키지를 설치할 때 반드시 가상환경이 activate 된 상태에서 진행해야 한다.

가상환경 실행 명령어: .\.venv\Scripts\activate

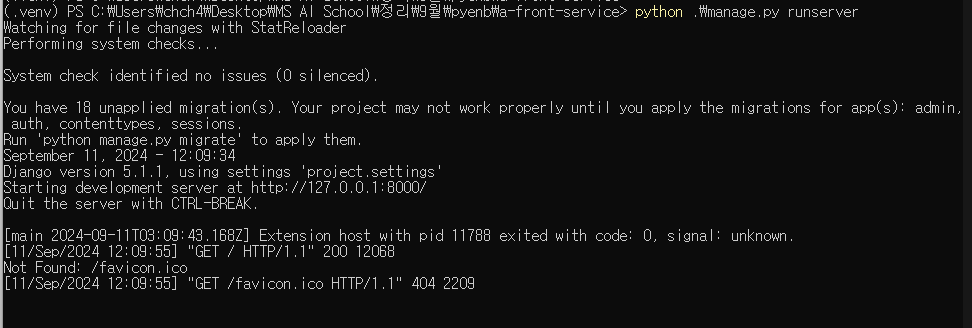


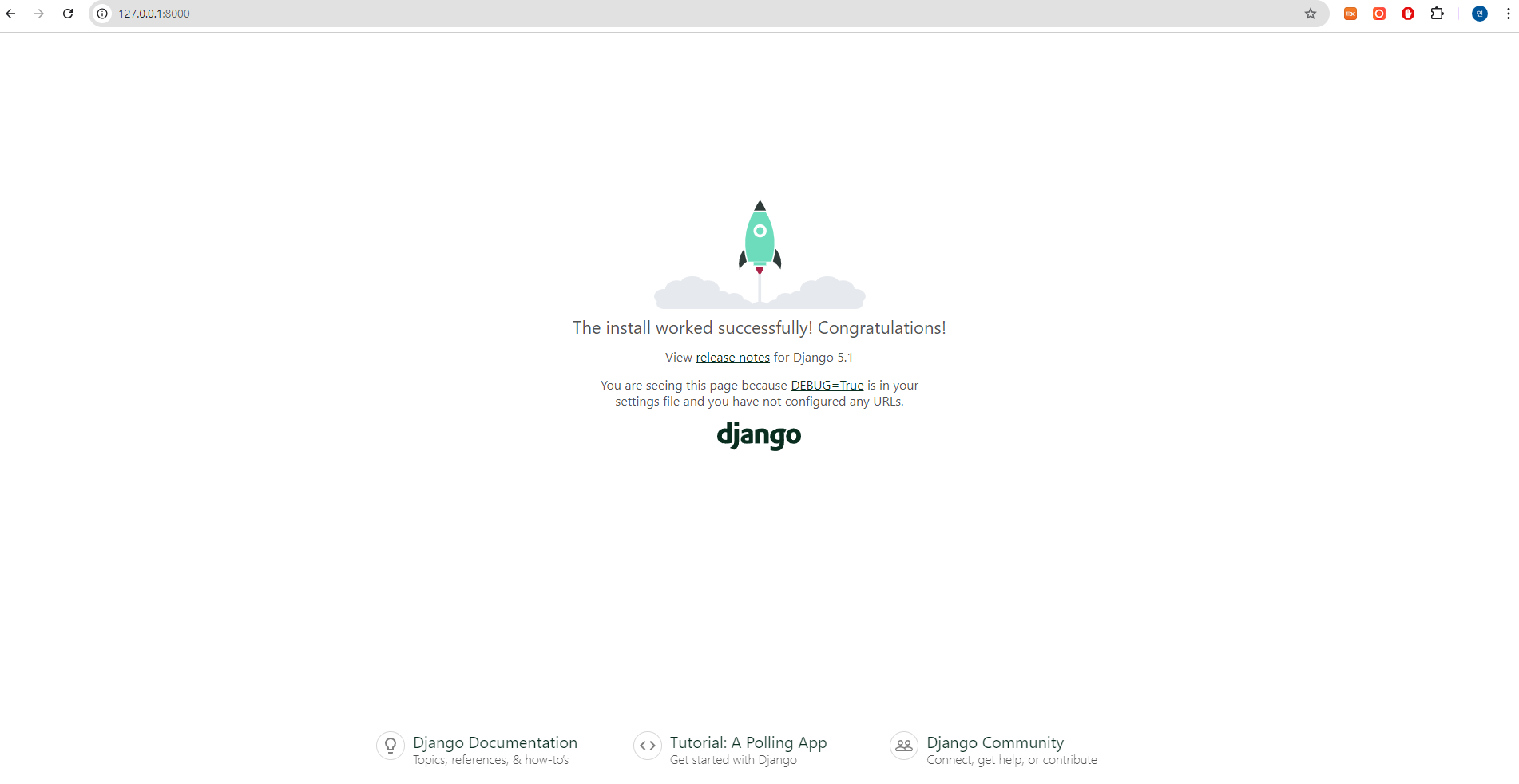
