

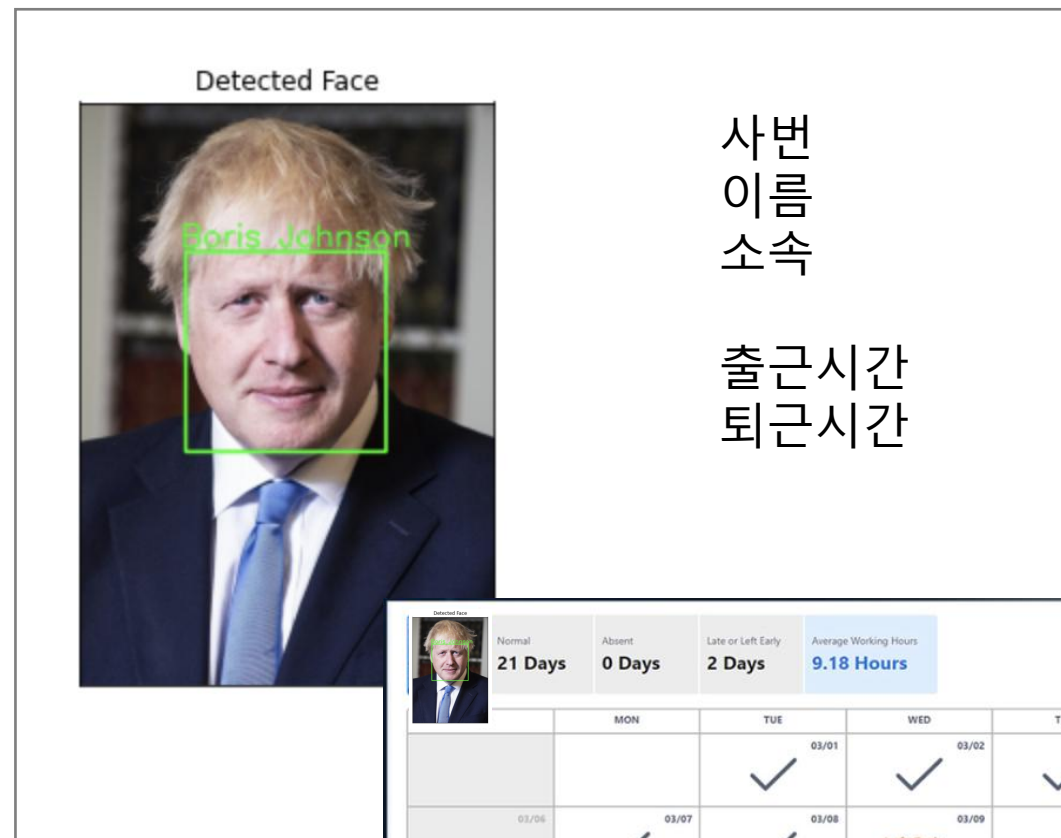
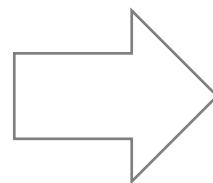
2025년 서울 민간기업 맞춤형 매력일자리 사업

