

Django Basic

5번째 세션

NEXT X LIKELION 김이린

| 지금까지...

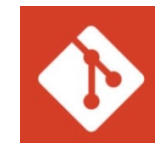
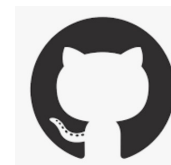
Programming Language



Framework



Tools



장고? 프레임워크?



(보안이 우수하고 유지보수가 편리한) 웹사이트를 신속하게 개발하도록 도움을 주는 파이썬 기반 웹 프레임워크



웹 애플리케이션을 구축할 때, 공통적인 개발 환경을 제공해주는 구조를 짜놓고 그 위에 덧붙여 만들도록 하는 것

cf) 라이브러리?

모듈과 패키지들 등 도구의 묶음 (프레임워크에서 라이브러리를 가져다 쓴다!)

장고의 장점

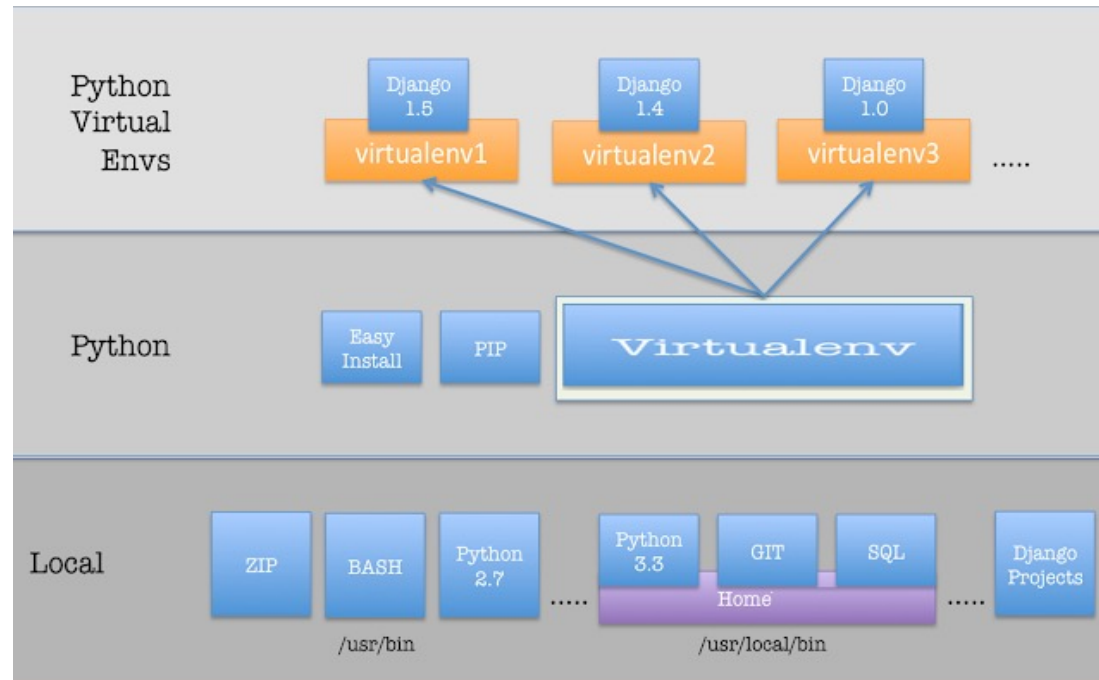
The Django logo, featuring the word "django" in a white, lowercase, sans-serif font on a dark green rectangular background.

The web framework for
perfectionists with deadlines.

1. 빨리 만들 수 있다
2. 안전하다
 - xss, csrf 등 해킹에 대한 보안 취약점 보완 방법 기본 제공
3. 이미 구현되어 있는 기능이 많다
 - 로그인, 관리자 기능, 데이터베이스 등

가상환경 복습

프로젝트 별로 독립된 개발환경 구축



- 서로 다른 가상환경에 설치된 모듈의 영향을 받지 않는 **독립적** 공간
- 한 가상환경 내에서 **통일된 라이브러리 버전**을 사용하므로 협업 시 버전 충돌 방지

| 중간점검

Python 버전 통일 확인



```
$ python3 --version
```



```
$ python --version
```

가상환경 켜지 않은 곳에서 Python 3.9.8
- Anaconda의 경우 버전 충돌이 일어나니 삭제..!

| pipenv

pipenv 명령어 정리

`$ pipenv shell` - 가상환경 생성 및 시작

`$ exit` - 가상환경 종료

`$ pipenv install 패키지명` - 해당 패키지 설치

`$ pipenv uninstall 패키지명` - 해당 패키지 제거

| 장고 설치

Django를 설치해봅시다!

1. 작업할 폴더 생성

```
$ mkdir session5
```

2. 해당 폴더로 이동

```
$ cd session5
```

3. Pipenv 로 가상환경 생성 (pipenv 켜져있는지 확인!)

```
$ pipenv shell
```

4. Pipenv를 통해 가상환경 내에 Django 패키지 설치

```
$ pipenv install django
```

설치 후, `$ django-admin --version` 으로 설치 확인! (pipenv graph 해도 됩니당...)

| 장고 초기 세팅

Django 세팅 to-do list

`manage.py`

1. 장고 프로젝트 생성
2. 서버 실행
3. 프로젝트 내 장고 앱 생성
4. 데이터 베이스 생성
5. 관리자 계정 생성

장고 프로젝트 생성

장고 초기 세팅

장고 프로젝트 생성

장고의 프로젝트는 장고의 각종 세팅을 포함하는 기본 프로젝트 구조를 의미!