#### django 설치





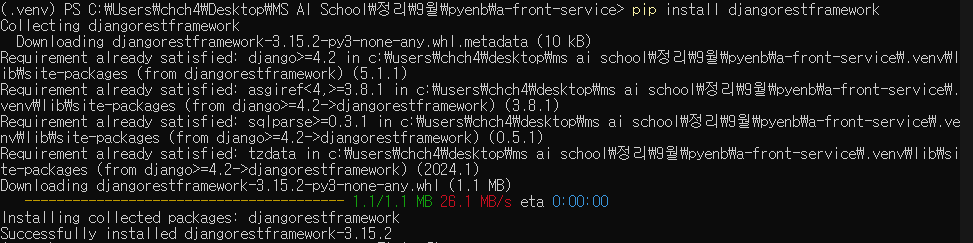
서버 구동



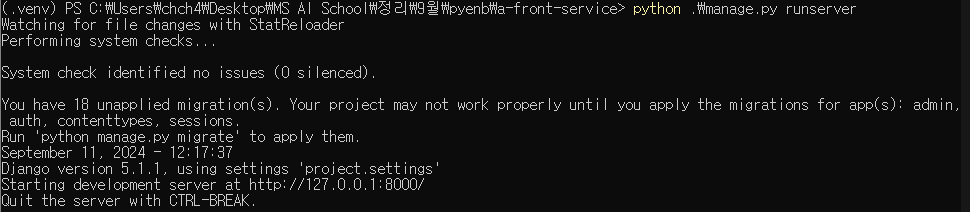


#### django 라이브러리 설치

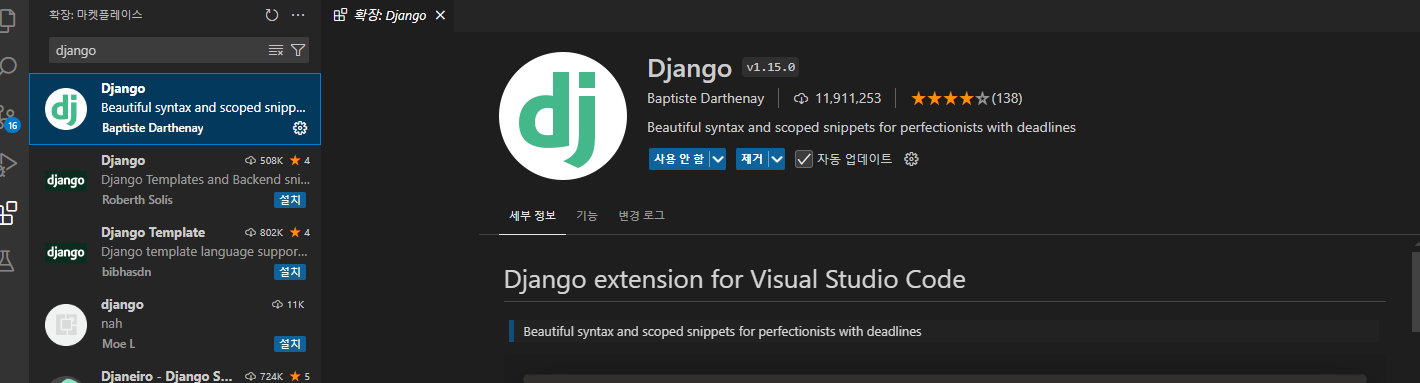
djangorestframework 설치



