

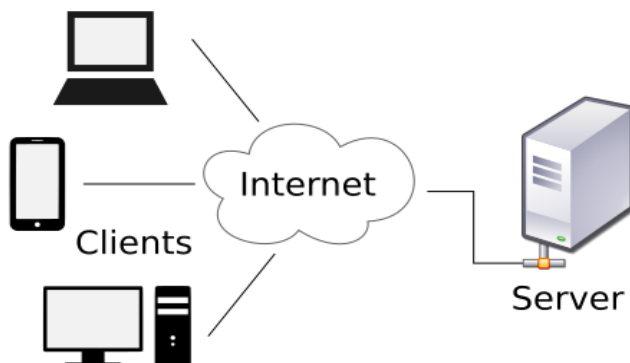
[Django(장고) 프로그래밍]

웹의 기본 이해

웹은 보통 **월드 와이드 웹(WWW)**이라고도 하며, 인터넷이라는 연결을 통해 여러 컴퓨터가 연결되어 서로 정보를 나누는 연결망들을 의미합니다. 이 웹이라는 것을 어떤 관점에서 보냐에 따라서 다음과 같은 개념이 생긴다.

1. 정보를 제공하는 컴퓨터와 정보를 받아가는 컴퓨터

서버	정보를 제공하기 위해 고정된 주소(도메인, 고정 IP 등을 차지하며 방문하는 컴퓨터들에게 필요한 정보를 제공한다. 보통의 우리가 접속하는 웹 사이트들이 이에 해당한다.
클라이언트	정보를 제공받기 위해 서버를 찾아 접속하는 컴퓨터입니다. 보통은 고정된 주소 없이 인터넷 연결을 통해 서버에 접근한다. 일반적으로 우리가 웹 브라우징 을 위해 쓰고 있는 컴퓨터들이 이에 해당한다.



2. 웹 프로그램의 구조 별 개념

프론트엔드	프론트엔드는 보통 사용자들이 보는 화면의 모습을 결정한다. HTML, CSS, JavaScript 프로그래밍으로 보통 구성되고 클라이언트의 웹 브라우저에서 코드가 실행되거나 그려진다.
백엔드	백엔드는 사용자가 접속하면 그 에 맞는 데이터를 보내주기 위해 여러가지 처리를 하는 부분을 담당하는 로직을 구성한다. 우리가 배울 Django 기반의 Python 프로그래밍이나 데이터베이스 등이 여기에 속한다. 프론트엔드 코드 조각들을 여기서 가지고 있다가, 사용자에게 적절하게 처리해서 보내주는 역할도 한다. 보통 서버에서 프로그램이 실행된다.(CGI, PHP, ASP, Servlet, JSP, NodeJS, Spring MVC)

3. 서버-클라이언트 간 통신 방향 별 개념

요청(Request)	클라이언트가 서버로 원하는 정보를 받기 위해 필요한 정보를 보내는 과정을 요청 이라고 한다. 보통은 클라이언트의 IP와 접속하고 있는 브라우저 프로그램이나 모바일 여부, 요청하는 URL과 방식, 그리고 필요한 경우 서버가 처리하기 위한 데이터들을 보낸다.
응답(Response)	요청을 받은 서버가 받은 데이터를 처리하여 사용자에게 정보를 내려주는 것을 응답 이라고 한다. 응답에는 여러가지 형태의 데이터가 내려갈 수 있는데, 우리가 보통 웹 사이트에 접속했을 때 보여지는 화면들은 응답에 HTML, CSS, JavaScript 코드가 포함되어 그걸 웹 브라우저가 해석하고 실행하여 화면에 그려지게 되는 케이스가 속한다.(JSON, XML...)

HTTP 프로토콜(통신규약)

HTTP(Hypertext Transfer Protocol)는 웹을 개발하는 사람이라면 누구나 다 알아야 하는 통신 프로토콜이다. 프로토콜이란 상호 간에 정의한 규칙을 의미하며 특정 기기 간에 데이터를 주고받기 위해 정의되었다. 통신 프로토콜을 쉽게 풀어보면 “나는 이렇게 줄 테니 넌 이렇게 받고 난 너가 준거 그렇게 받을게” 를 의미한다.

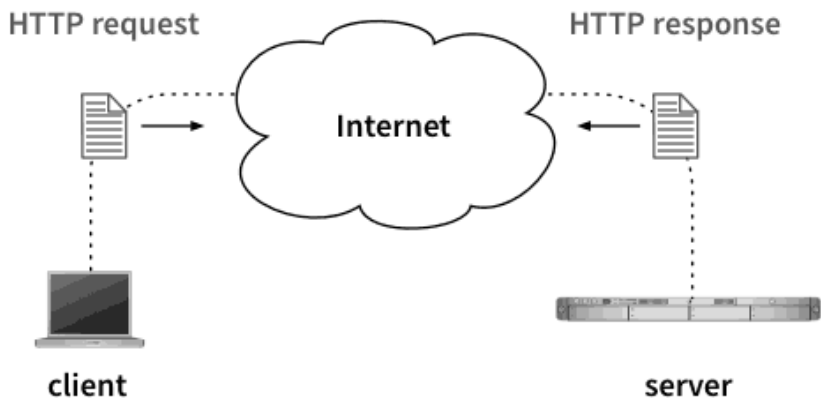
웹에서는 브라우저와 서버 간에 데이터를 주고받기 위한 방식으로 HTTP 프로토콜을 사용하고 있다.

- HTTP 프로토콜의 특징

HTTP 프로토콜은 **상태가 없는(stateless) 프로토콜**입니다. 여기서 ‘상태가 없다’ 라는 말은 데이터를 주고 받기 위한 각각의 데이터 요청이 서로 독립적으로 관리가 된다는 말이다. 좀 더 쉽게 말해서 이전 데이터 요청과 다음 데이터 요청이 서로 관련이 없다는 것이다. 이러한 특징 덕분에 서버는 세션과 같은 별도의 추가 정보를 관리하지 않아도 되고, 다수의 요청 처리 및 서버의 부하를 줄일 수 있는 성능 상의 이점이 생긴다. HTTP 프로토콜은 일반적으로 TCP/IP 통신 위에서 동작하며 기본 포트는 **80**번이다.

- HTTP Request & HTTP Response

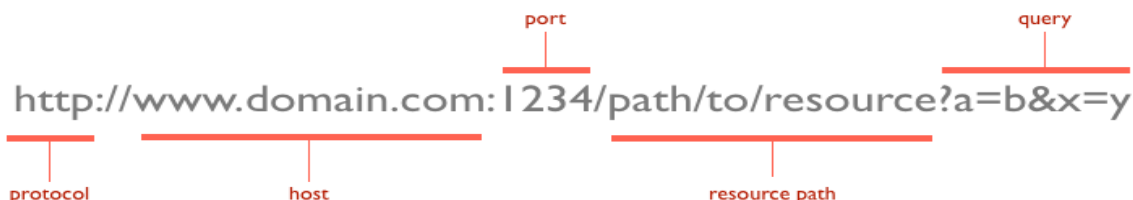
HTTP 프로토콜로 데이터를 주고받기 위해서는 아래와 같이 요청(Request)을 보내고 응답(Response)을 받아야 한다.



그리고 요청과 응답을 이해하기 위해서는 먼저 클라이언트(Client)와 서버(Server)를 이해해야 한다. 클라이언트란 요청을 보내는 쪽을 의미하며 웹 관점에서는 브라우저를 의미한다. 서버란 요청을 받는 쪽을 의미하며 데이터를 보내주는 원격지의 컴퓨터를 의미한다.

- URL

URL(Uniform Resource Locators)은 개발자가 아니더라도 이미 우리에게 익숙한 용어이다. 서버에 자원을 요청하기 위해 입력하는 영문 주소이며 URL 구조는 아래와 같다.(resource path 가 생략되면 기본파일을 달라는 의미로 해석 - index.html)



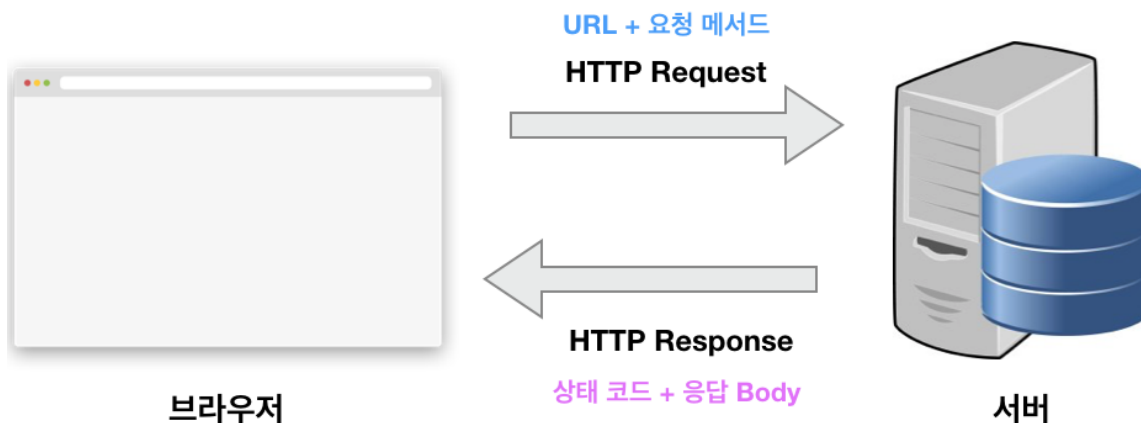
- HTTP 요청 메서드

앞에서 살펴본 URL을 이용하면 서버에 특정 데이터를 요청할 수 있다. 여기서 요청하는 데이터에 특정 동작을 수행하고 싶으면 HTTP 요청 메서드(HTTP Request Methods)를 이용한다.(GET 이 기본)

GET : 존재하는 자원에 대한 요청	PUT : 존재하는 자원에 대한 변경
POST : 새로운 자원을 생성	DELETE : 존재하는 자원에 대한 삭제

- HTTP 상태 코드(200, 404, 500)

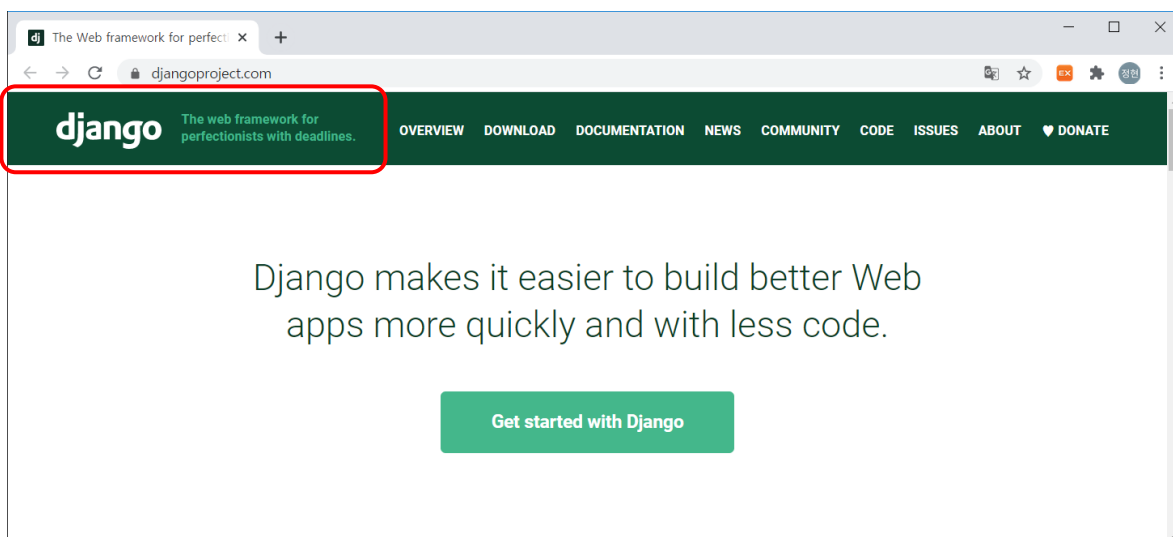
2xx - 성공, 3xx - 리다이렉션, 4xx - 클라이언트 에러, 5xx - 서버 에러



Django(장고)

장고는 파이썬으로 작성된 오픈 소스 웹 애플리케이션 프레임워크로, 모델-뷰-컨트롤러 패턴을 따르고 있다. 현재는 장고 소프트웨어 재단에 의해 관리되고 있다. 고도의 데이터베이스 기반 웹사이트를 작성하는 데 있어서 수고를 덜어주는 것이 장고의 주된 목표이다.

장고 공식 웹 사이트 URL : <https://www.djangoproject.com/>



- 장고로 개발된 웹 사이트



Instagram



Pinterest



Udemy



Sentry



Coursera



MIT

[라이브러리와 프레임워크]

- 라이브러리(API)란

재사용이 필요한 기능으로 반복적인 코드 작성을 없애기 위해 언제든지 필요한 곳에서 호출하여 사용할 수 있도록 Class나 Function으로 만들어진 것이다. 사용 여부는 코드 작성자 선택 사항이며 새로운 라이브러리 제작 시에도 엄격한 규칙이 존재하지 않는다. 제작 의도에 맞게 작성하면 된다.

간략 설명: 프로그램 제작 시 필요한 기능

비교 설명: 자동차 바퀴, 자동차 헤드라이트, 자동차 에어백

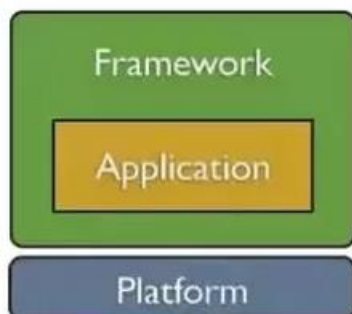


- 프레임워크란

원하는 기능 구현에만 집중하여 빠르게 개발 할 수 있도록 기본적으로 필요한 기능을 갖추고 있는 것으로 위에서 설명한 라이브러리가 포함되어 있다. 프레임워크만으로는 실행되지 않으며 기능 추가를 해야 되고 프레임워크에 의존하여 개발해야 되며 프레임워크가 정의한 규칙을 준수해야 한다.

간략 설명: 프로그램 기본 구조(빠대)

비교 설명: 자동차 프레임



정해져 있는 개발규칙 준수!!! 준수!!!