# 인공지능, Dlib, Django, OpenCV

- 2025.05.12~15 -

# 최종결과물 목표





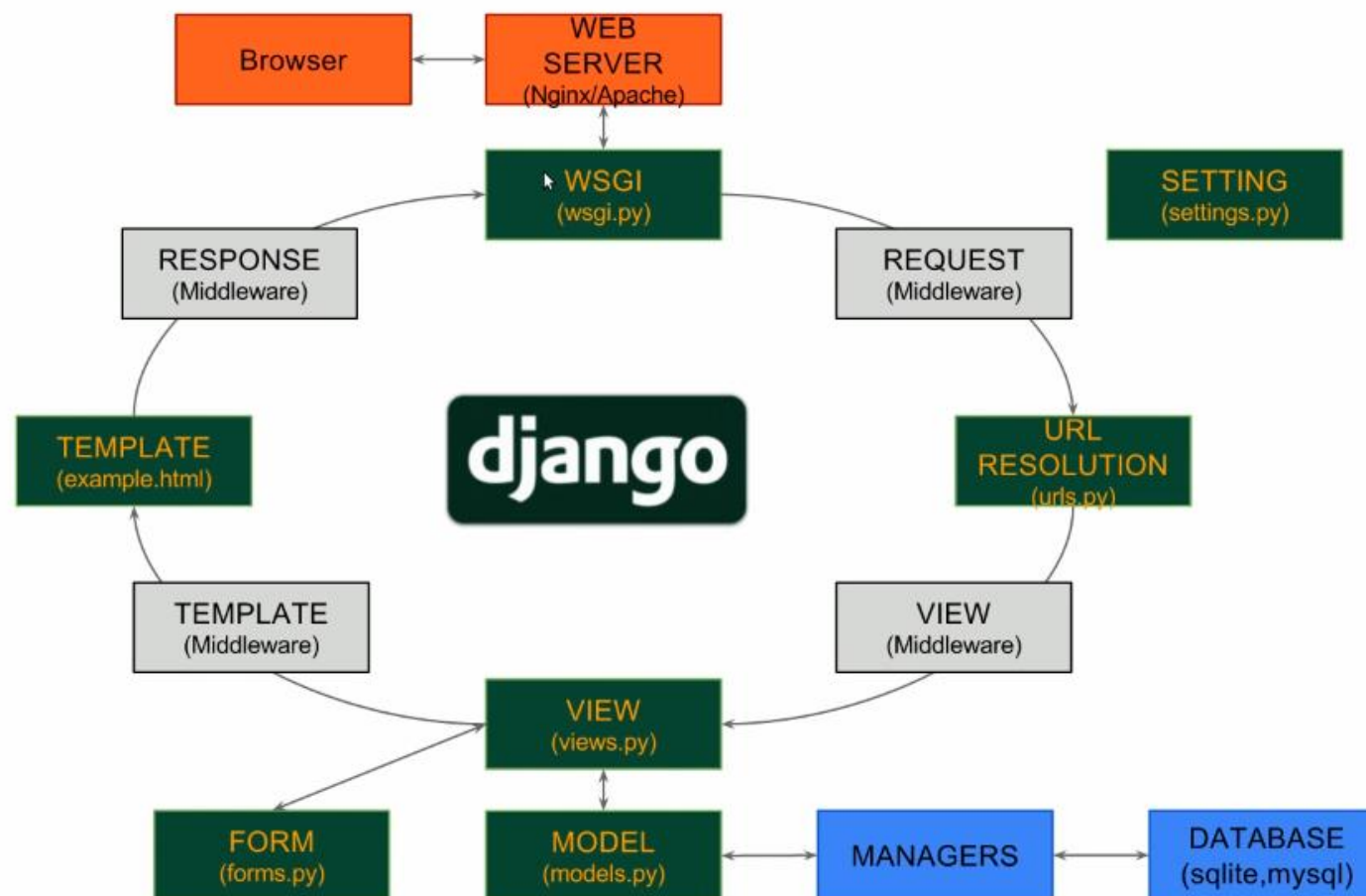
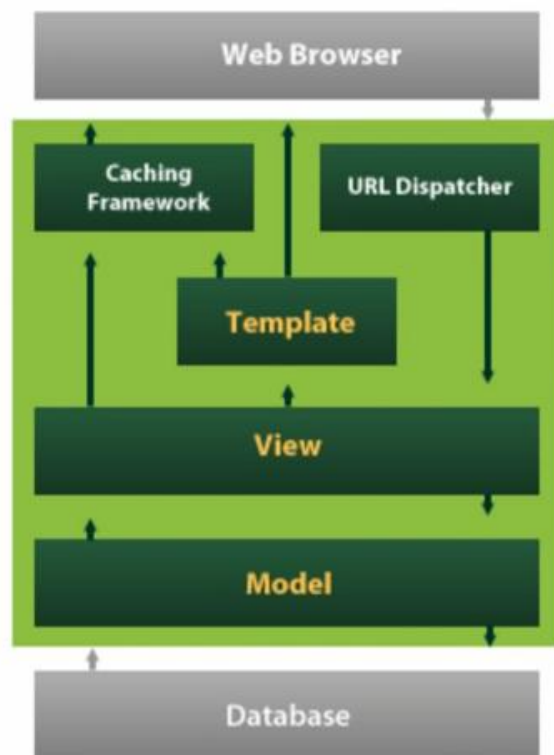
| Detected Face | Normal  | Absent     | Late or Left Early | Average Working Hours |
|---------------|---------|------------|--------------------|-----------------------|
|               | 21 Days | 0 Days     | 2 Days             | 9.18 Hours            |
|               | MON     | TUE        | WED                | THU                   |
|               |         | 03/01      | 03/02              | 03/03                 |
|               |         | ✓          | ✓                  | ✓                     |
| 03/06         | 03/07   | 03/08      | 03/09              | 03/10                 |
|               | ✓       | ✓          | Left Early         | ✓                     |
| 03/13         | 03/14   | 03/15      | 03/16              | 03/17                 |
|               | ✓       | Left Early | ✓                  | ✓                     |
| 03/20         | 03/21   | 03/22      | 03/23              | 03/24                 |
|               | ✓       | ✓          | ✓                  | ✓                     |