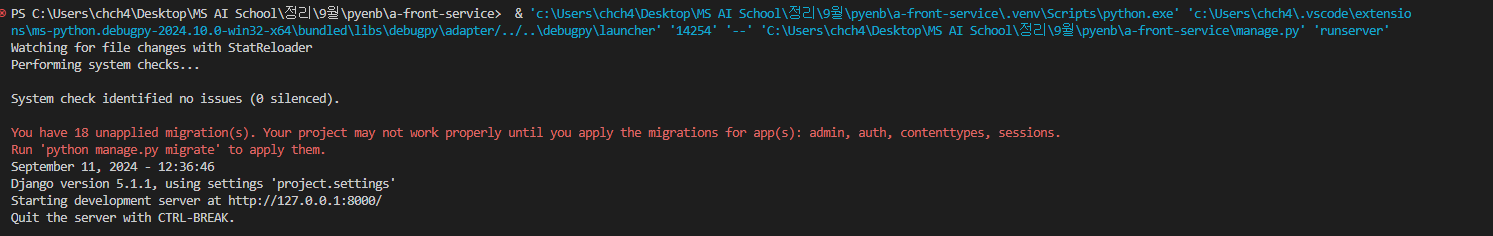
이후 실행해보면 에러가 발생하지 않는다.



#### VS Code 세팅

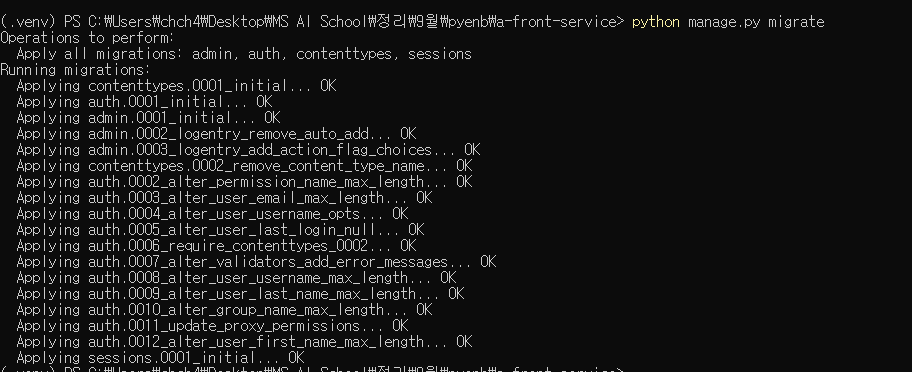


Django를 설치 후 vscode에서 디버그 형식으로 실행해도 된다.