- 장고 프로그램의 개발 패턴

장고 프로그래밍에서 사용되는 개발 패턴은 MVC(Model View Controller) 패턴이다. MVC는 SW 공학에서 사용되는 SW 개발 패턴으로 이 패턴을 성공적으로 사용하면 사용자 인터페이스로부터 비즈니스 로직을 분리하여 애플리케이션의 시각적인 요소나 그 이면에서 실행되는 비즈니스 로직을 서로 영향없이 쉽게 유지 보수할 수 있는 애플리케이션을 만들 수 있다.

MVC에서 Model은 애플리케이션에서 처리하는 데이터를 나타내며 View는 사용자 인터페이스 요소를 나타내고 Controller는 데이터와 비즈니스 로직 사이의 상호동작을 관리한다. 요청과 응답의 동작으로 이루어지는 웹은 MVC 패턴을 기반으로 웹 애플리케이션 개발 시 요청 로직과 응답 로직을 나누어 개발해서 요청 로직은 Controller로 응답 로직은 View로 그리고 처리 데이터는 Model로 개발하여 확장성과 유지보수성을 향상시킬 수 있게 되어 이미 많은 사이트들이 이 MVC 패턴을 적용하여 개발되고 운용되고 있다.

장고 프레임 워크에서는 View를 Template, Controller는 View라고 표현하며, MVC를 MTV라고 한다. Model은 데이터베이스에 저장되는 데이터의 영역 Template은 사용자에게 보여지는 영역 View는 실질적으로 프로그램 로직이 동작하여 적절한 처리 결과를 Template에게 전달하는 역할을 수행한다.

소프트웨어 디자인 패턴
MVC

django

Model

Model

View

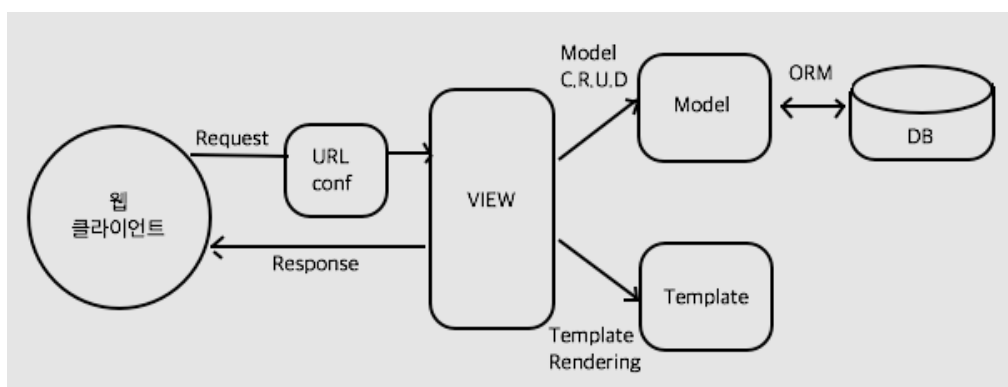
Template

Controller

View



- 장고의 처리 흐름



웹 클라이언트의 요청을 받아 장고에서 MTV 모델에 따라 처리하는 과정이다

- 클라이언트로부터 요청을 받으면 URLconf 모듈을 이용하여 URL을 분석한다.
- URL 분석 결과를 통해 해당 URL에 매칭되는 뷰를 실행한다.
- 뷰는 자신의 로직을 실행하고, 데이터베이스 처리가 필요하면 모델을 통해 처리하고 그 결과를 반환 받는다.

- 뷰는 자신의 로직 처리가 끝나면 템플릿을 사용하여 클라이언트에 전송할 HTML 파일을 생성한다.
- 뷰는 최종 결과로 HTML 파일을 클라이언트에게 보내 응답한다.
- 장고 개발환경 구축

가상 환경(virtual environment)

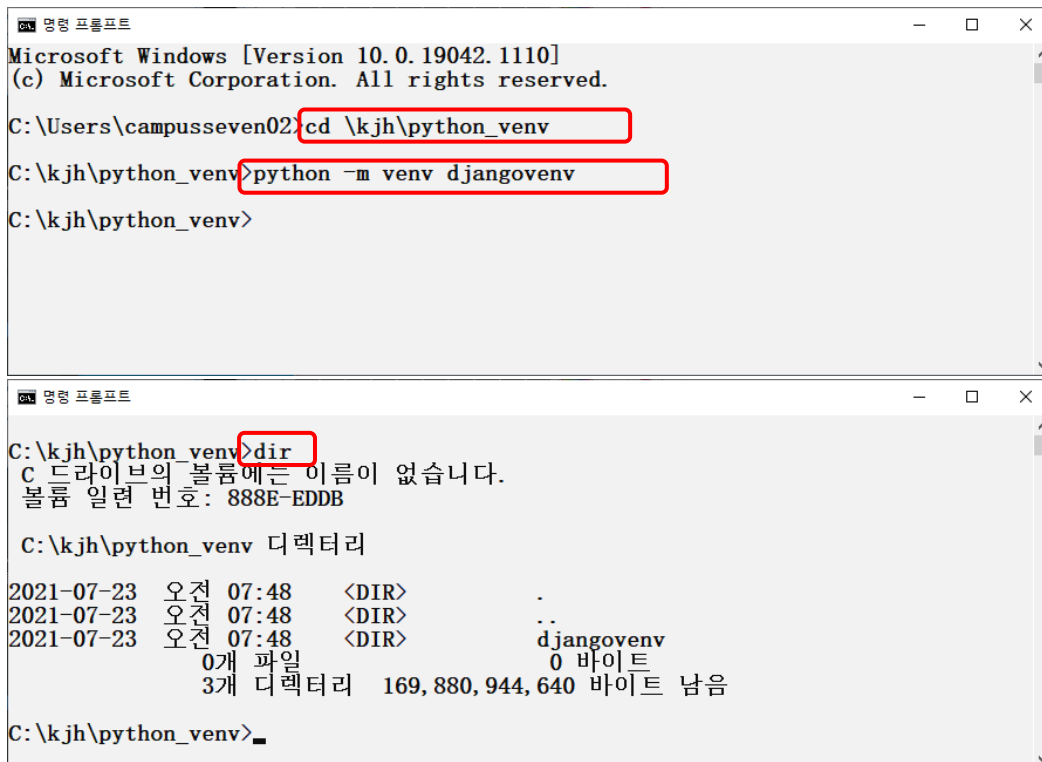
파이썬에서 가상 환경(virtual environment)은 하나의 PC에서 프로젝트 별로 독립된 파이썬 실행 환경(runtime/interpreter)을 사용할 수 있도록 해준다. 가상 환경을 사용하지 않으면 PC 내의 모든 프로젝트에서 운영체제에 설치된 하나의 파이썬 런타임을 사용하게 되고 동일한 위치에 외부 패키지를 설치하고 서로 공유하게 된다. 이럴 경우, 하나의 프로젝트에서 설치한 패키지의 버전이 다른 프로젝트에서 설치한 동일 패키지의 다른 버전과 충돌을 일으킬 소지가 생기기 때문에, 프로젝트 별로 또는 구현하려는 기술별로 독립된 가상 환경을 구성하여 사용하는 것이 권장된다.

파이썬에서 외부 패키지를 설치할 때는 일반적으로 pip이라는 패키지 매니저를 사용하는데, 기본적으로 운영체제에 파이썬이 설치된 위치의 site-packages 디렉터리에 안에 설치된다. 파이썬의 가상 환경을 이용하면 프로젝트 별로 따로 패키지를 설치하고, 다른 프로젝트로 부터 격리시킬 수 있기 때문에 시스템 전역 패키지 설치로 인한 불필요한 이슈를 방지할 수 있다. 파이썬 3.3 부터는 venv 모듈이 내장되기 때문에 별도 패키지 설치없이 파이썬만 설치되어 있으면 바로 가상 환경 구성이 가능하게 되었다.

[가상환경 만들기과 가상환경 안에 장고 개발 환경 설치하기]

(1) c:\xxx\python_venv 라는 폴더를 생성한다.

(2) 명령프롬프트 창에서 c:\xxx\python_venv 폴더로 옮겨 다음 명령을 입력하고 실행시켜 가상 환경, djangoenv 를 생성한다.



```

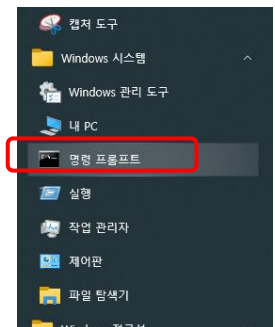
Microsoft Windows [Version 10.0.19042.1110]
(c) Microsoft Corporation. All rights reserved.

C:\Users\campussevent02>cd \kjh\python_venv
C:\kjh\python_venv>python -m venv djangoenv
C:\kjh\python_venv>

C:\kjh\python_venv>dir
C 드라이브의 볼륨에는 이름이 없습니다.
볼륨 일련 번호: 888E-EDDB

C:\kjh\python_venv 디렉터리
2021-07-23 오전 07:48 <DIR>      .
2021-07-23 오전 07:48 <DIR>      ..
2021-07-23 오전 07:48 <DIR>      djangoenv
                0개 파일              0 바이트
                3개 디렉터리  169,880,944,640 바이트 남음

C:\kjh\python_venv>
  
```



(3) 생성된 djangoenv 폴더의 Scripts 라는 폴더로 옮겨서 activate 프로그램을 실행하여 가상환경을 활성화 한다.


```
명령 프롬프트
C:\kjh\python_venv>cd djangoenv\Scripts
C:\kjh\python_venv\djangoenv\Scripts>activate.
```

(4) djangoenv라는 가상환경을 활성화하면 다음과 같이 명령 프롬프트 맨 앞에 (djangoenv)가 붙은 것을 볼 수 있다.

```
명령 프롬프트
(djangoenv) C:\kjh\python_venv\djangoenv\Scripts>.
```

이렇게 가상환경을 활성화한 상태에서 이 가상환경 안에 장고 개발 환경을 설치하기 위해 pip 명령을 실행한다.

```
명령 프롬프트
(djangoenv) C:\kjh\python_venv\djangoenv\Scripts>pip install django
```

```
C:\WINDOWS\system32\cmd.exe
(djangoenv) C:\kjh\python_venv\djangoenv\Scripts>pip install django
Collecting django
  Downloading Django-4.1.1-py3-none-any.whl (8.1 MB)
    ----- 8.1/8.1 MB 39.8 MB/s eta 0:00:00
Collecting asgiref<4,>=3.5.2
  Downloading asgiref-3.5.2-py3-none-any.whl (22 kB)
Collecting tzdata
  Downloading tzdata-2022.4-py2.py3-none-any.whl (336 kB)
    ----- 336.7/336.7 KB 20.4 MB/s eta 0:00:00
Collecting sqlparse>=0.2.2
  Downloading sqlparse-0.4.3-py3-none-any.whl (42 kB)
    ----- 42.8/42.8 KB ? eta 0:00:00
Installing collected packages: tzdata, sqlparse, asgiref, django
Successfully installed asgiref-3.5.2 django-4.1.1 sqlparse-0.4.3 tzdata-2022.4
WARNING: You are using pip version 22.0.4; however, version 22.2.2 is available.
You should consider upgrading via the 'C:\kjh\python_venv\djangoenv\Scripts\python.exe -m pip install --upgrade pip' command.
(djangoenv) C:\kjh\python_venv\djangoenv\Scripts>
```

위의 화면은 장고 개발환경이 설치되는 화면이다. 그런데 제일 아래에 보면 우리 시스템에 설치된 pip 의 버전 관해서 경고 오류가 출력되는 것을 볼 수 있다. 다음 명령을 실행시켜 업그레이드 해놓는다.(명령어 : `python -m pip install --upgrade pip`)

```
C:\WINDOWS\system32\cmd.exe
(djangoenv) C:\kjh\python_venv\djangoenv\Scripts>python -m pip install --upgrade pip

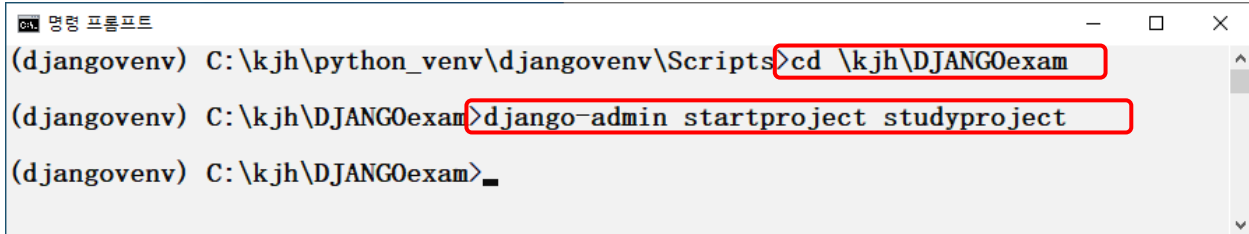
Requirement already satisfied: pip in c:\kjh\python_venv\djangoenv\lib\site-packages (22.0.4)
Collecting pip
  Downloading pip-22.2.2-py3-none-any.whl (2.0 MB)
    ----- 2.0/2.0 MB 25.8 MB/s eta 0:00:00
Installing collected packages: pip
  Attempting uninstall: pip
    Found existing installation: pip 22.0.4
    Uninstalling pip-22.0.4:
      Successfully uninstalled pip-22.0.4
  Successfully installed pip-22.2.2
(djangoenv) C:\kjh\python_venv\djangoenv\Scripts>
```

[장고 프로젝트 만들기]

(1) `c:\xxx\DJANGOexam` 이라는 폴더를 생성한다.