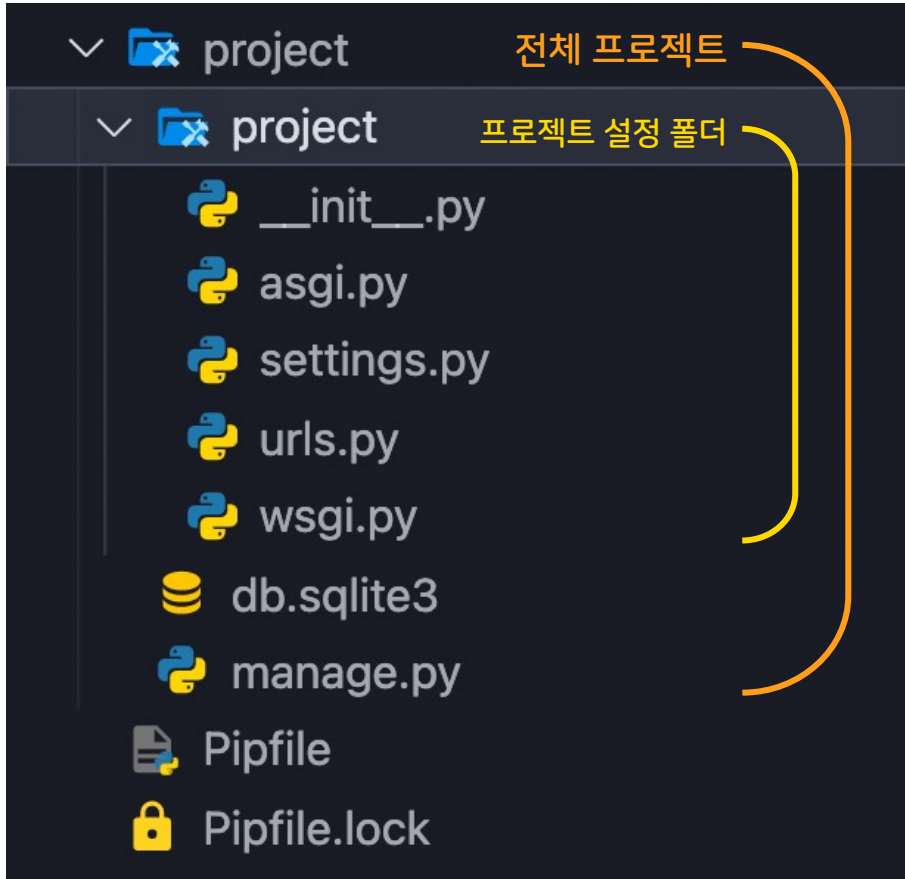
```
$ django-admin startproject 프로젝트명
```

ex) `project`

장고 프로젝트 생성

장고 초기 세팅

프로젝트 구조



manage.py

- django project관리를 도와주는 command-line utility 제공
 - 다른 작업 없이 컴퓨터에서 웹 서버 실행
 - django app 생성
 - 관리자 계정 생성
 - DB 초기화 및 변경사항 반영

settings.py

- django project의 각종 설정을 포함하고 있는 파일

urls.py

- url 패턴 목록을 포함하고 있는 파일

프로젝트 실행방법

manage.py가 있는 위치에서 (/session5/project/)

\$ python manage.py runserver

=> 장고 서버 실행

장고 서버 시작

Starting development server at `http://127.0.0.1:8000/`
Quit the server with `CONTROL-C`.

=> 개발 서버가 <http://127.0.0.1:8000/> 에서 시작되었다.

<http://127.0.0.1:8000/>

<http://localhost:8000/>

localhost == 127.0.0.1 == 여러분의 PC를 가리키는 주소

다른 사람은 접속할 수 없고, 로컬에서만 접속 가능!

💡 :8000 은 포트번호를 의미



The install worked successfully! Congratulations!

You are seeing this page because `DEBUG=True` is in your settings file and you have not configured any URLs.

프로젝트 내 장고 앱 생성

장고 초기 세팅

장고 앱 생성

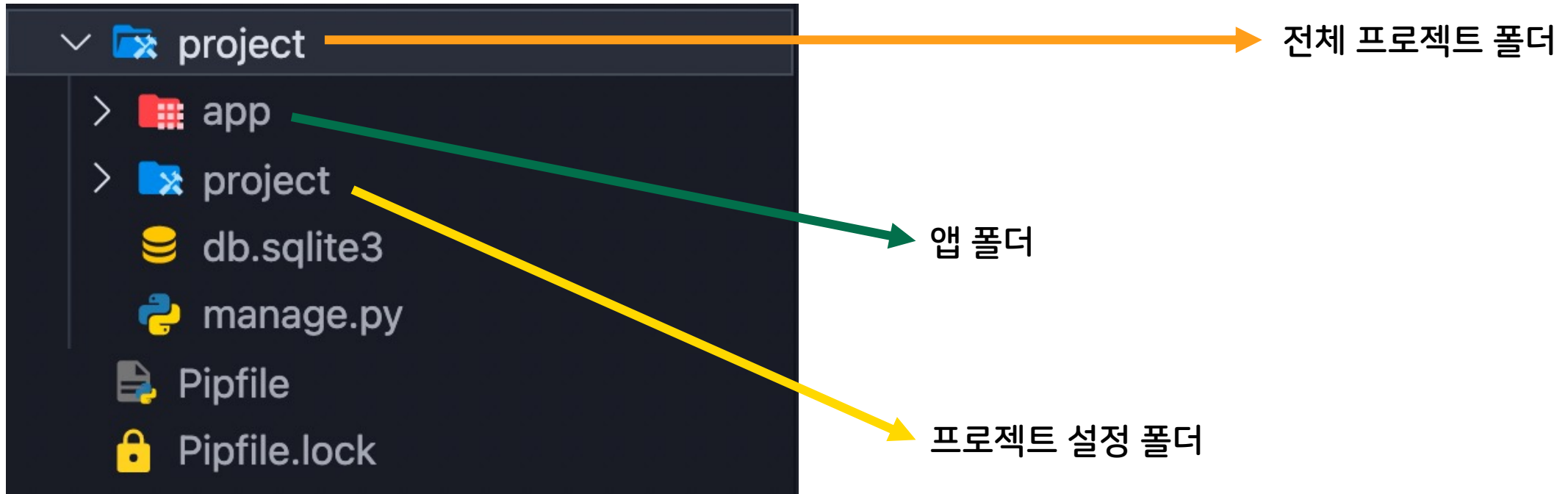
```
$ python manage.py startapp 앱 이름
```