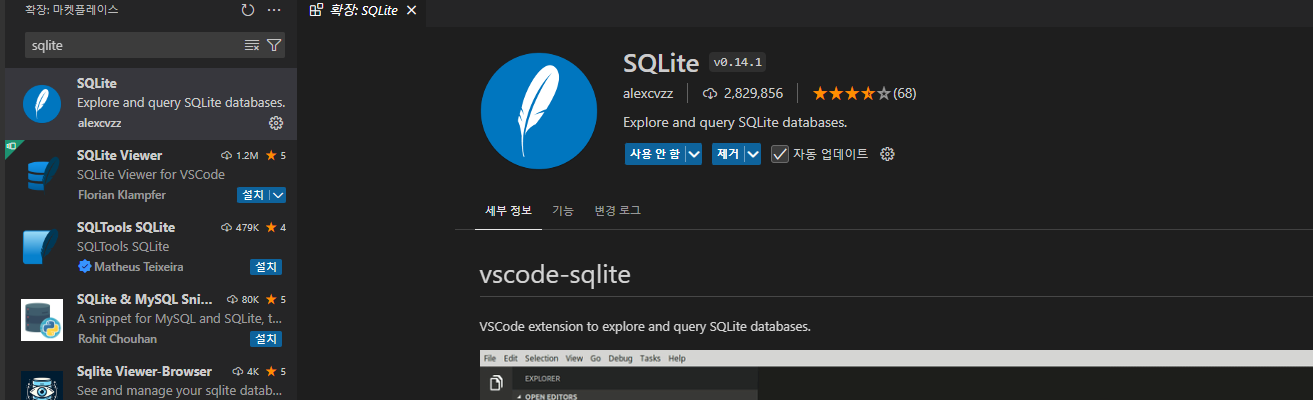


# <오후>

python manage.py migrate 명령어를 통해 데이터 베이스를 초기화한다.

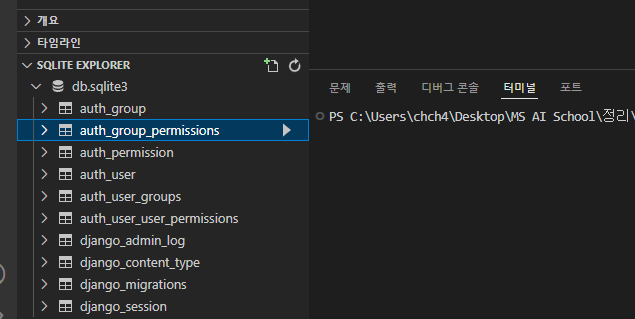


Sqlite 데이터베이스 확인을 위해 플러그인을 설치한다.



명령 팔레트에 sqlite의 open database를 선택하면 아래와 같이 생성이 된다.