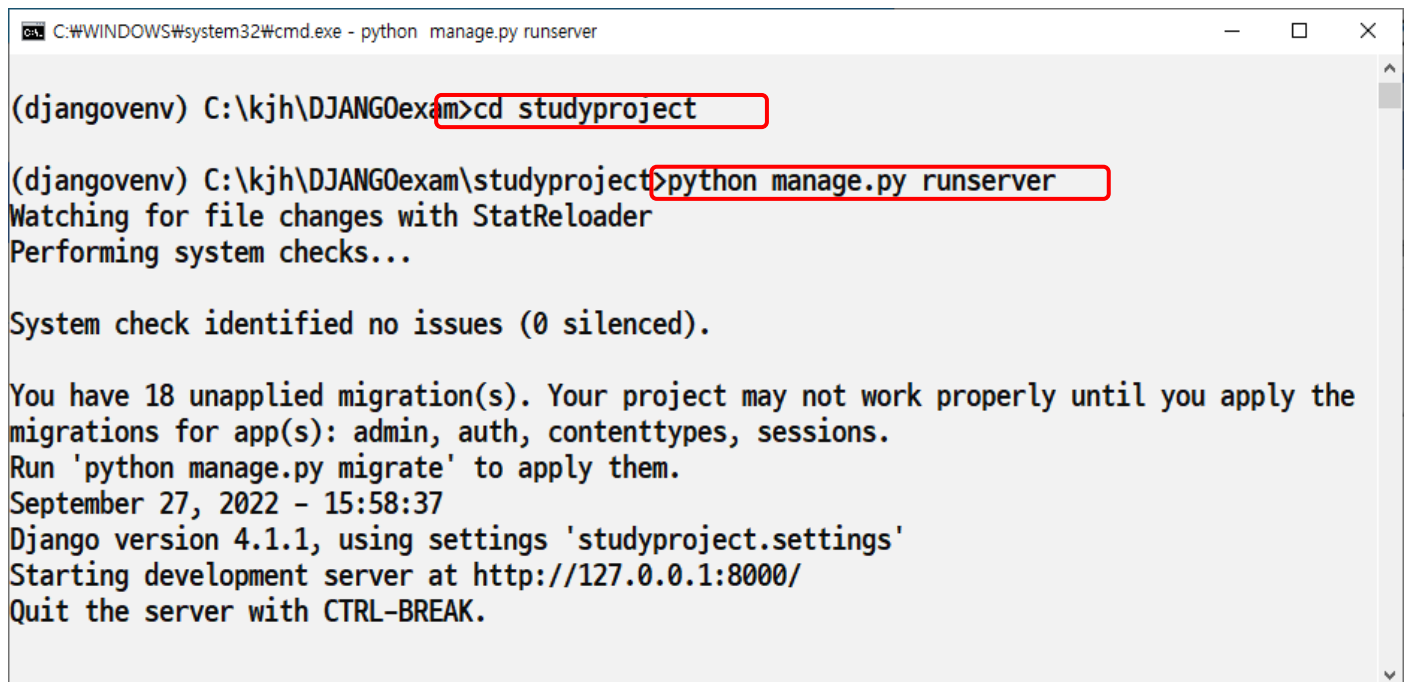
(2) 명령프롬프트 창에서 `c:\xxx\DJANGOexam` 폴더로 이동해서 다음 명령을 입력하고 실행시켜 학습용 장고 프로젝트를 생성한다.

(이동하는 명령 : `cd c:\xxx\DJANGOexam`)



```
(djangoven) C:\kjh\python_venv\djangoven\Scripts>cd \kjh\DJANGOexam
(djangoven) C:\kjh\DJANGOexam>django-admin startproject studyproject
(djangoven) C:\kjh\DJANGOexam>
```

(3) 생성된 `studyproject` 폴더로 이동해서 `python manage.py runserver` 를 입력하고 실행시켜 장고에 내장된 서버를 기동시킨다. (이동하는 명령 : `cd studyproject`)