ex) app

💡 하나의 프로젝트는 여러 앱으로 구성될 수 있음
장고에서의 앱은 프로젝트 내에서 특정 기능을 담당하는 web application을 의미!

e.g. 쇼핑몰 안에 게시판, 결제, 장바구니의 각기 다른 기능은 각각의 코드를 묶어서 관리하는 게 좋겠죠?

프로젝트 구조



장고 앱 생성

장고 초기 세팅

앱 생성 후에는 settings.py 에 추가!

```
INSTALLED_APPS = [  
    'django.contrib.admin',  
    'django.contrib.auth',  
    'django.contrib.contenttypes',  
    'django.contrib.sessions',  
    'django.contrib.messages',  
    'django.contrib.staticfiles',  
    'app',  
]
```

→ settings.py의 INSTALLED_APPS 리스트에 새로 생성한 앱의 이름(ex. app)을 추가

settings.py 자세히 보기

DEBUG

- 개발모드에서는 True
- 배포시에는 False

INSTALLED_APPS

- 해당 프로젝트에 설치된 앱들 명시
- 하나의 프로젝트에서 여러 개의 앱 생성 가능

TEMPLATES

- 템플릿 엔진, 경로, 옵션 등 설정

TIME_ZONE

- 시간 설정

DATABASES

- 데이터베이스 설정
- sqlite 가 기본값

추가 setting은 docs 참조: <https://docs.djangoproject.com/en/3.1/ref/settings/>

TIME_ZONE 한국 시간에 맞추기

settings.py의 TIME_ZONE을 변경!

```
TIME_ZONE = 'Asia/Seoul'
```

데이터 베이스 생성

장고 초기 세팅

데이터 베이스 생성을 위해

```
$ python manage.py makemigrations
```

```
$ python manage.py migrate
```

데이터베이스 관련 변경사항이 있을 때마다 적용을 위해 해당 작업을 해야 합니다!

DB에 대해서는 다음 세션에서 자세하…!

관리자 계정 생성

장고 초기 세팅

관리자 계정 생성을 위해

```
$ python manage.py createsuperuser
```

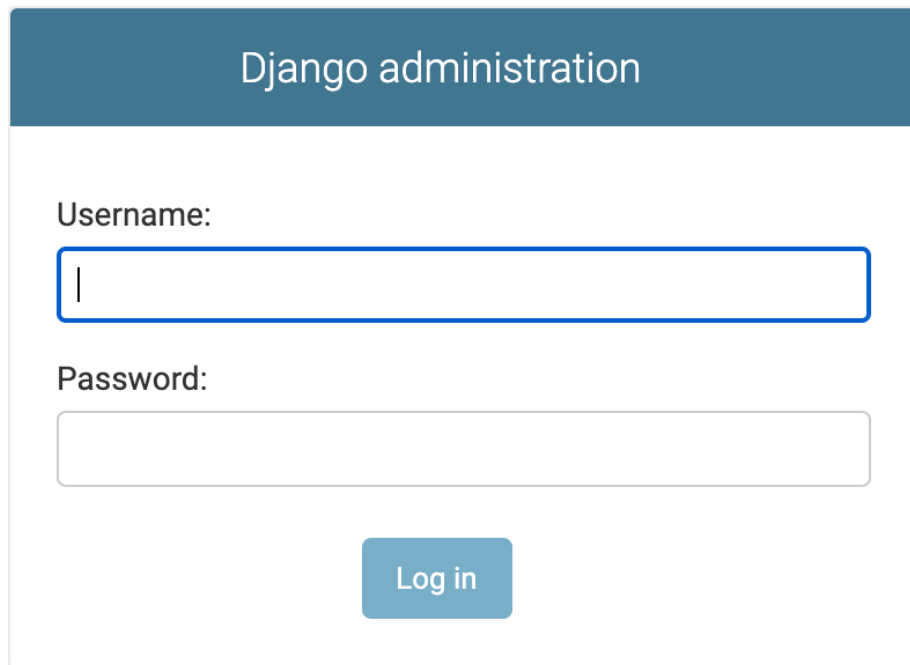
장고는 기본적으로 관리자 기능을 제공한다!

관리자 계정 생성 후 Django admin 기능 사용 가능

관리자 계정 생성

장고 초기 세팅

관리자 페이지 사용



Django administration

Username:

Password:

Log in

<http://127.0.0.1:8000/admin>

슈퍼 유저로 로그인 후에 사용가능

runserver 잊지 않았죠..?