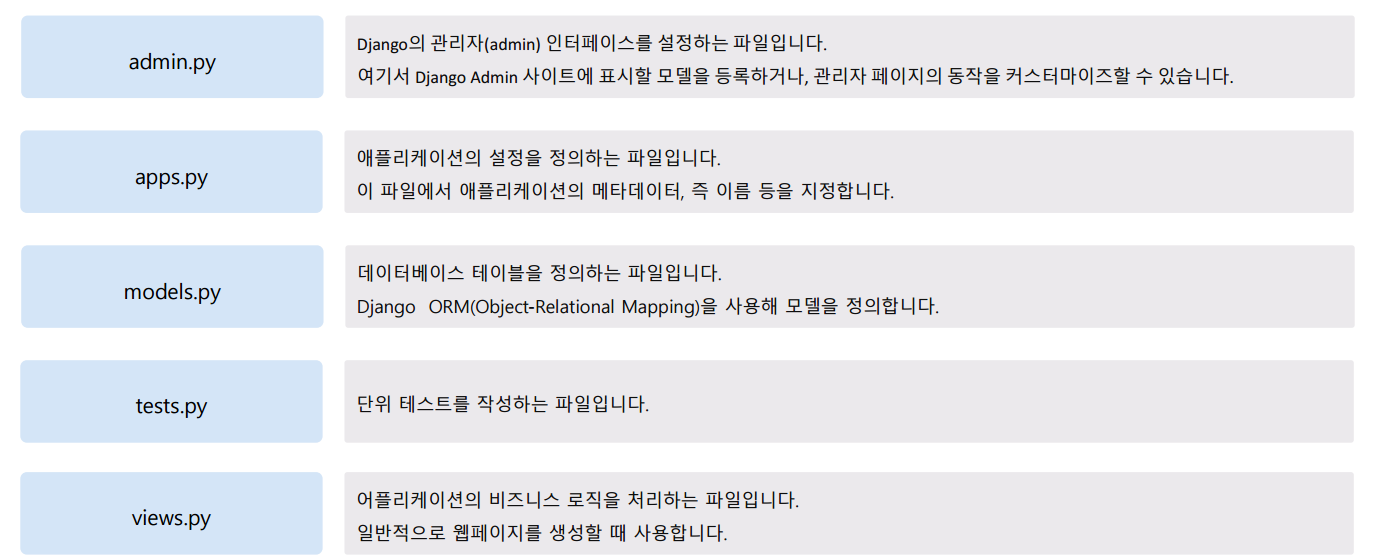




#### Django에 Application 추가

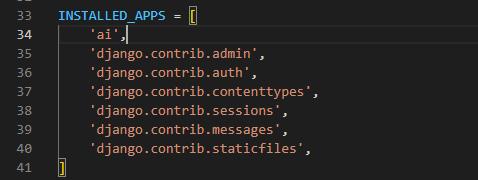


Djano application 파일 주요 기능

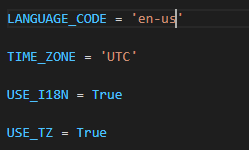
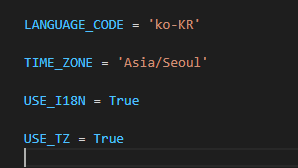


#### 프로젝트 설정

project -> settings.py 에 INSTALLED\_APPS에 ‘ai’를 추가



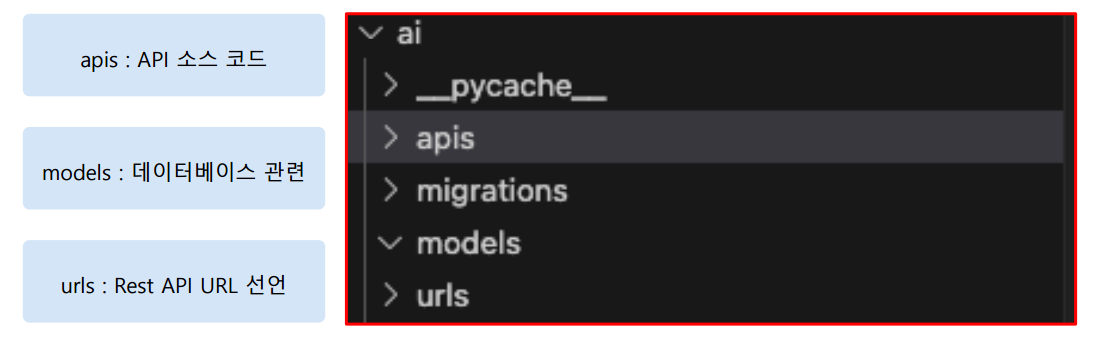
언어와 타임존을 변경

 -> 

한국어로 변경된 것을 확인할 수 있다.



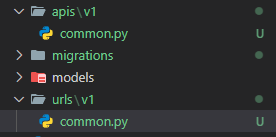
Ai 폴더 안에 apis, models, urls 폴더를 생성한다.



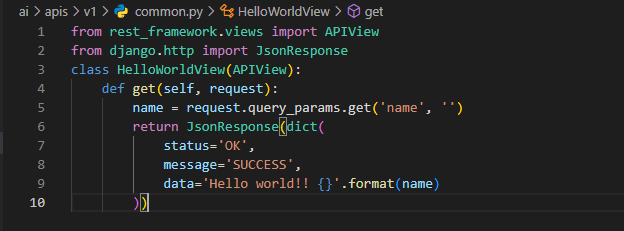
#### 더미 API 개발

apis 하위에 v1 폴더를 만들고 v1 폴더에 common.py 파일을 생성

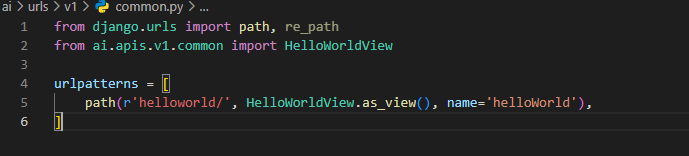
urls 도 동일하게 만들어 준다.

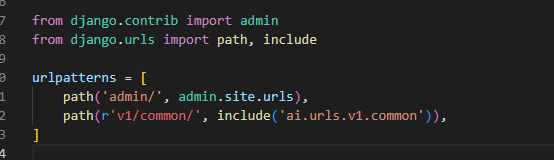


ai/apis/v1/common.py 파일에 코드를 작성한다.