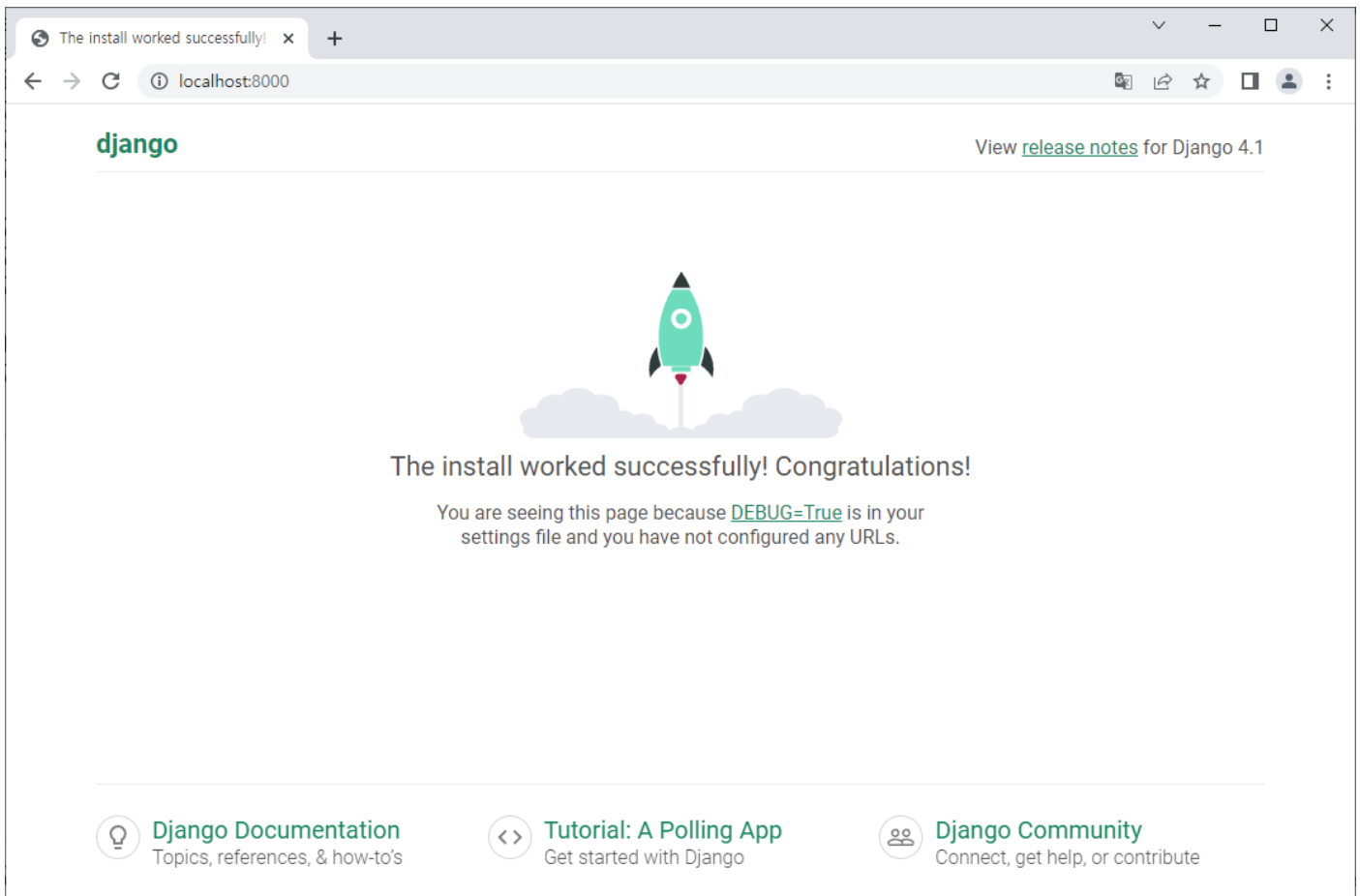


```
C:\WINDOWS\system32\cmd.exe - python manage.py runserver
(djangoven) C:\kjh\DJANGOexam>cd studyproject
(djangoven) C:\kjh\DJANGOexam\studyproject>python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).

You have 18 unapplied migration(s). Your project may not work properly until you apply the
migrations for app(s): admin, auth, contenttypes, sessions.
Run 'python manage.py migrate' to apply them.
September 27, 2022 - 15:58:37
Django version 4.1.1, using settings 'studyproject.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

(4) 크롬 브라우저에서 `http://localhost:8000/` 입력하여 요청하면 장고의 기본 웹 페이지가 출력되는 것을 볼 수 있다.



-----→ 이 화면이 잘 보인다면!!! 추카추카!!

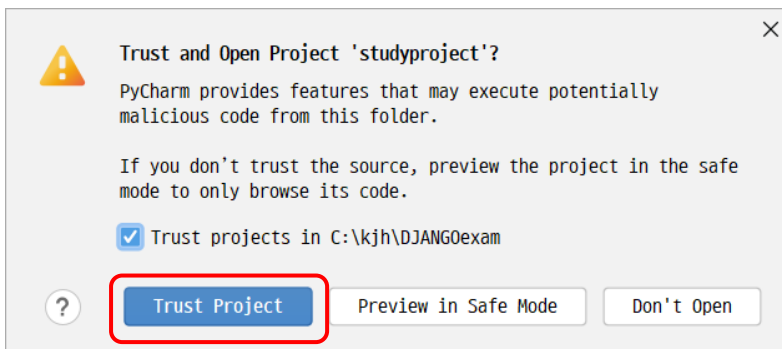
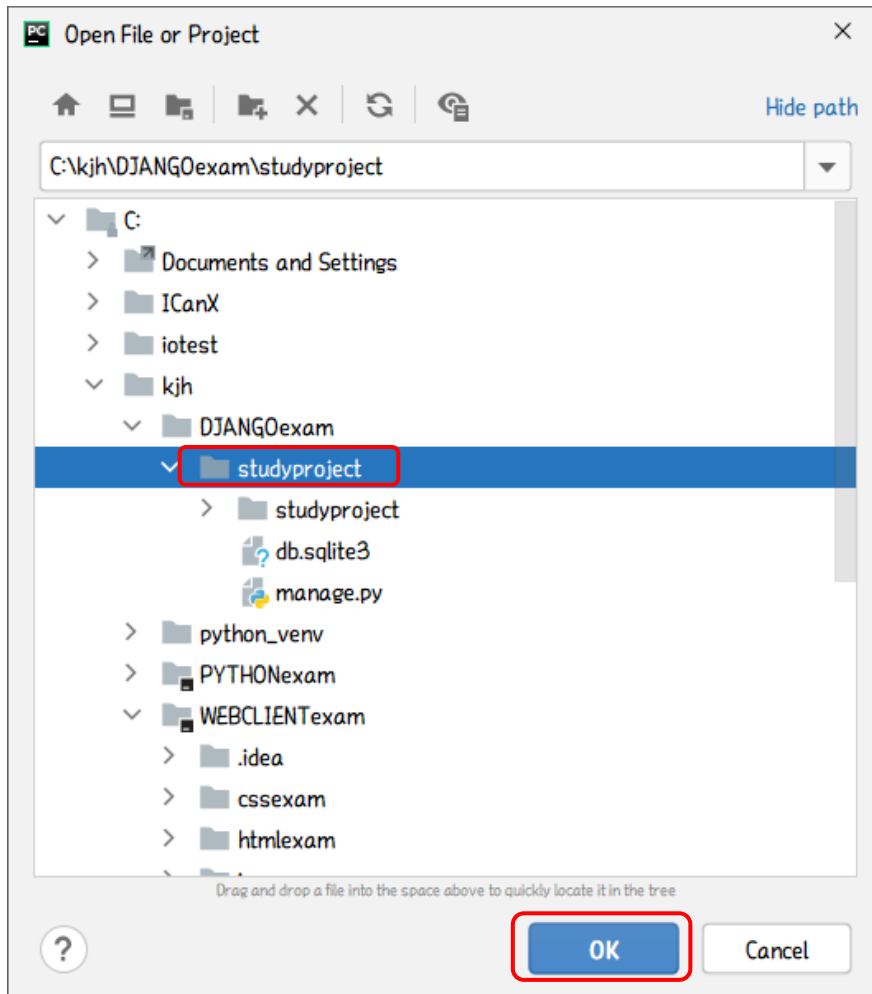


cmd 창에서 ctrl+c를 입력하여 서버를 종료한다.

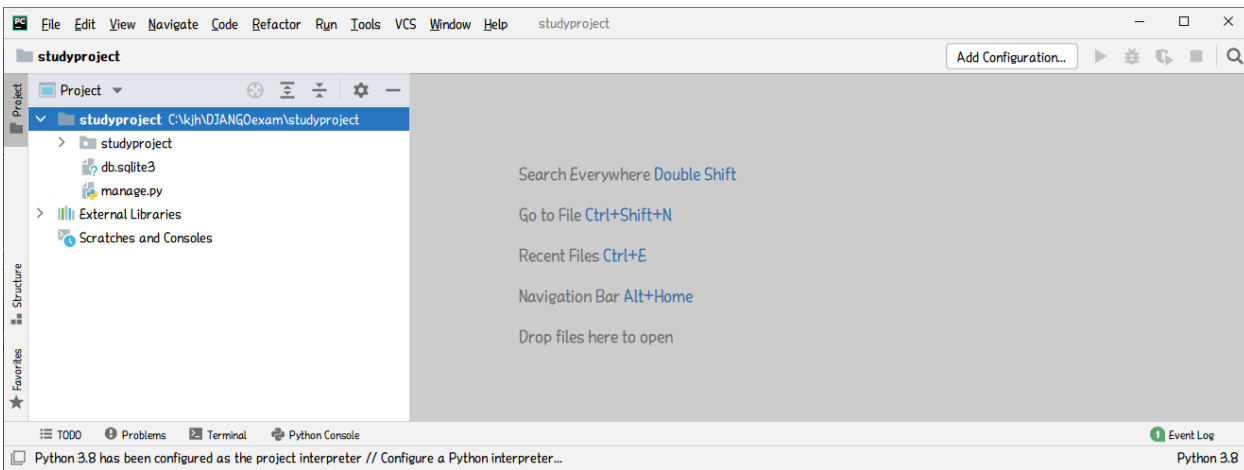
```
C:\WINDOWS\system32\cmd.exe - python manage.py runserver
migrations for app(s): admin, auth, contenttypes, sessions.
Run 'python manage.py migrate' to apply them.
September 27, 2022 - 15:58:37
Django version 4.1.1, using settings 'studyproject.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
[27/Sep/2022 15:59:21] "GET / HTTP/1.1" 200 10681
[27/Sep/2022 15:59:22] "GET /static/admin/css/fonts.css HTTP/1.1" 200 423
[27/Sep/2022 15:59:22] "GET /static/admin/fonts/Roboto-light-webfont.woff HTTP/1.1" 200 856
92