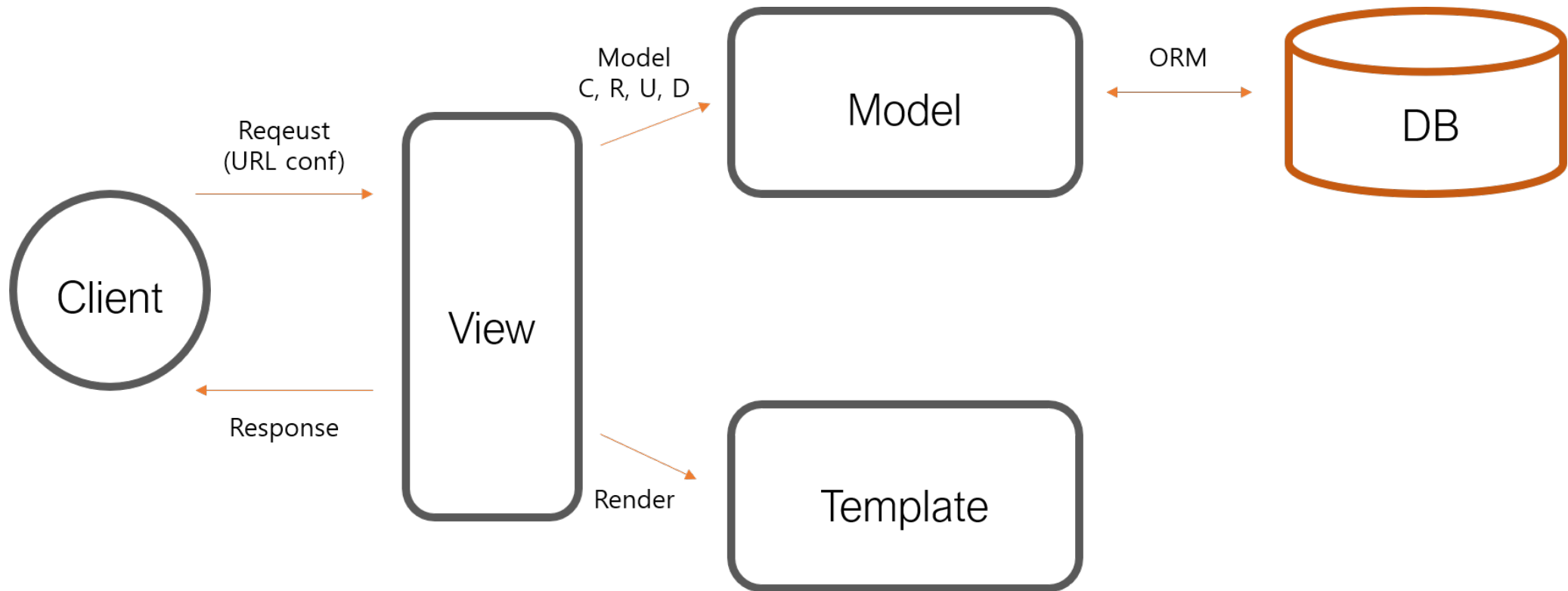
| 장고의 동작 과정

MTV

- 장고의 설계 패턴
- 일반적으로 웹 개발 시에 언급되는 MVC 패턴 기반

오늘 꼭 기억해 주세요!!!