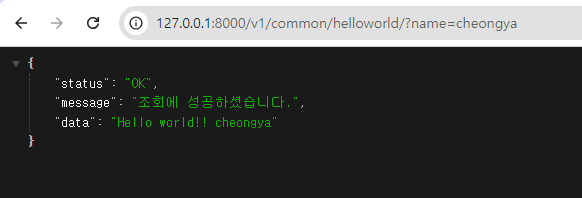


ai/urls/v1/common.py 파일을 열어서 helloworld url을 등록하고, 등록된 common.py의 url을 project의 urls.py에 등록해준다.



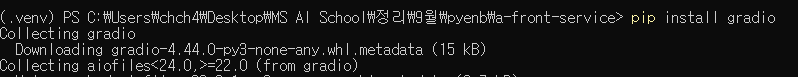


Json 형태의 데이터를 정상적으로 받는 것을 확인할 수 있다.

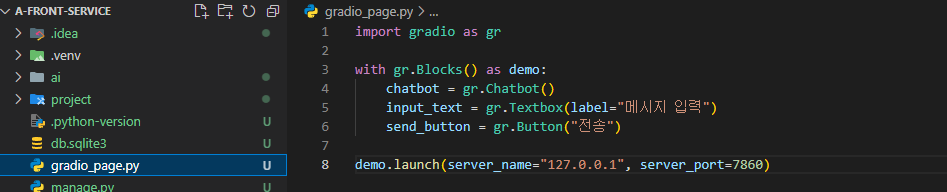


#### Gradio 설치

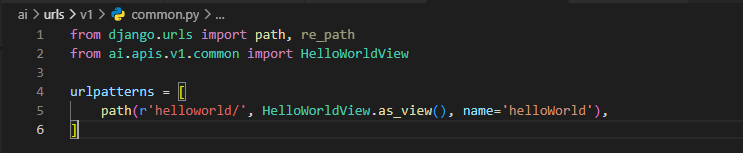
pip install gradio 명령어를 이용해 gradio 설치



gradio\_page.py 파일을 생성 후 샘플 코드 작성

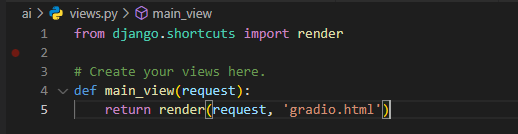


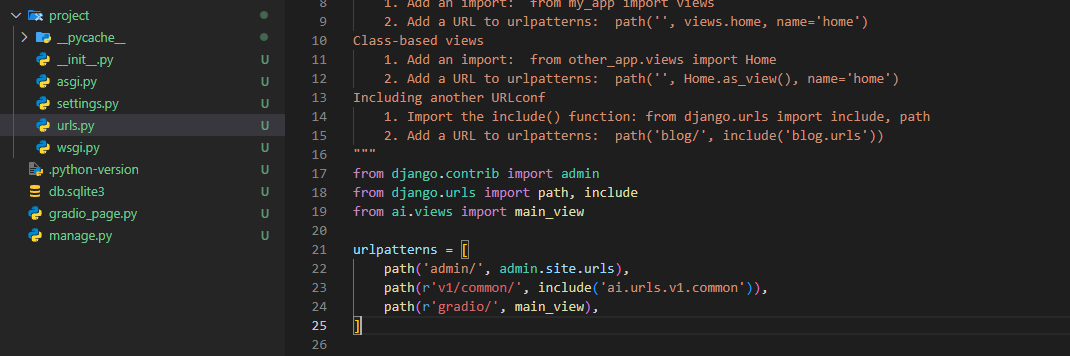
Django가 Gradio 화면을 리턴할 수 있도록 코드를 작성한다. ai 폴더 하위에 templates 폴더를 만들고 gradio.html을 만들어 준다. 화면에 gradio 화면이 보여지도록 src와 width, height를 설정해준다.