[27/Sep/2022 15:59:22] "GET /static/admin/fonts/Roboto-Regular-webfont.woff HTTP/1.1" 200 8
5876
[27/Sep/2022 15:59:22] "GET /static/admin/fonts/Roboto-Bold-webfont.woff HTTP/1.1" 200 8618
4
Not Found: /favicon.ico
[27/Sep/2022 15:59:22] "GET /favicon.ico HTTP/1.1" 404 2116
```

이제부터는 생성된 프로젝트를 파이참으로 데리고 들어간다.

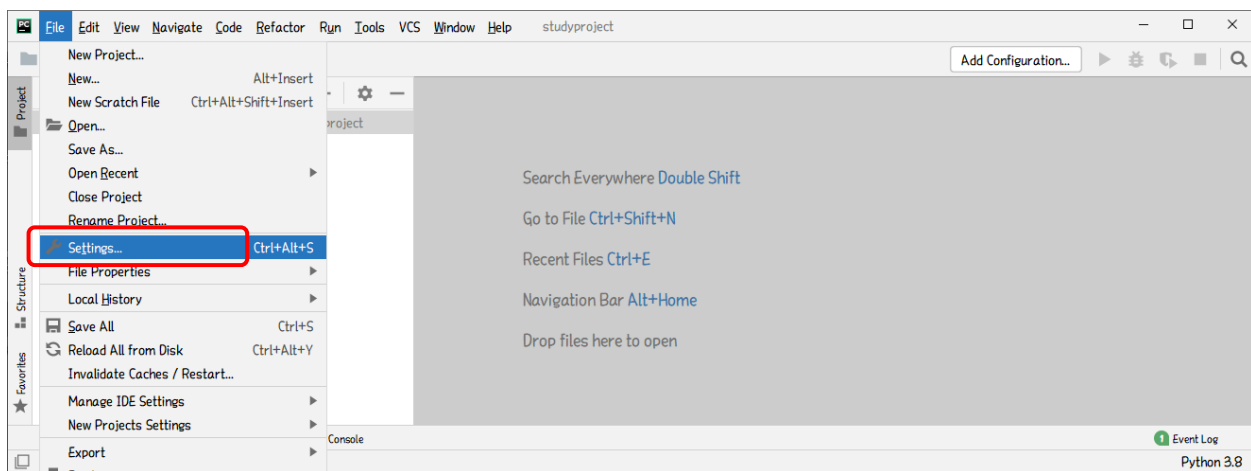
(1) 파이참에서 File>Open 을 선택하고 c:\xxx\DJANDOexam\studyproject을 오픈한다. 다음 화면을 참조한다.



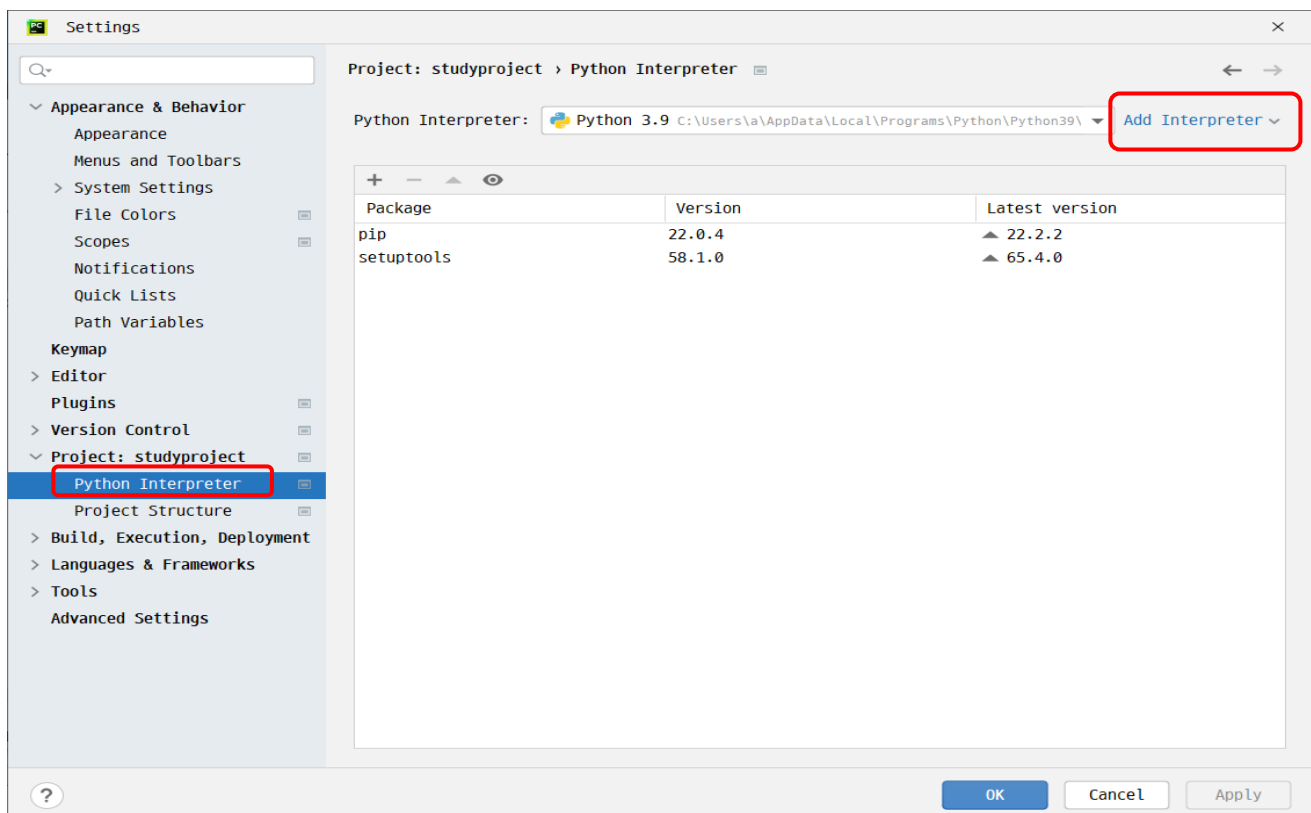
오픈이 잘 진행되면 다음과 같이 studyproject 가 인식되는 것을 볼 수 있다.

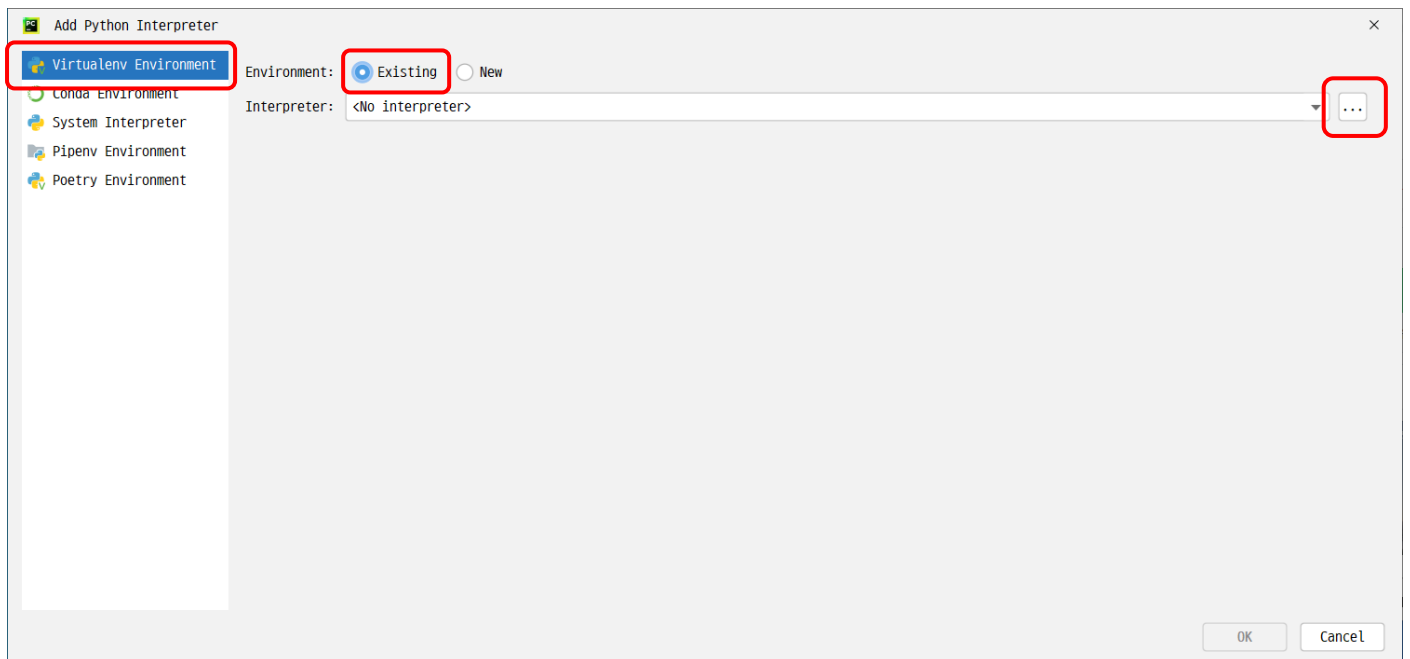
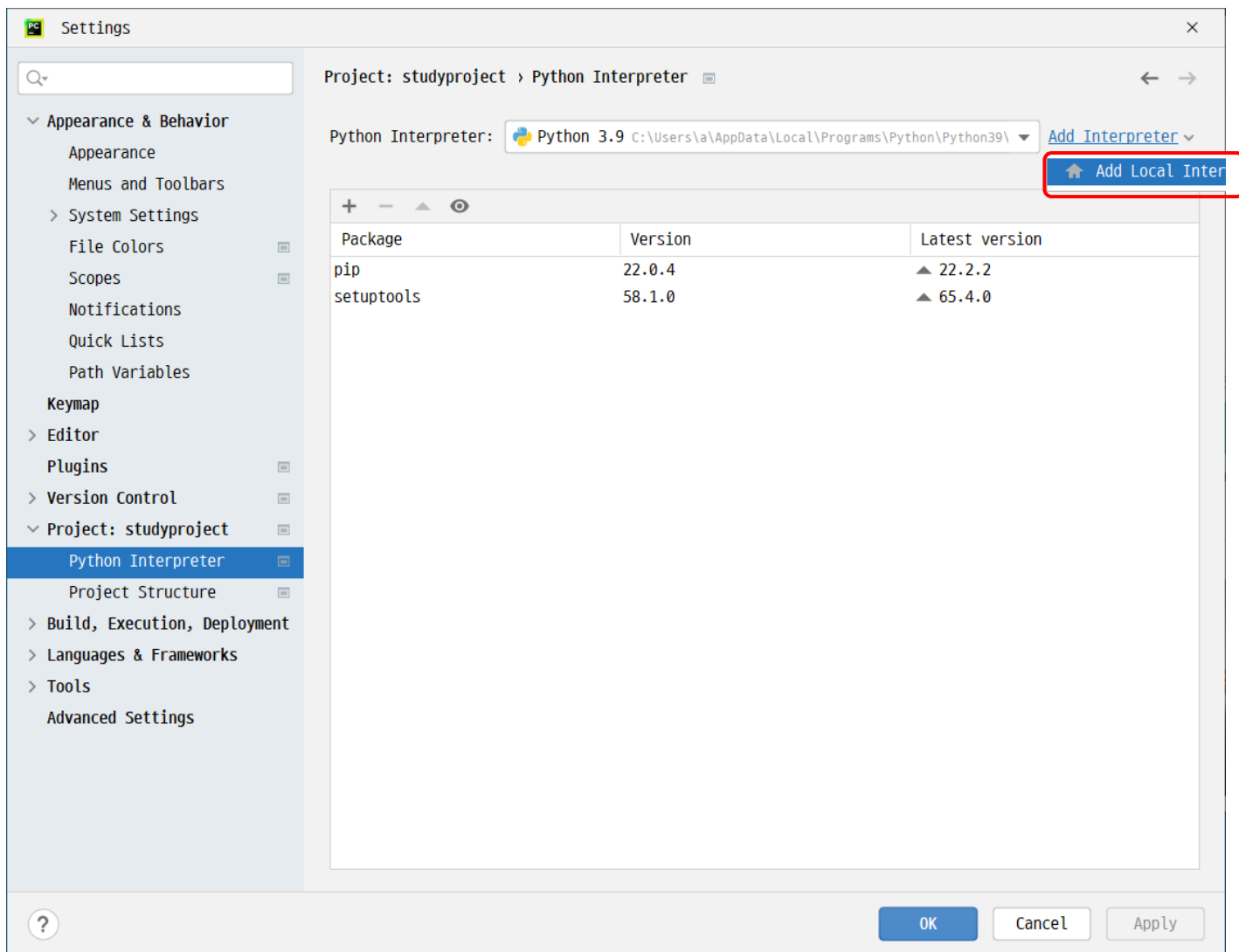


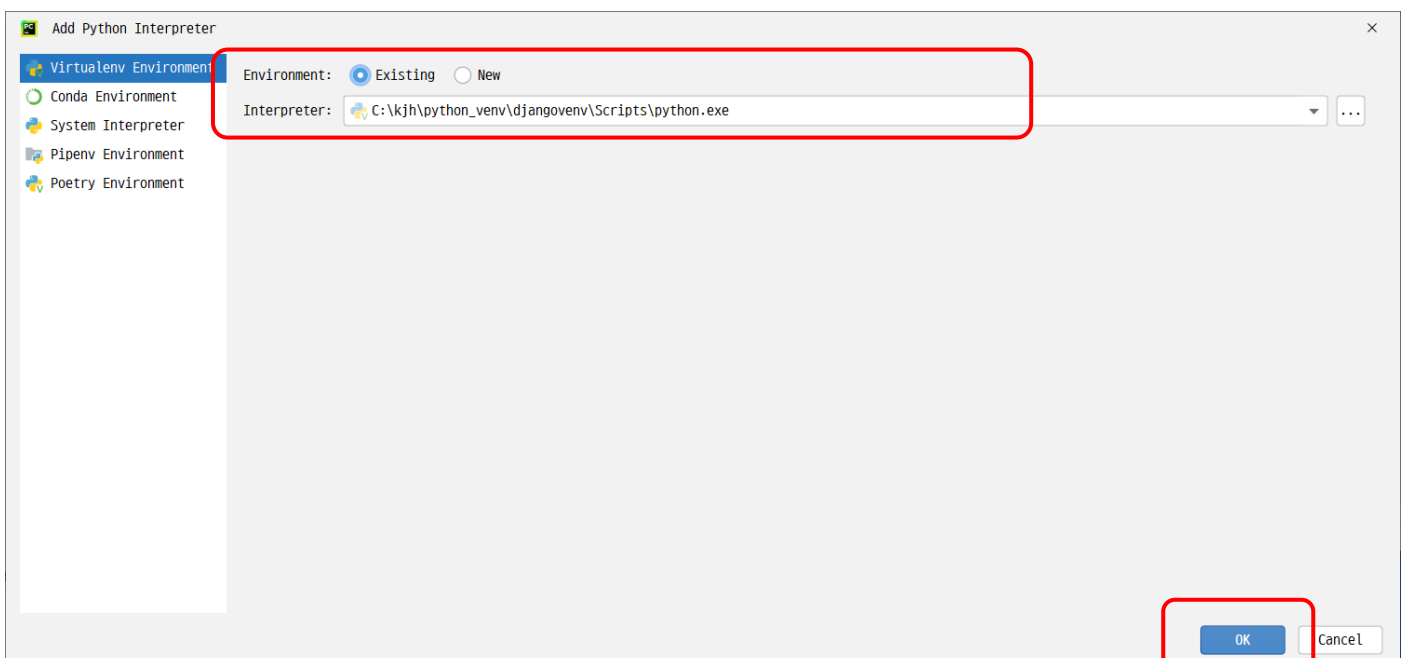
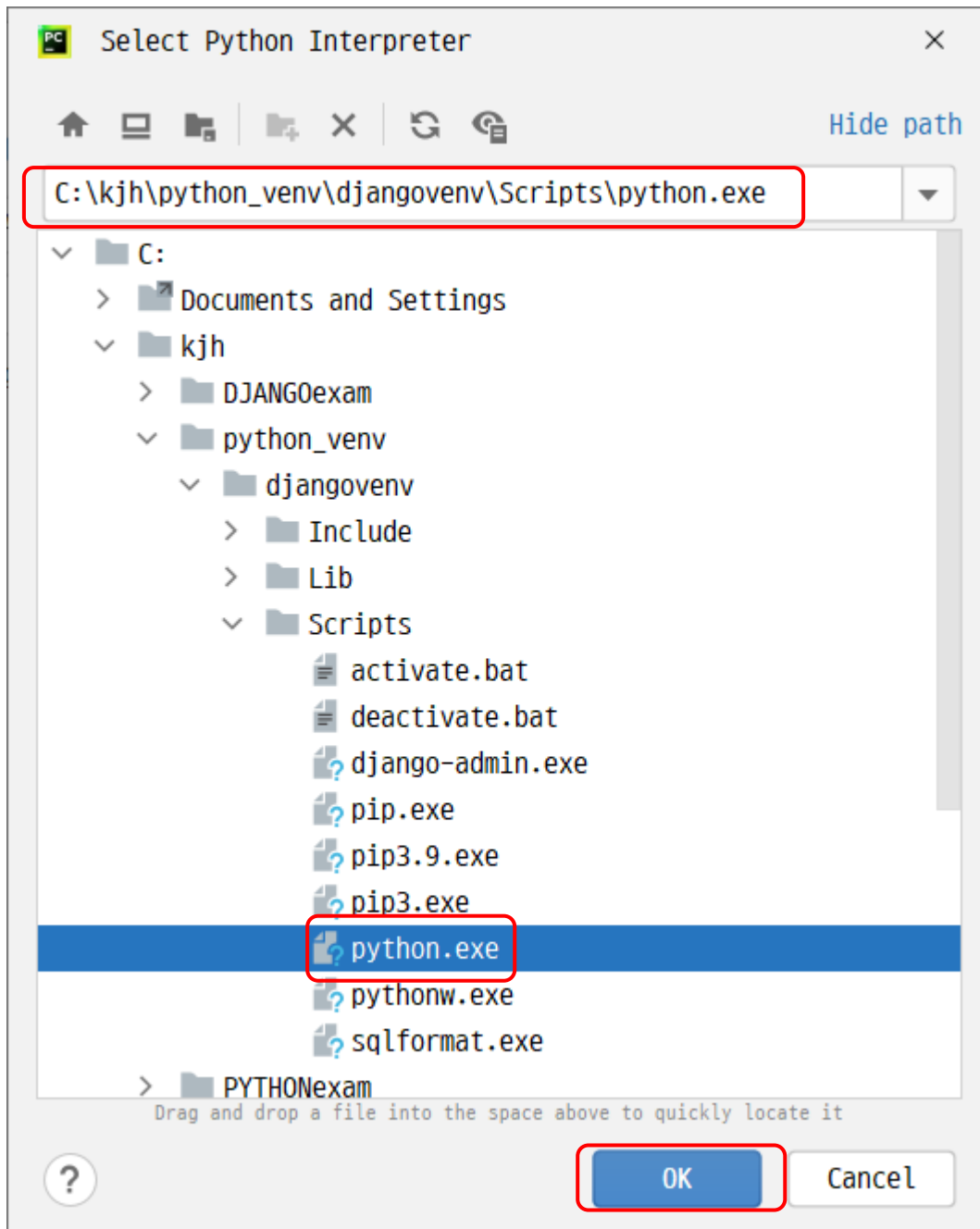
(2) **File>Settings** 를 선택하고 **studyproject**의 프로젝트 인터프리터를 가상환경으로 선택한다. 다음 화면과 같이 선택하고 OK 버튼을 클릭한다.

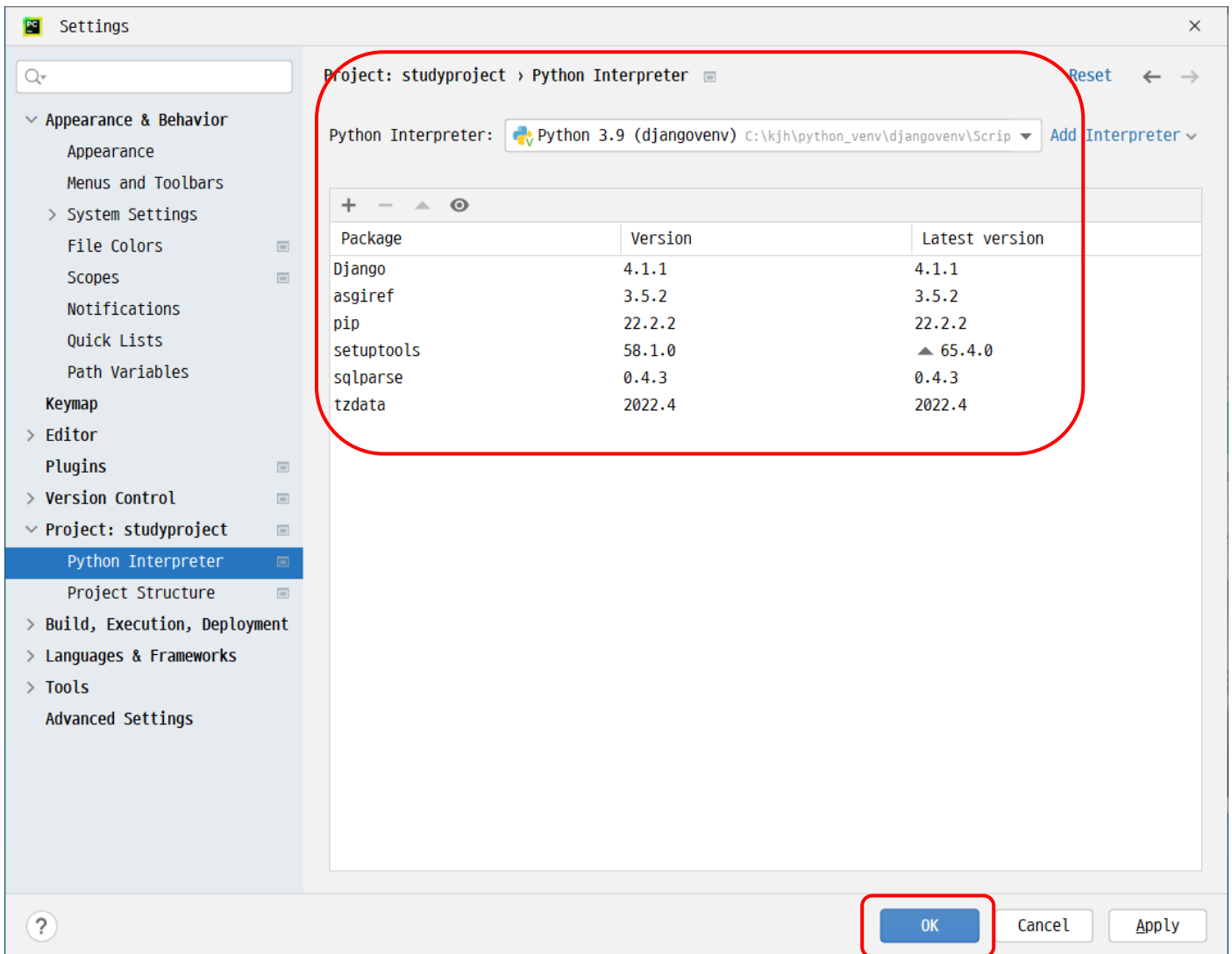


다음과 같은 서브윈도우가 출력되면 **Project: studyproject** 의 **Python Interpreter** 를 선택한다.

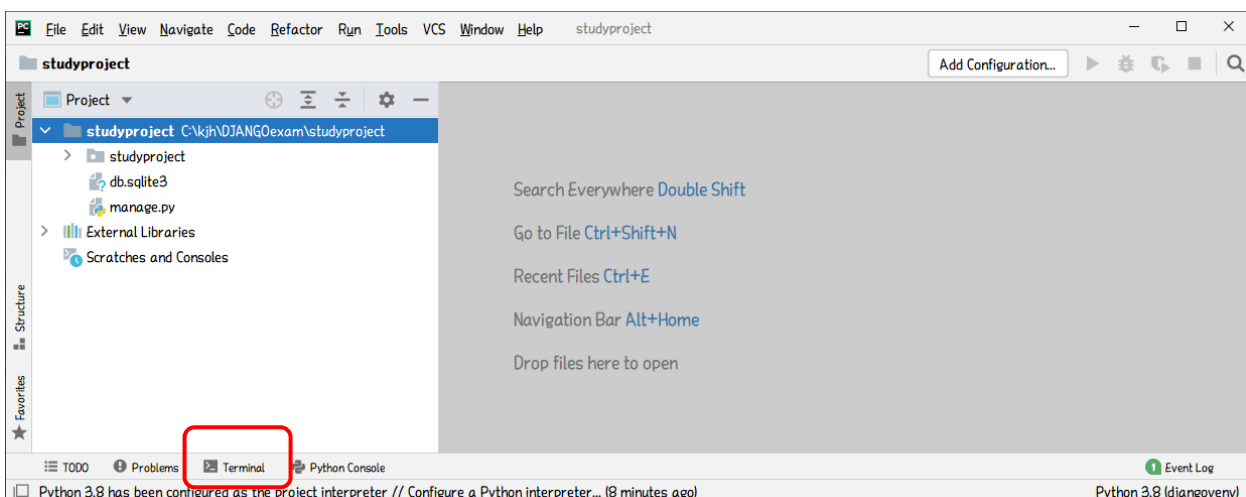




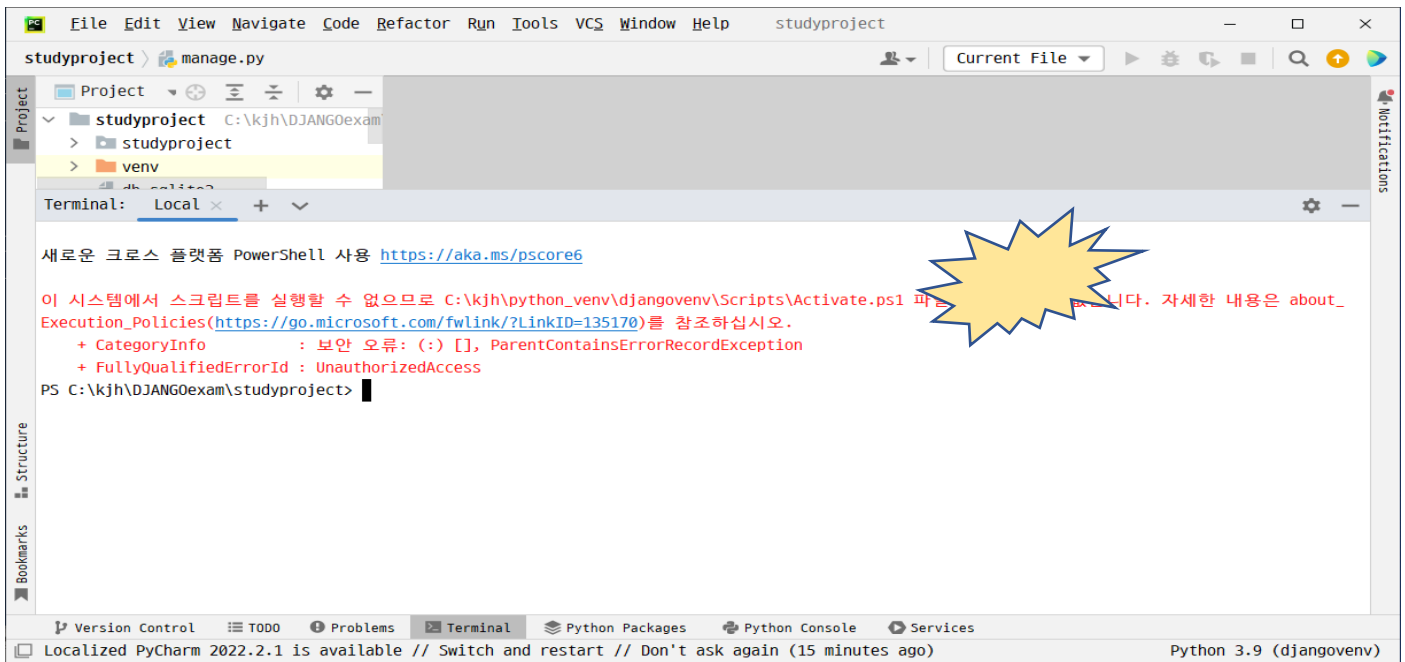




(3) 환경 구성이 끝날 때까지 잠시 기다린 후에 파이참 하단 왼쪽에 있는 **Terminal**을 선택하여 cmd 창과 비슷한 창을 출력한다.

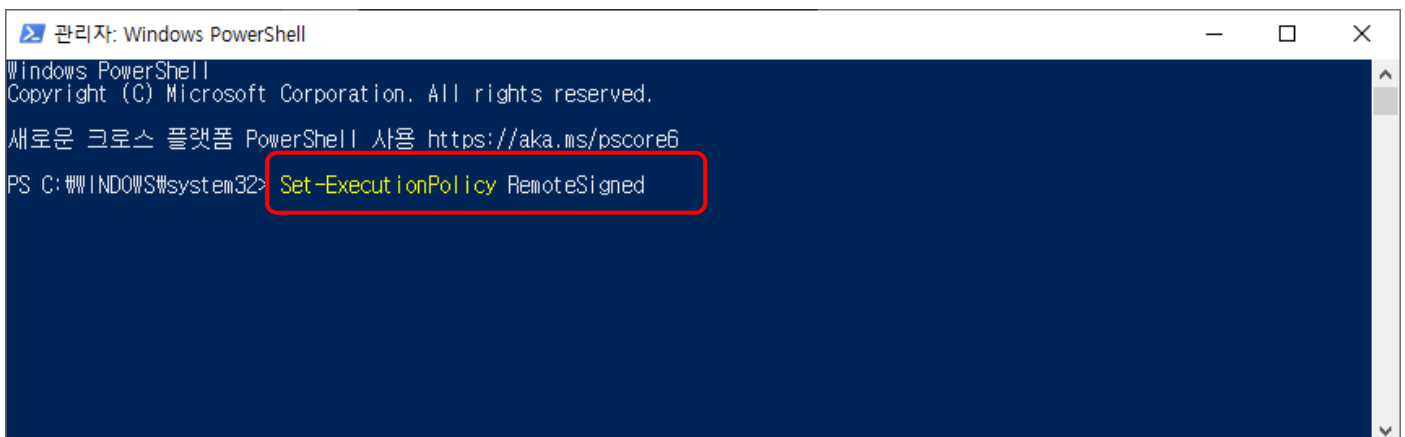


파이참에서는 파워셸을 사용하는데 마이크로소프트사의 보안 정책에 의해서 다음과 같이 보안오류가 발생하는 것을 볼 수 있다.
해결방법은 존재한다. 보안 정책을 변경하는 것이다.



시작메뉴에서 오른쪽 버튼을 클릭하면 출력되는 메뉴에서 파워셸을 관리자 모드로 실행한다.

다음 명령을 입력하고 실행시킨다.



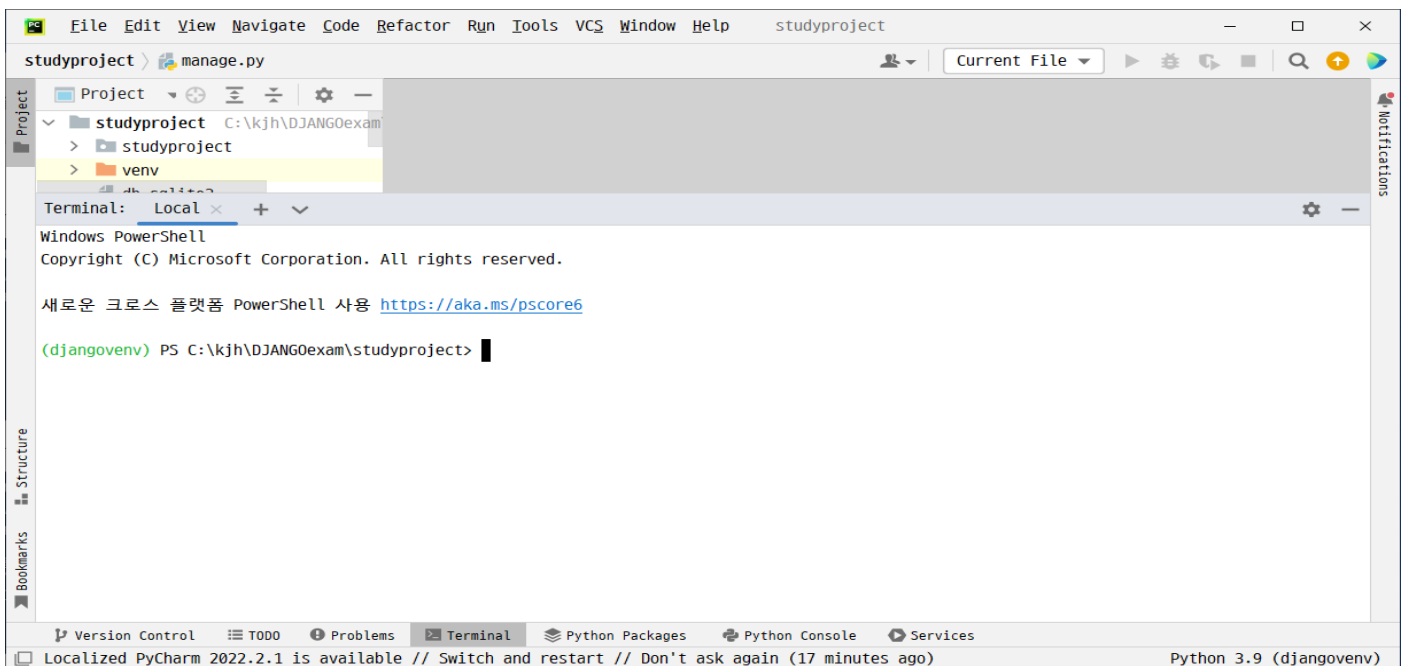
```
관리자: Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