장고의 동작 과정



| MTV 패턴

Model - 데이터베이스 설계

- 데이터베이스에 저장되는 데이터의 영역

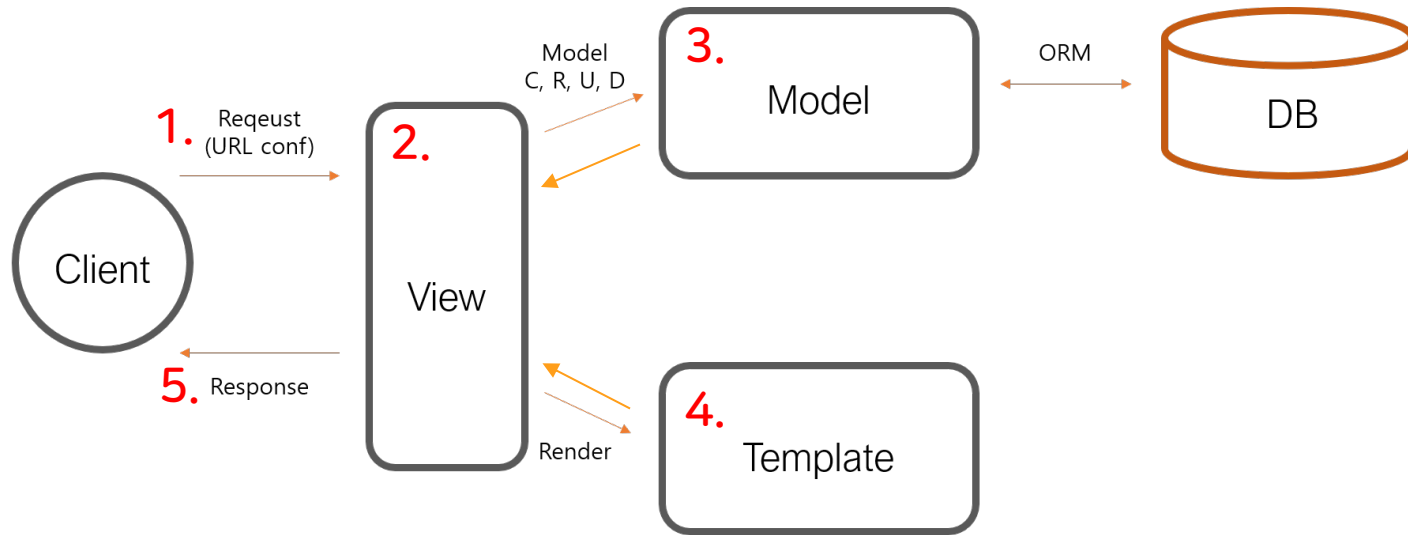
Template - 화면 UI 설계

- 사용자에게 보여지는 영역, 화면
- HTML

View - 프로그램 로직 설계

- 요청에 따라 Model에서 필요한 데이터를 가져와 처리
- 처리 결과를 Template에 전달

MTV 패턴



1. urls.py에서 요청 들어온 URL을 분석 (URL conf)
2. 해당 URL에 매칭되는 View 실행
3. View는 요청에 따라 Model을 통해 DB처리
 - 생성(Create), 조회(Read), 수정(Update), 삭제>Delete)
4. View는 Template를 사용하여 클라이언트에게 응답할 HTML 파일 렌더링
5. View는 최종적으로 HTML파일을 클라이언트에게 Response

| MTV 패턴

MTV의 장점

- 데이터(Model), 사용자 인터페이스(Template), 데이터를 처리하는 로직(View)을 구분해서 한 요소가 다른 요소들에 영향을 주지 않도록 설계하는 방식
- 각각 독립적인 영역에서 개발이 가능

| MTV 패턴 연습

예시를 보면서 같이 만들어 봅시다!

'/hello' 로 요청 보내면 환영인사를 보여주기!

MTV 패턴 연습

1. URL conf

URL 지정해주기 => urls.py

```
project > project >  urls.py > ...
```

```
16 from django.contrib import admin
17 from django.urls import path
18 from app import views
19
20 urlpatterns = [
21     path('admin/', admin.site.urls),
22     path('hello', views.hello, name='hello'),
23 ]
```

URL과 view 연결을 위해
app 디렉토리 안의 views를 import

views에서 작성해줄 함수명

Html에서 URL에 접근하기 위해 사용할 이름! {% url '<name>' %}

MTV 패턴 연습

2. View

View 작성하기 => views.py

```
project > app > views.py > hello
1  from django.shortcuts import render
2
3  # Create your views here.
4
5
6  def hello(request):
7      # 로직 작성 부분
8      return render(request, 'hello.html')
```

url 'hello'에 연결되는 함수 hello 작성

'템플릿 요소를 응답에 포함시켜 화면에 띄워줄게!'

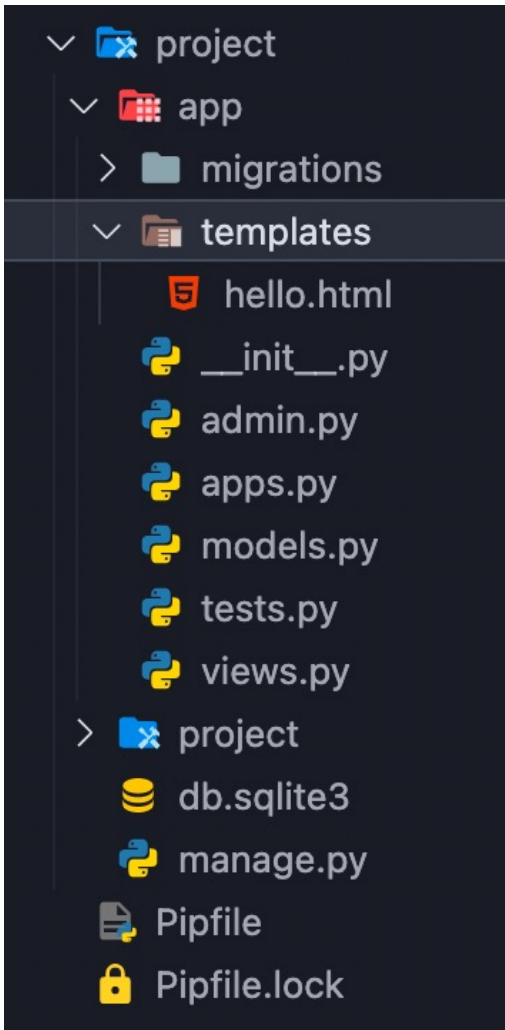
- 첫번째 인수: request
- 두번째 인수: 템플릿 이름
- 세번째 인수(선택적): context,
view에서 사용하던 파이썬 변수를 html에 넘길 수 있음

Views에서 처리한 정보를 'hello.html'로 보낸다!
데이터 처리할 게 없어서 3. Model 과정은 없음

MTV 패턴 연습

4. Template

Template 만들기



1. app 디렉토리 밑에 templates 폴더 생성

2. templates 폴더 안에 hello.html 생성

장고는 기본적으로 앱(app) 디렉토리 하위에 있는 templates 디렉토리로 템플릿 디렉토리로 인식
settings.py의 TEMPLATES 옵션에서 변경 및 추가 가능