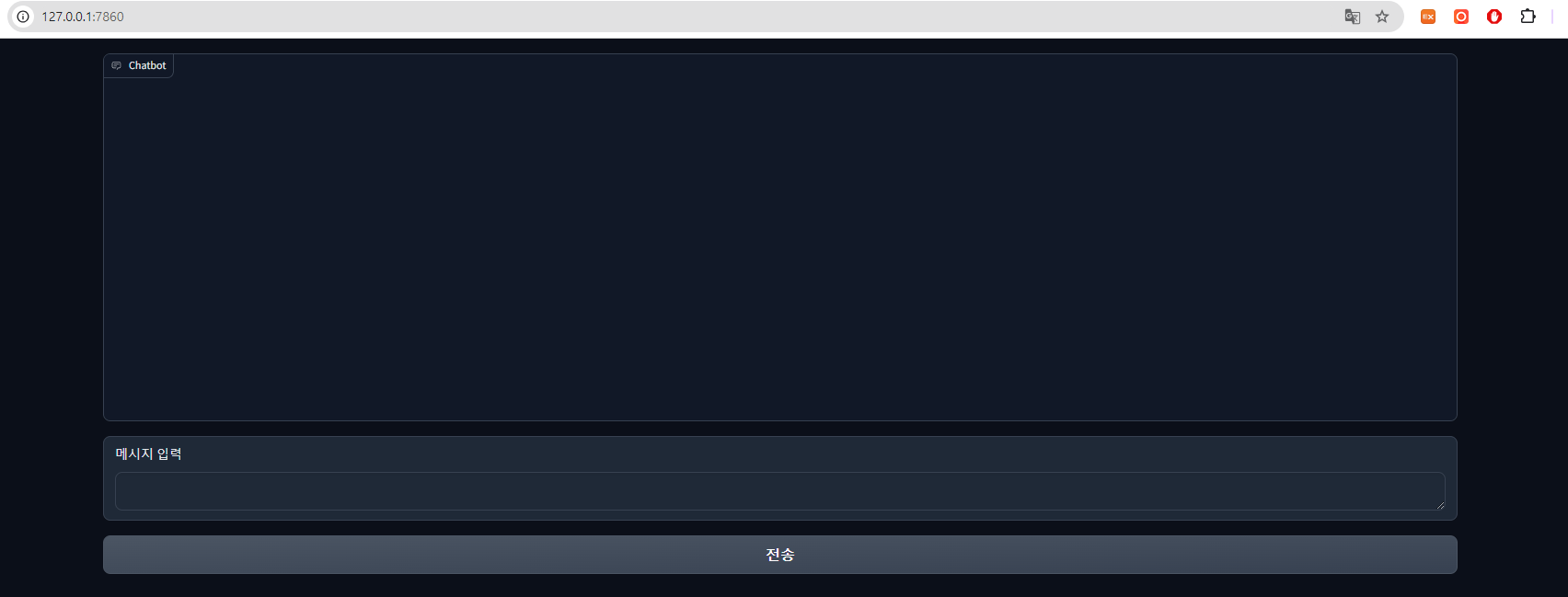
화면 요청이 들어왔을 때 gradio.html이 리턴될 수 있도록 main\_view 함수를 만든다.

project/urls.py 에 url을 등록해준다.





이후 gradio\_page.py를 개별로 실행 후 127.0.0.1:7860에 접속해 본다.



실행한 상태로 url에 등록해 둔 대로 주소/gradio/에 접속해 본다.



### 별첨

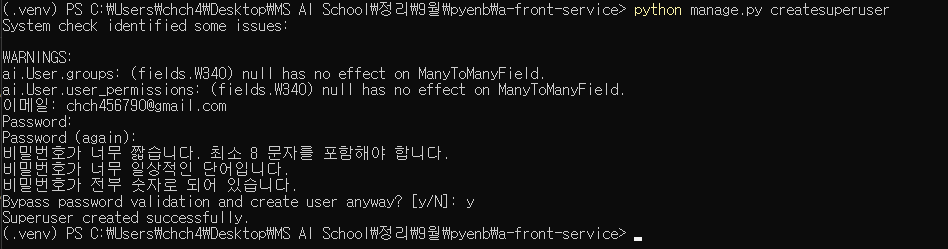
python manage.py makemigrations: migrations 생성

python manage.py migrate: 데이터 베이스 초기화

### API 테스트

#### superuser 생성

python manage.py createsuperuser 명령어를 사용하여 superuser 생성



SQlite db에 생성된 것을 확인할 수 있다.