새로운 크로스 플랫폼 PowerShell 사용 https://aka.ms/pscore6

PS C:\WINDOWS\system32> Set-ExecutionPolicy RemoteSigned

실행 규칙 변경
실행 정책은 신뢰하지 않는 스크립트로부터 사용자를 보호합니다. 실행 정책을 변경하면 about_Execution_Policies 도움말
(https://go.microsoft.com/fwlink/?LinkID=135170)에 설명된 보안 위험에 노출될 수 있습니다. 실행 정책을
변경하시겠습니까?
[Y] 예(Y) [A] 모두 예(A) [N] 아니요(N) [L] 모두 아니요(L) [S] 일시 중단(S) [?] 도움말 (기본값은 "N") Y
PS C:\WINDOWS\system32>
```

다시 **Terminal** 을 선택해 본다. 그러면 다음과 같이 하단에 cmd 창하고 비슷하게 명령을 입력할 수 있는 파워셸화면이 출력된다. 그리고 오류가 해결된 것을 볼 수 있다.



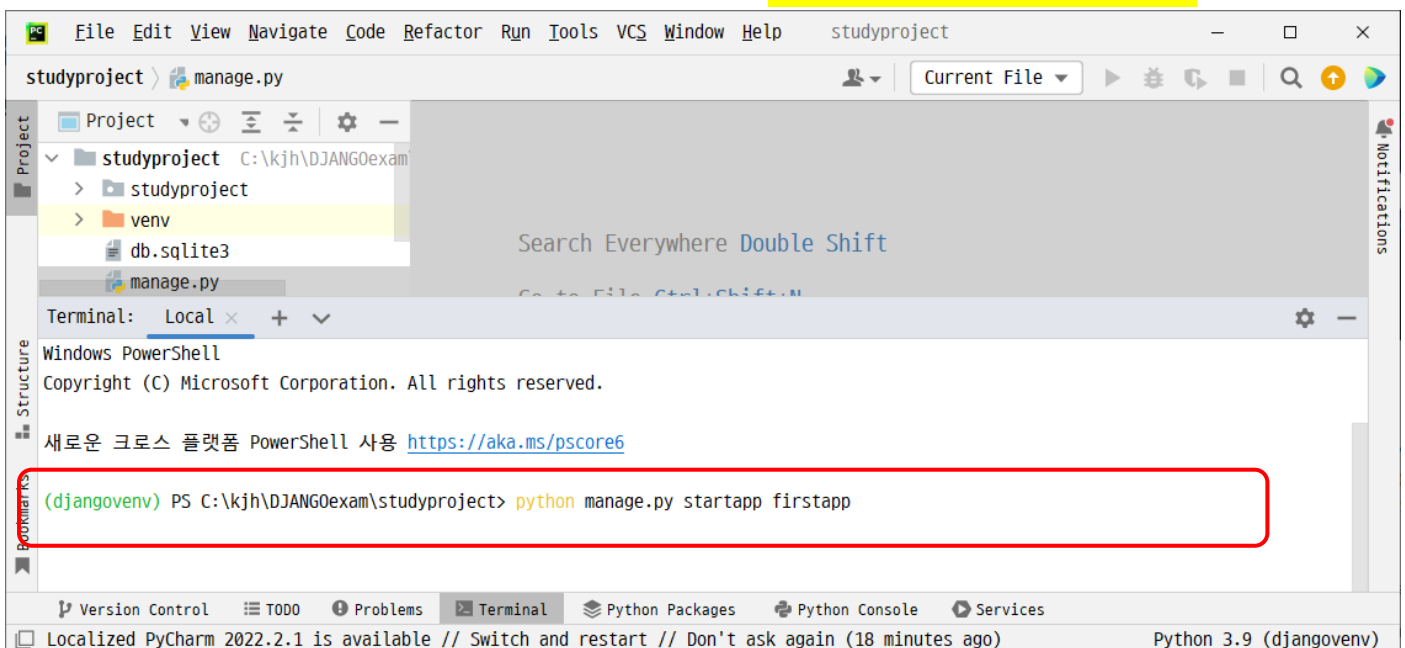
```
studyproject > manage.py

Terminal: Local x + v
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

새로운 크로스 플랫폼 PowerShell 사용 https://aka.ms/pscore6

(djangovenv) PS C:\kjh\DJANGOexam\studyproject>
```

(4) 학습용 소스들을 작성할 **firstapp** 이라는 장고 앱을 하나 생성한다. **python manage.py startapp firstapp**

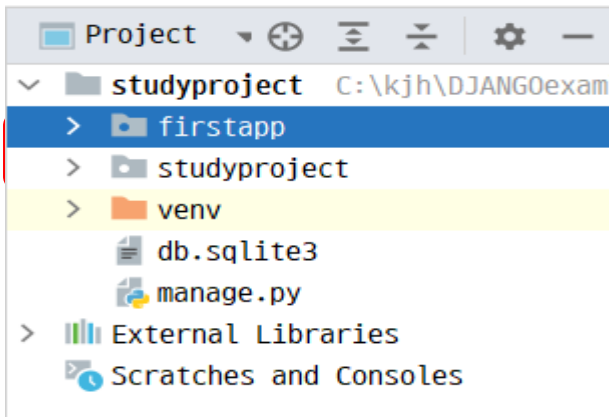


```
studyproject > manage.py

Terminal: Local x + v
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

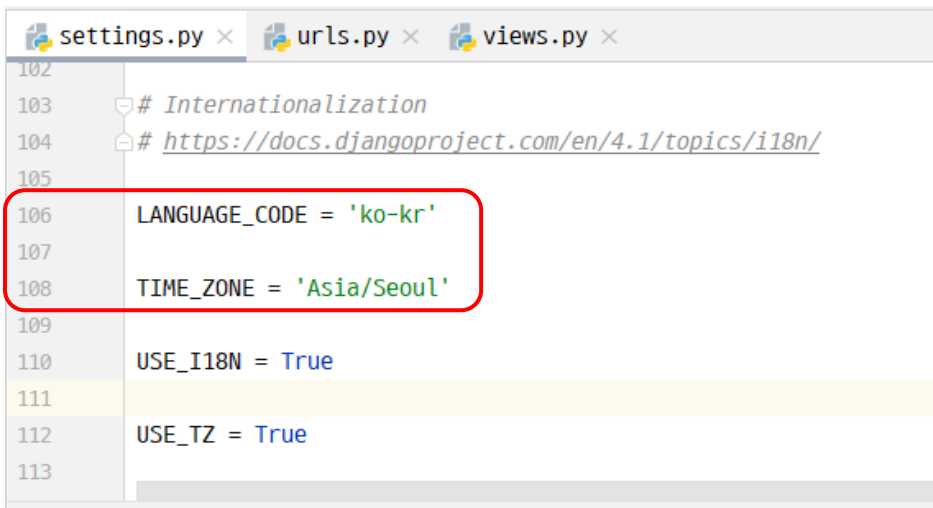
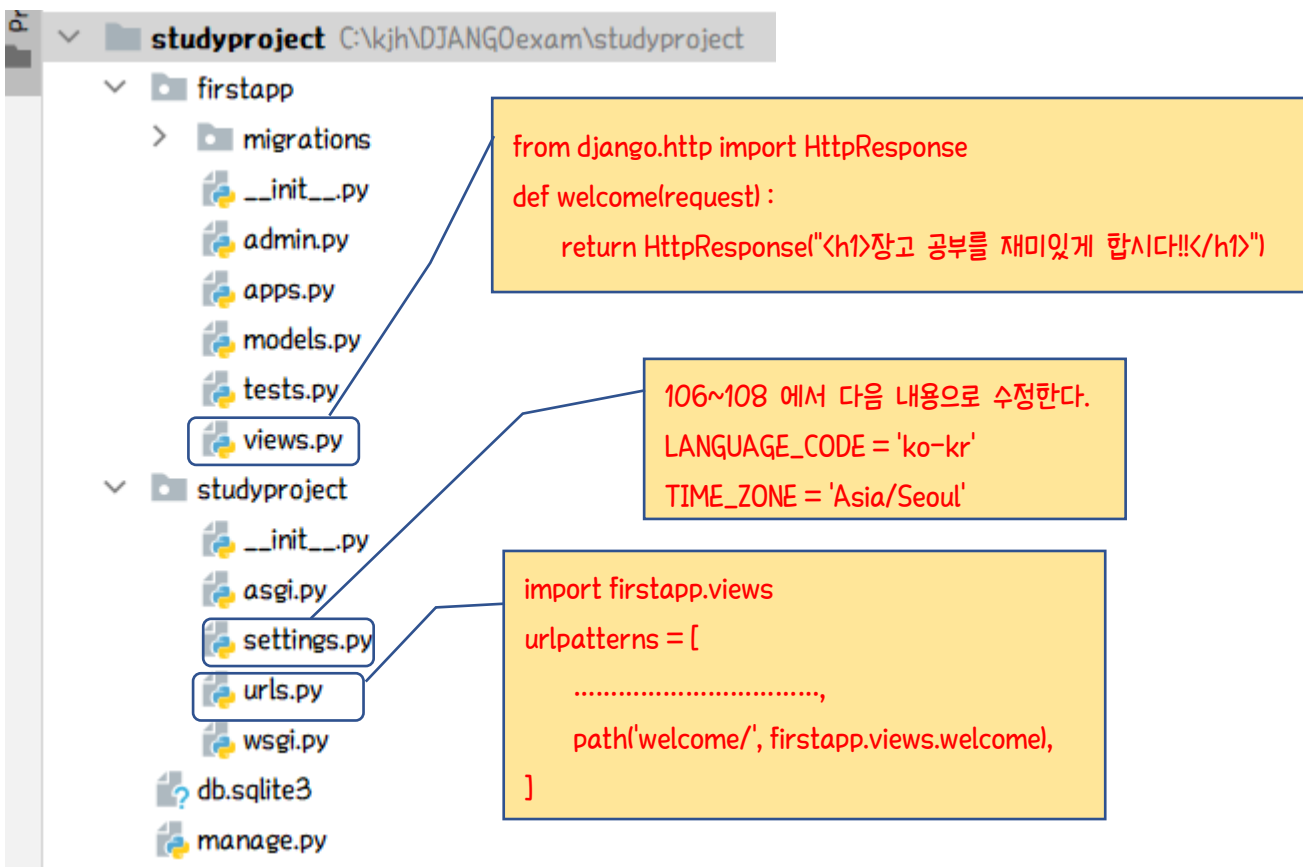
새로운 크로스 플랫폼 PowerShell 사용 https://aka.ms/pscore6

(djangovenv) PS C:\kjh\DJANGOexam\studyproject> python manage.py startapp firstapp
```



studypoint 아래에 firstapp 이라는 앱폴더가 생성된 것을 볼 수 있다. 안보이면 프로젝트를 닫았다가 다시 열어본다.

(5) firstapp 이라는 장고 앱의 폴더가 다음과 같은 디렉토리 구조로 생성된다. 표시된 파일들을 제시된 내용으로 수정한다.