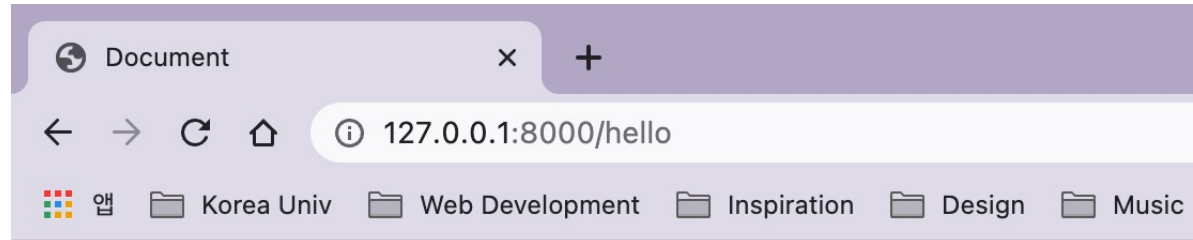
MTV 패턴 연습

4. Template

hello.html 작성

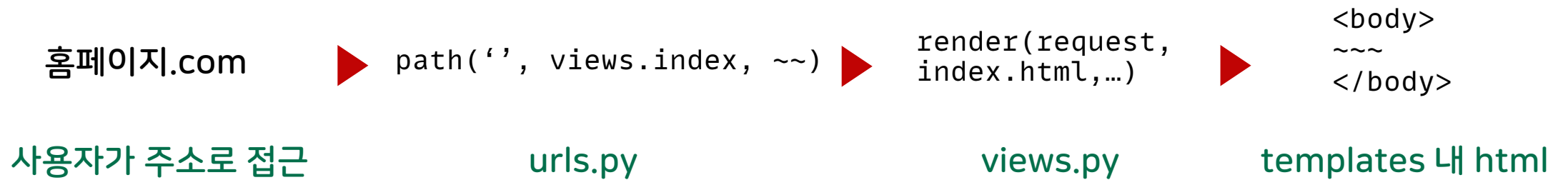
```
project > app > templates >  hello.html > ...  
1  <!DOCTYPE html>  
2  <html lang="en">  
3    <head>  
4      <meta charset="UTF-8" />  
5      <meta http-equiv="X-UA-Compatible" content="IE=edge" />  
6      <meta name="viewport" content="width=device-width, initial-scale=1.0" />  
7      <title>Document</title>  
8    </head>  
9    <body>  
10     <h1>NEXT 멋사 10기 여러분 안녕하세요~</h1>  
11   </body>  
12 </html>
```

<http://127.0.0.1/hello>



NEXT 멋사 10기 여러분 안녕하세요~

| MTV 패턴 기억



| 정리

```
$ pipenv shell
```

```
$ pipenv install django
```

```
$ django-admin startproject 프로젝트 이름
```

```
$ cd 프로젝트 폴더명
```

```
$ python manage.py startapp 앱 이름
```

```
$ python manage.py makemigrations
```

```
$ python manage.py migrate
```

```
$ python manage.py createsuperuser
```

```
$ python manage.py runserver
```

요청과 응답

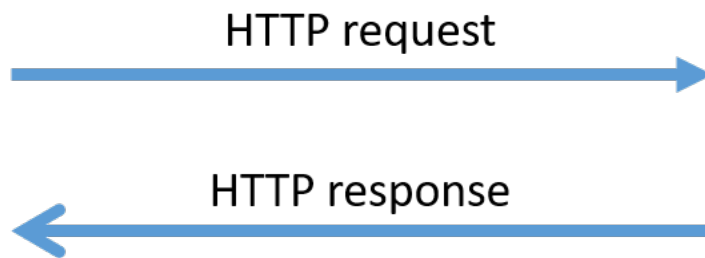
Request? Response?

💡 Http 프로토콜: 인터넷상에서 데이터를 주고 받기 위한 서버/클라이언트 모델을 따르는 통신 규약



Client

요청을 보내는 쪽, 웹의 브라우저



Server

요청을 받고 응답을 보내주는 쪽,
서비스를 제공하는 역할

| 요청과 응답



웹 클라이언트(브라우저)

```
{ "dinner" : "sushi" }
```

요청(Request)



응답(Response)



```
{ 🍣 }
```



웹 서버

사용자: 저녁 초밥 먹고 싶어요!

| Http method - GET/POST

GET

존재하는 자원을 조회할 때

읽기

어떤 내용을 가지고 오고 싶을 때!

URL에 변수(데이터)를 포함시켜 요청

POST

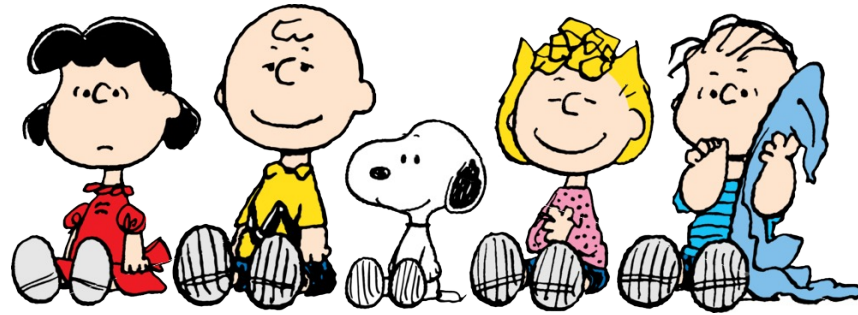
새로운 자원을 생성

저장

새로운 내용을 쓸 때

URL에 변수(데이터)를 노출하지 않고 요청

오늘의 목표!



글자수 세기 페이지 만들기

- 텍스트를 입력하고 제출 시 몇 글자인지 알려주는 페이지

실습

글자수 세기 페이지 만들기

pipfile이 위치한 경로에서 project 만들기 시작

```
$ django-admin startproject CountProject
```

```
$ cd CountProject
```

```
$ python manage.py startapp CountApp
```

```
$ python manage.py makemigrations
```

```
$ python manage.py migrate
```

```
$ python manage.py createsuperuser
```

```
$ python manage.py runserver
```

실습

글자수 세기 페이지 만들기

settings.py

```
INSTALLED_APPS = [  
    ...,  
    'CountApp',  
]
```

```
TIME_ZONE = 'Asia/Seoul'
```

실습

글자수 세기 페이지 만들기

필요한 페이지는?

글자를 입력하는 페이지 => count

글자수를 세서 결과를 보여주는 페이지 => result

실습

글자수 세기 페이지- count

1. URL conf

urls.py

```
from CountApp import views

urlpatterns = [
    path('admin/', admin.site.urls),
    path('count', views.count, name='count'),
]
```

실습

글자수 세기 페이지- count

2. View

CountProject/CountApp/
views.py

```
from django.shortcuts import render

# Create your views here.
def count(request):
    return render(request, 'count.html')
```

💡 따로 처리할 로직은 없으므로 html로 렌더링만 해준다!

실습

글자수 세기 페이지 만들기

4. Template

count.html 생성

CountApp/templates/count.html

SESSION5PRACTICE

CountProject

CountApp

migrations

templates

count.html

__init__.py

admin.py

apps.py

models.py

tests.py

views.py

CountProject

db.sqlite3

manage.py

Pipfile

Pipfile.lock

실습

글자수 세기 페이지- count

4. Template

count.html

```
<body>
  <h1>글자를 입력해 보세요!</h1>
  <form action="{% url 'result' %}" method="post">{% csrf_token %}
    <textarea name="text" cols="30" rows="10"></textarea>
    <br />
    <input type="submit" value="제출" />
  </form>
</body>
```

💡 form 태그를 사용해서 formData를 'result' url로 보낸다!

{% %}

Template 내부에서 사용하는 django 문법

- 나중에 Template 엔진이 해당 값으로 변경해줌

💡 검색어: django template language

csrf_token

Csrf 해킹 공격 방지

* http method 가 POST일 때 꼭 적어준다

실습

글자수 세기 페이지- count

count.html

💡 csrf 공격 방지 - POST일 때 써준다.

```
<body>
  <h1>글자를 입력해 보세요!</h1>
  <form action="{% url 'result' %}" method="post">{% csrf_token %}
    <textarea name="text" cols="30" rows="10"></textarea>
    <br />
    <input type="submit" value="제출" />
  </form>
</body>
```

💡 urls.py의 urlpattern중 name='result'인 url로 formData전송한다!

💡 http method 중 POST 사용한다

실습

글자수 세기 페이지 만들기

한 페이지만 더 만들면!

~~글자를 입력하는 페이지 => count~~

글자수를 세서 결과를 보여주는 페이지 => result

실습

글자수 세기 페이지- result

urls.py

```
from CountApp import views
```

```
urlpatterns = [  
    path('admin/', admin.site.urls),  
    path('count', views.count, name='count'),  
    path('result', views.result, name='result'),  
]
```

💡 template에서 사용할 name은 result로 한다!

💡 'result' url과 CountApp/views의 함수 result를 연결!

실습

글자수 세기 페이지- result

views.py

💡 count -> result 로 넘어온 request

💡 html 태그 name과 똑같이

```
def result(request):  
    text = request.POST['text']  
    total_len = len(text)  
    no_blank_len = len(text.replace(' ', ''))  
    return render(request, 'result.html', {'total_len': total_len, })
```

💡 html로 렌더링 할 데이터를 딕셔너리 형태로 넘겨준다!

실습

글자수 세기 페이지- result

result.html

```
return render(request, 'result.html', {'total_len': total_len,})
```

```
<body>  
  <p>글자수: {{total_len}}</p>  
</body>
```

key값

{{ }}

💡 views에서 넘겨준 데이터를 html에서 사용할 때

실습

직접 해보기

직접 해봅시다!

입력한 텍스트, 공백 제외 글자수도 추가로 보여주기

💡 hint - 수정할 파일
views.py
result.html

views.py

```
def result(request):  
    text = request.POST['text']  
    total_len = len(text)  
    no_blank_len = len(text.replace(' ', ''))  
  
    return render(request, 'result.html', {  
        'text': text,  
        'total_len': total_len,  
        'no_blank_len': no_blank_len})
```

실습

직접해보기- 정답

result.html

```
<body>
  <p>입력한 텍스트: {{text}}</p>
  <p>글자수: {{total_len}}</p>
  <p>공백제외 글자수: {{no_blank_len}}</p>
</body>
```


CSS 적용하기

Static files => 이미지, 자바스크립트, CSS 같은 파일들

Django에서 이러한 정적 파일을 관리하는 기능 제공!