```

settings.py x urls.py x views.py x
14 2. Add a URL to urlpatterns: path('blog/', includ
15
16 from django.contrib import admin
17 from django.urls import path
18 import firstapp.views
19 urlpatterns = [
20     path('admin/', admin.site.urls),
21     path('welcome/', firstapp.views.welcome),
22 ]
23

```

```

settings.py x urls.py x views.py x
1 from django.shortcuts import render
2
3 from django.http import HttpResponse
4 def welcome(request) :
5     return HttpResponse("<h1>장고 공부를 재미있게 합시다!!</h1>")
6

```

(6) settings.py, views.py, urls.py 이 세 개의 파일의 수정이 성공적으로 된 것을 확인한 후에 서버를 기동시킨다.

The screenshot shows the PyCharm IDE interface. The top pane displays the 'views.py' file with the following code:

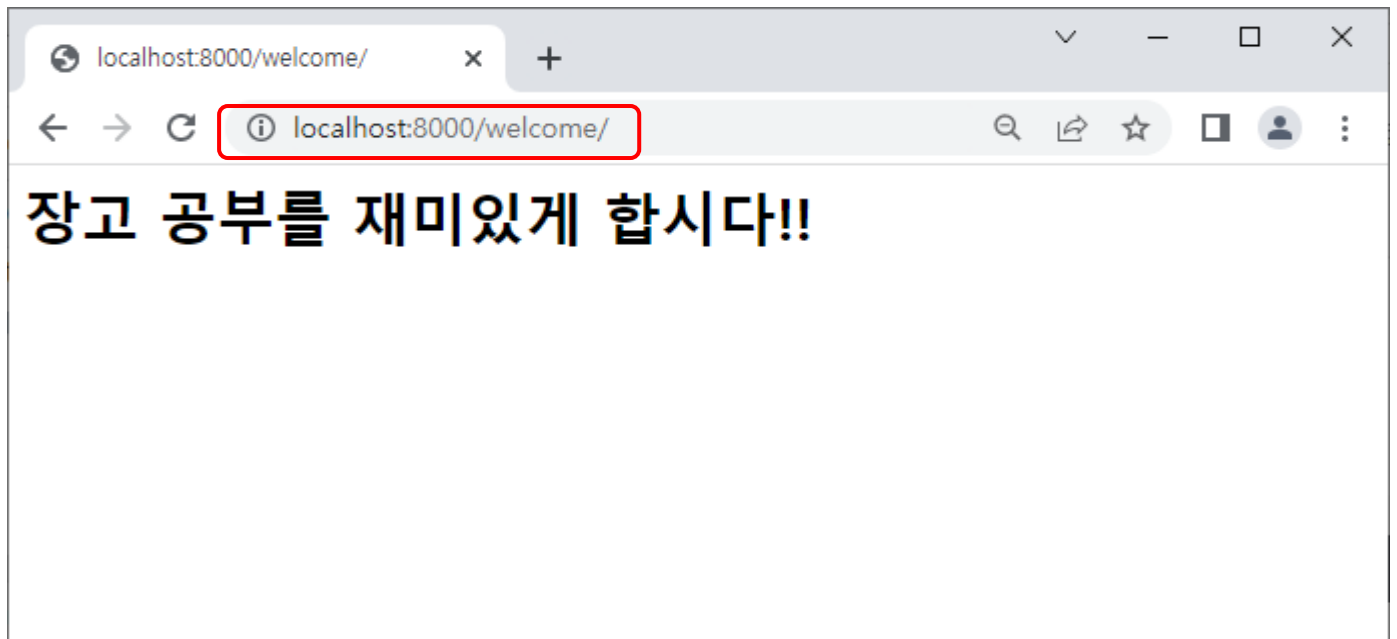
```

from django.http import HttpResponse
def welcome(request) :
    return HttpResponse("<h1>장고 공부를 재미있게 합시다!!</h1>")

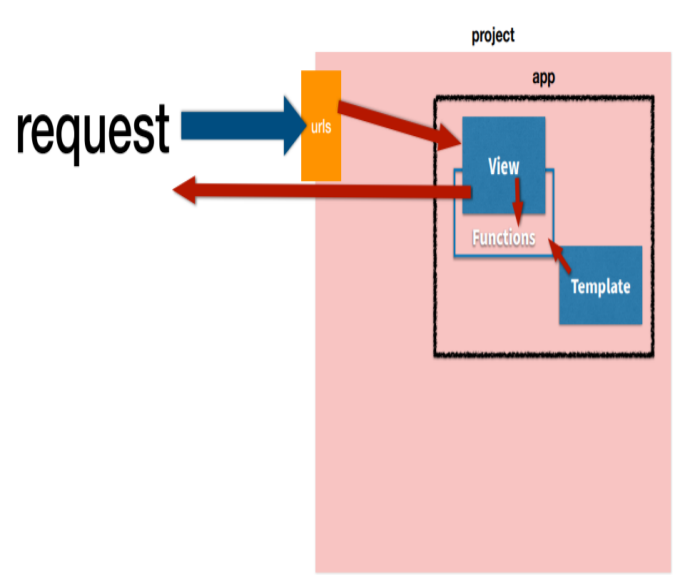
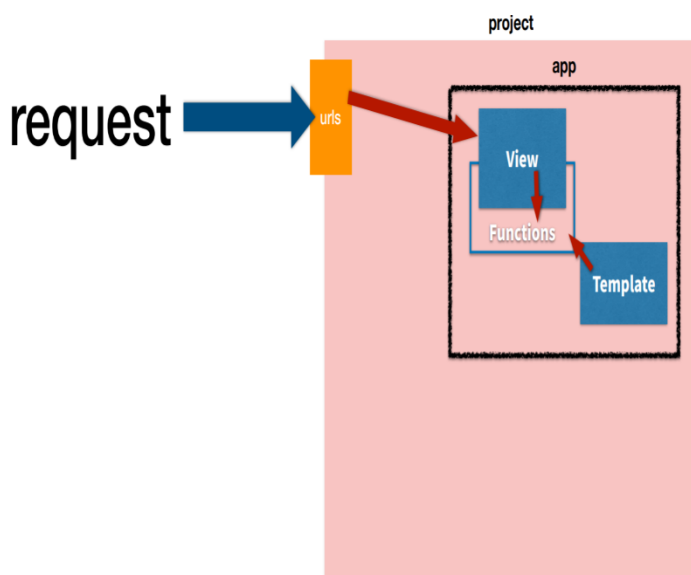
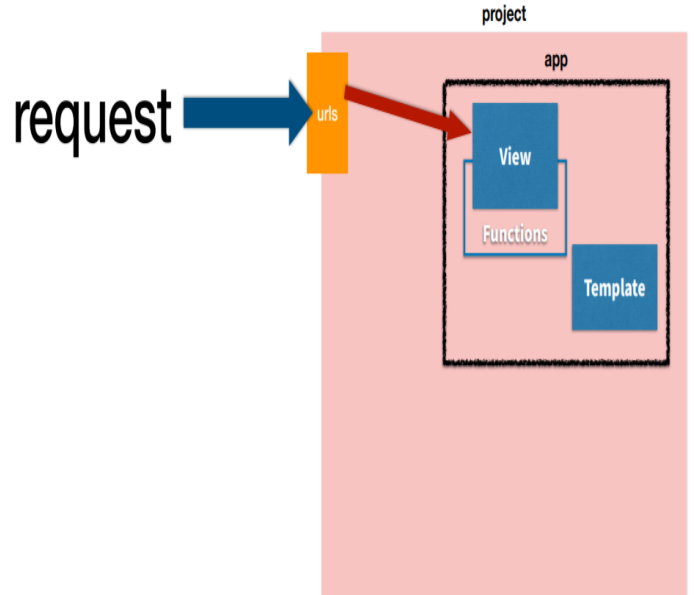
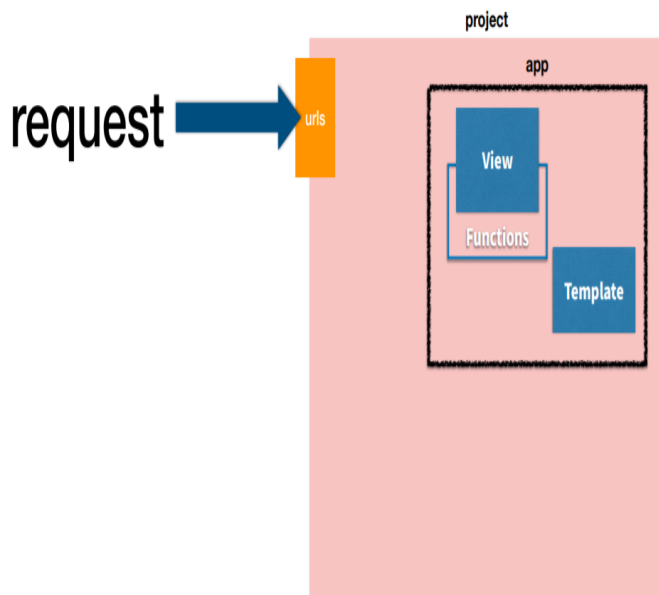
```

The bottom pane shows the terminal output of running the command `python manage.py runserver`. The output indicates that the server is running successfully at `http://127.0.0.1:8000/`.

(7) 브라우저에서 다음 URL 로 요청해 본다.(<http://localhost:8000/welcome/>)



[Views와 Templates]



urls.py - 프로젝트 폴더(메인)와 앱폴더(서브)

views.py - 앱폴더

templates\xxxx.html - 앱폴더

[장고를 이용한 웹 서버 프로그래밍]

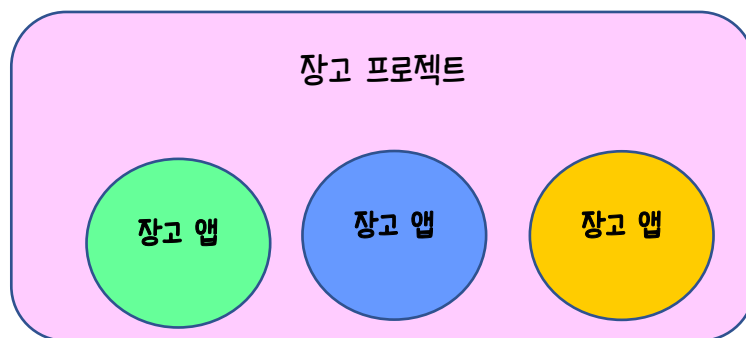
요청 시 사용될 URL 문자열의 패스와 호출될 뷰의 함수를 결정한다. - 라우팅을 정한다고 한다.	urls.py
GET 방식 요청을 처리할 것인지, POST 방식 요청을 처리할 것인지 결정하고 요청에 대한 기능을 수행하도록 위에서 설정한 함수에 로직을 구현한다.	views.py
수행한 결과를 클라이언트로 응답하는 기능의 템플릿을 구현한다.	templates/xxx.html

기동된 서버를 종료할 때는 다음과 같이 Terminal 창에서 Ctrl+C를 입력한다.

```
Terminal: Local x + v
You have 18 unapplied migration(s). Your project may not work properly until you apply the migrations for app(s): admin, auth, contenttypes, sessions.
Run 'python manage.py migrate' to apply them.
September 28, 2022 - 07:41:41
Django version 4.1.1, using settings 'studyproject.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
Not Found: /welcome/
[28/Sep/2022 07:42:04] "GET /welcome/ HTTP/1.1" 404 2218
Not Found: /favicon.ico
[28/Sep/2022 07:42:04] "GET /favicon.ico HTTP/1.1" 404 2224
[28/Sep/2022 07:42:09] "GET /welcome/ HTTP/1.1" 200 50
(djangoven) PS C:\kjh\DJANGOexam\studyproject>
```

Ctrl+C

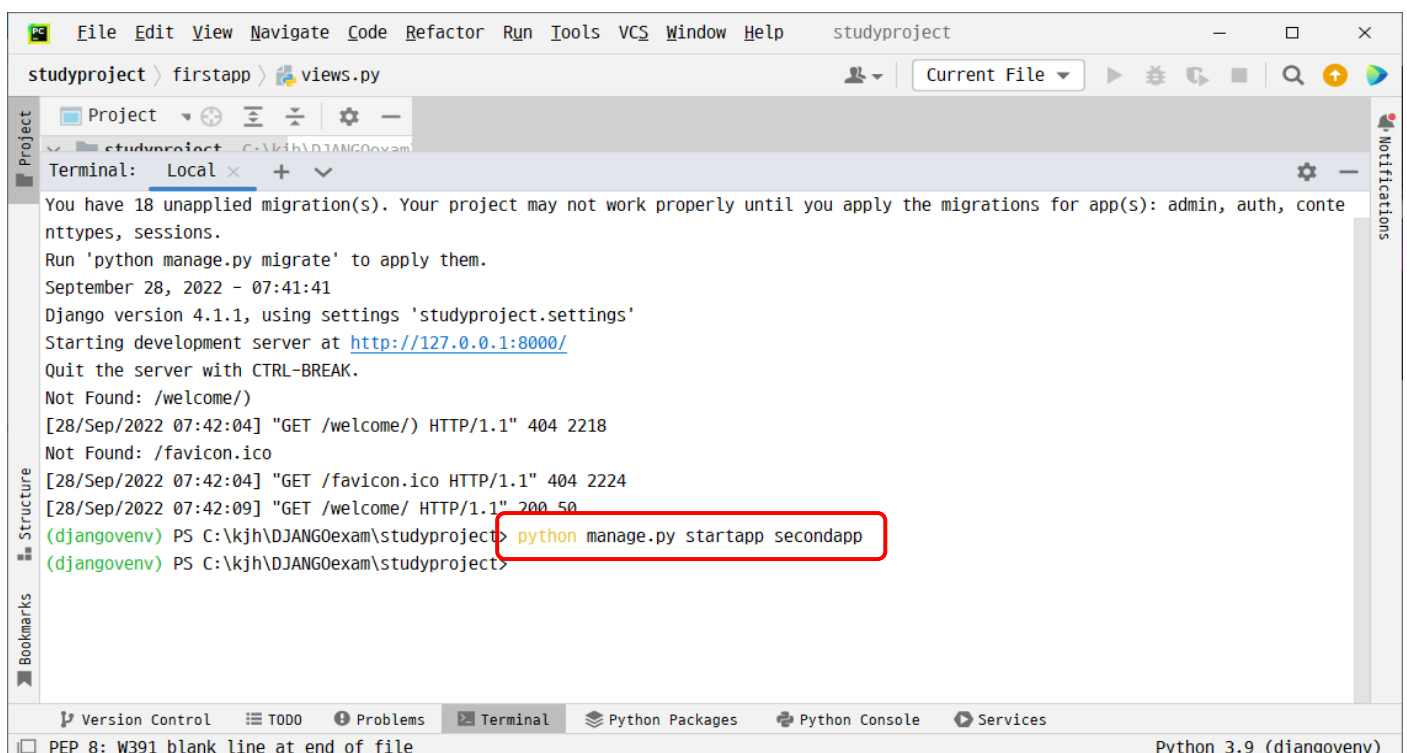
[장고로 구현하는 웹 사이트 구성]



[두 번째 장고 앱 생성]

(1) 파이썬 하단 왼쪽에 있는 Terminal을 선택하여 cmd 창과 비슷한 창을 출력한다.(터미널 창이라고 한다.)

(2) 학습용 소스들을 작성할 secondapp 이라는 장고 앱을 하나 생성한다.



The image shows a Django project structure in an IDE. The 'Project' pane on the left lists the following files and folders:

- firstapp
- secondapp
- migrations
- templates
 - exam1.html
- __init__.py
- admin.py
- apps.py
- models.py
- tests.py
- urls.py
- views.py
- studyproject
 - __init__.py
 - asgi.py
 - settings.py
 - urls.py
 - wsgi.py
 - db.sqlite3

Callouts provide the following code snippets:

- exam1.html:**

```
<body>
<h1>안녕? 템플릿(html)을 통해서 응답해요</h1>
</body>
```
- urls.py (firstapp):**

```
from django.urls import path
from . import views
urlpatterns = [
    path("", views.exam1, name='exam1'),
]
```
- views.py (firstapp):**

```
from django.shortcuts import render
from django.http import HttpResponse
from django.template import loader

def exam1(request):
    template = loader.get_template('exam1.html')
    return HttpResponse(template.render(None, request))
```
- urls.py (studyproject):**

```
from django.contrib import admin
from django.urls import path, include
import firstapp.views

urlpatterns = [
    path('admin/', admin.site.urls),
    path('welcome/', firstapp.views.welcome),
    path('secondapp/', include('secondapp.urls')),
]
```
- settings.py (studyproject):**

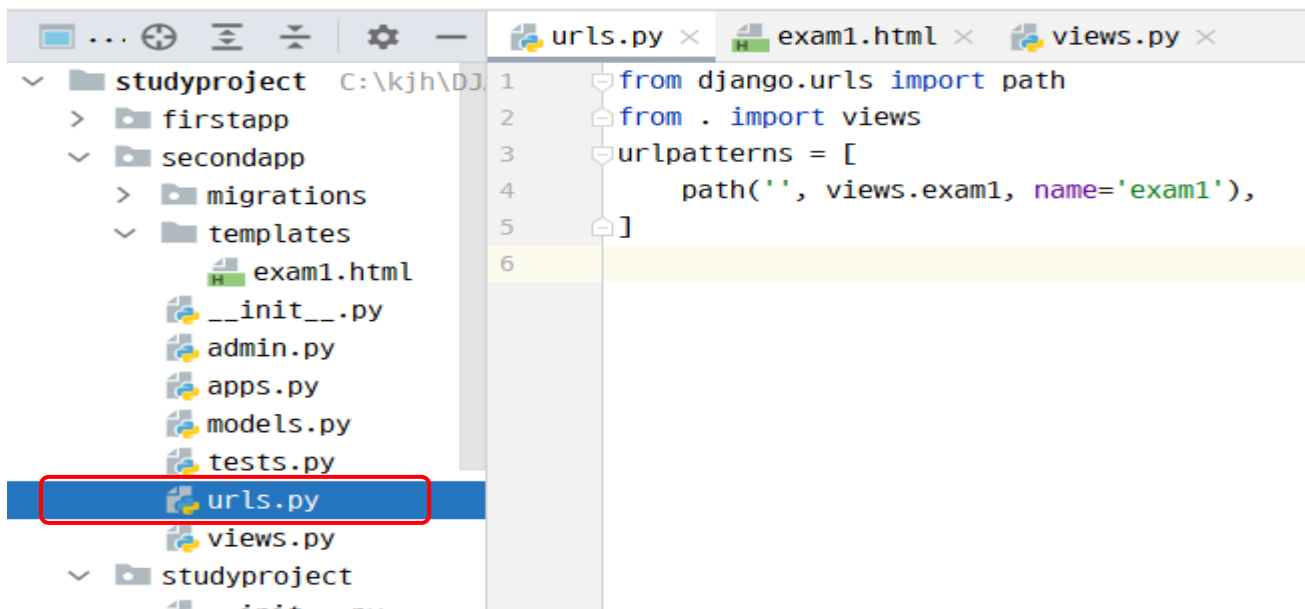
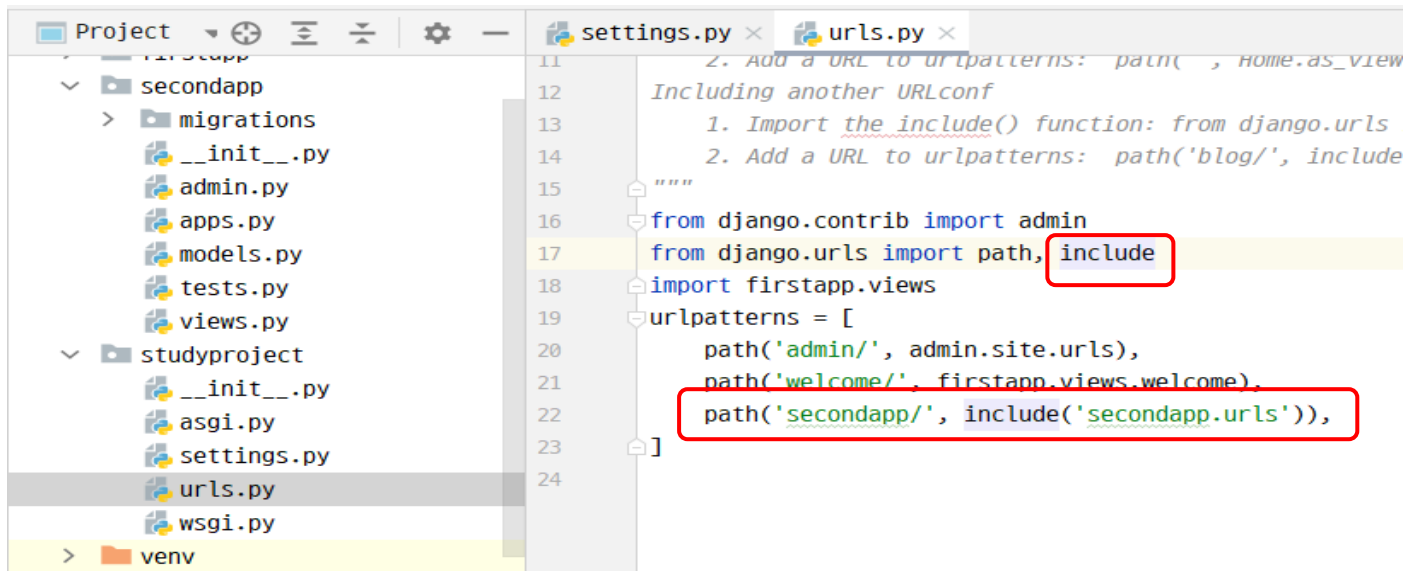
```
INSTALLED_APPS = [
    'secondapp',
    :
]
```

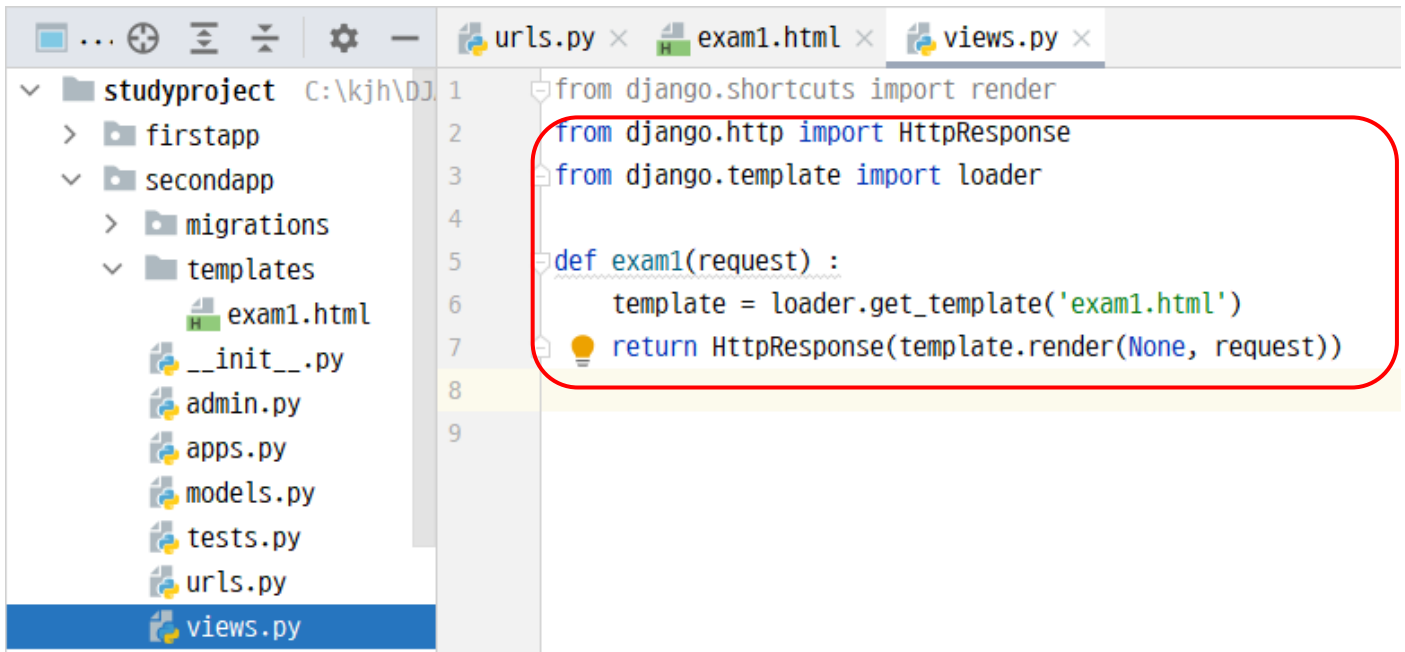
The image shows the 'settings.py' file in an IDE. The 'Project' pane on the left lists the following files and folders:

- migrations
- __init__.py
- admin.py
- apps.py
- models.py
- tests.py
- views.py
- studyproject
 - __init__.py
 - asgi.py
 - settings.py
 - urls.py
 - wsgi.py
- venv
- db.sqlite3

The 'settings.py' file content is as follows:

```
27 ALLOWED_HOSTS = []
28
29
30
31 # Application definition
32
33 INSTALLED_APPS = [
34     'secondapp',
35     'django.contrib.admin',
36     'django.contrib.auth',
37     'django.contrib.contenttypes',
38     'django.contrib.sessions',
39     'django.contrib.messages',
40     'django.contrib.staticfiles',
41 ]
42
```





장고 웹 서버를 기동한다.

```
(djangoenv) PS C:\kjh\DJANGOexam\studyproject> python manage.py runserver
```

Watching for file changes with StatReloader

Performing system checks...

System check identified no issues (0 silenced).

You have 18 unapplied migration(s). Your project may not work properly until you apply the migrations for nttypes, sessions.

Run 'python manage.py migrate' to apply them.

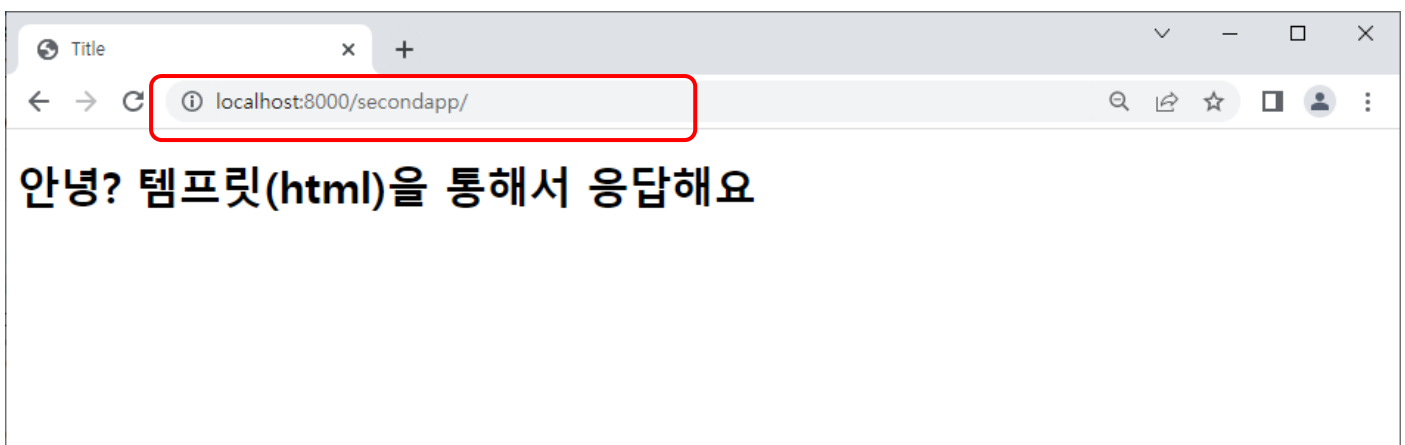
September 28, 2022 - 08:24:03

Django version 4.1.1, using settings 'studyproject.settings'

Starting development server at <http://127.0.0.1:8000/>

Quit the server with CTRL-BREAK.

브라우저에서 요청해 본다. (<http://localhost:8000/secondapp/>)



[Terminal 을 PowerShell 이 아니라 cmd 창으로 기동시키게 하여 해결하는 방법]