CSS 적용하기

settings.py static 관련 세팅

```
STATIC_URL = '/static/'
```

=> Templates와 마찬가지로 앱 디렉토리 하위에 static 디렉토리 인식

참고

STATICFILES_DIRS

- 프로젝트 전반에 사용하는 static 파일들의 디렉토리 목록

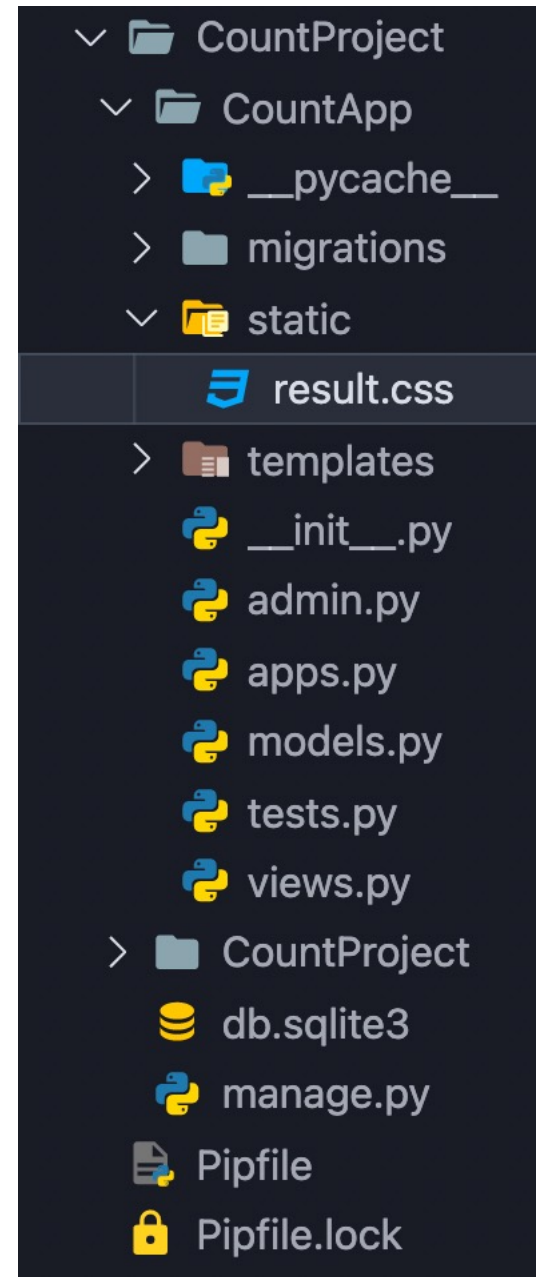
STATIC_ROOT

- 배포시 흩어져 있는 static file 들 모이는 위치
* python manage.py collectstatic

CSS 적용하기

CountApp/static 폴더 생성

result.css 생성



CSS 적용하기

result.css

```
body {  
    color: salmon;  
}
```

CSS 적용하기

html에 css 연결하기

result.html

💡 {% load static %} 꼭 써주기!

```
<head>
  {% load static %}
  <link rel="stylesheet" type="text/css" href="{% static 'result
.css' %}">
  ...
</head>
```

💡 {% static '<css 파일 이름>' %}

.gitignore

협업을 위한 .gitignore 파일

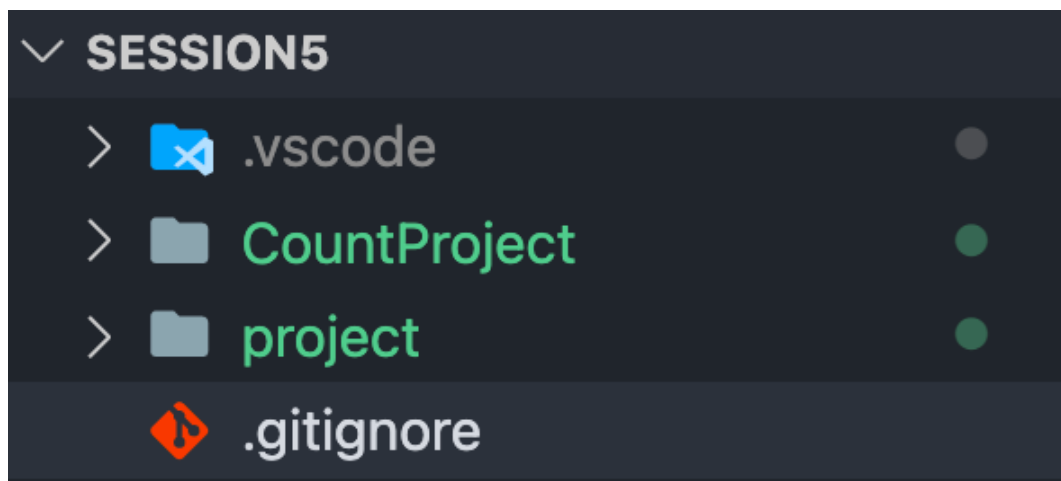
.gitignore 파일에 적힌 파일, 폴더는 git 버전관리에서 제외
=> github에도 올라가지 않음!

- 개인정보, 비밀번호 등의 공개되면 안되는 정보가 있는 파일,폴더
 - 각자 개발환경 관련 파일, 폴더
- ex) .vscode, migrations, db.sqlite3, __pycache__, DS_Store

.gitignore

최상단 디렉토리에 .gitignore파일 생성

(해당 위치에서 \$ git init 실시)



- 💡 이미 git으로 관리하고 있는 디렉토리 내에서 작업했다면, git init할 필요가 없다.
- 💡 위의 경우 git init한 위치가 아닌 프로젝트가 위치한 폴더의 최상단에 .gitignore생성

.gitignore

.gitignore

.vscode

migrations

db.sqlite3

__pycache__

.gitignore

.gitignore 작성시 순서 주의

.gitignore 파일 생성 및 작성

해당 파일들이 git에서 무시됨

```
$ git add .
```

\$ git add 를 하기 전에 .gitignore를 생성/작성한다.

과제

오늘의 과제는~



과제



텍스트를 입력해주세요.

1. 글자수 세기 프로젝트에 단어 개수 세는 기능 추가!
- Django의 동작 과정 생각하면서 작업해보기

2. count.html, result.html 꾸미기

공백포함

0 0 byte

공백제외

0 0 byte

단축키 안내 ▾