사용 인공지능 프레임워크  
Dlib, Face Recognition  
OpenCV, Django, Mariadb  
Web(html, css, JavaScript, tailwindcss)

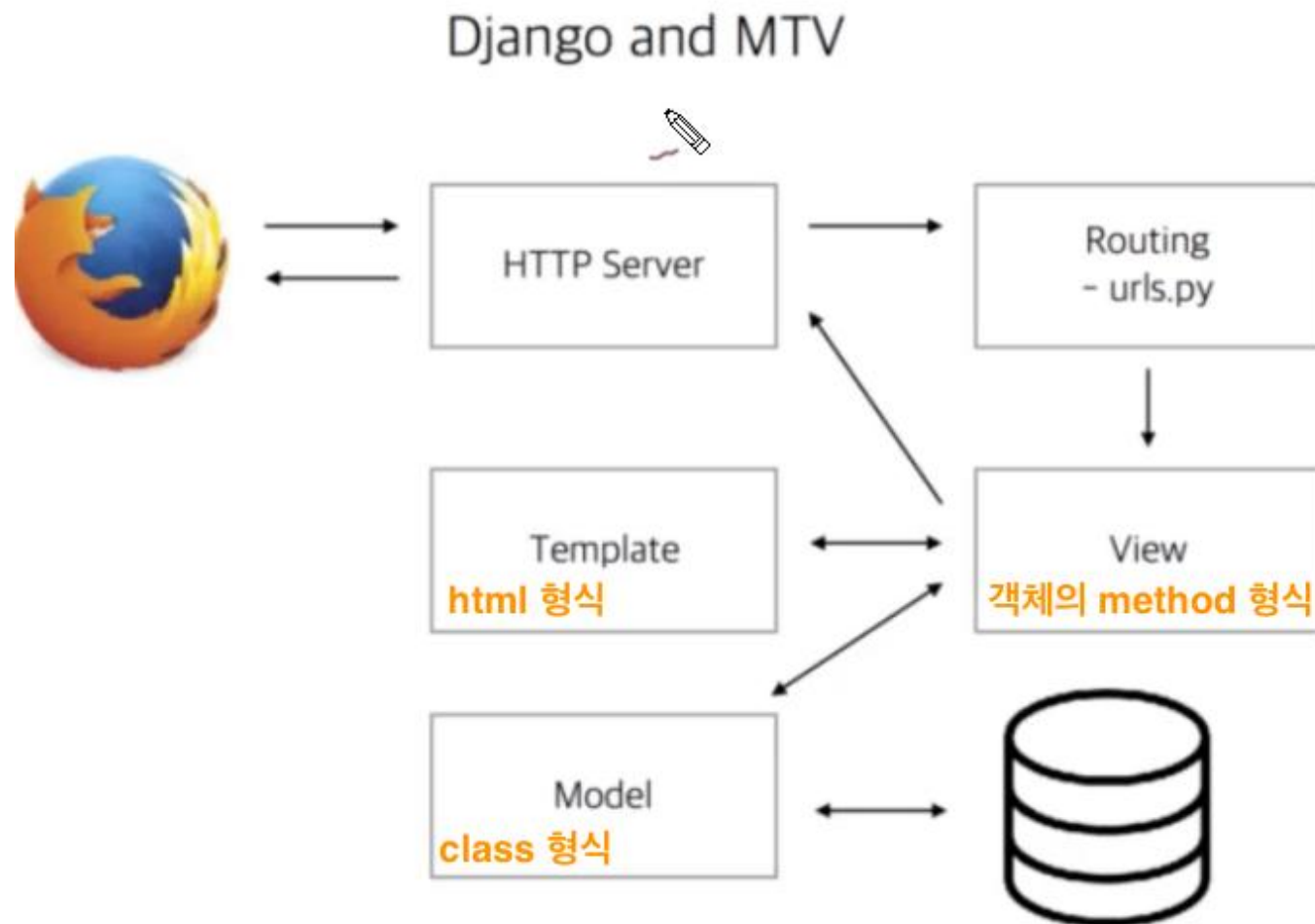
# Django 기본개념

# Django 개념

## Django 개념

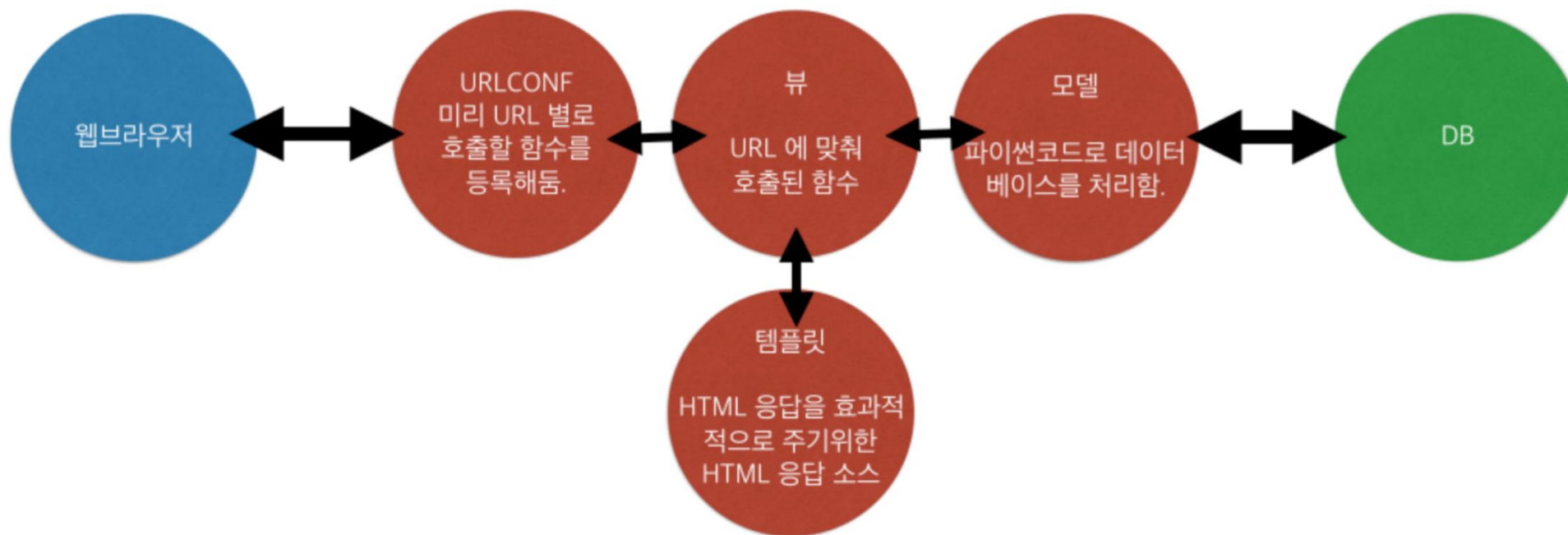


# Django MTV모델

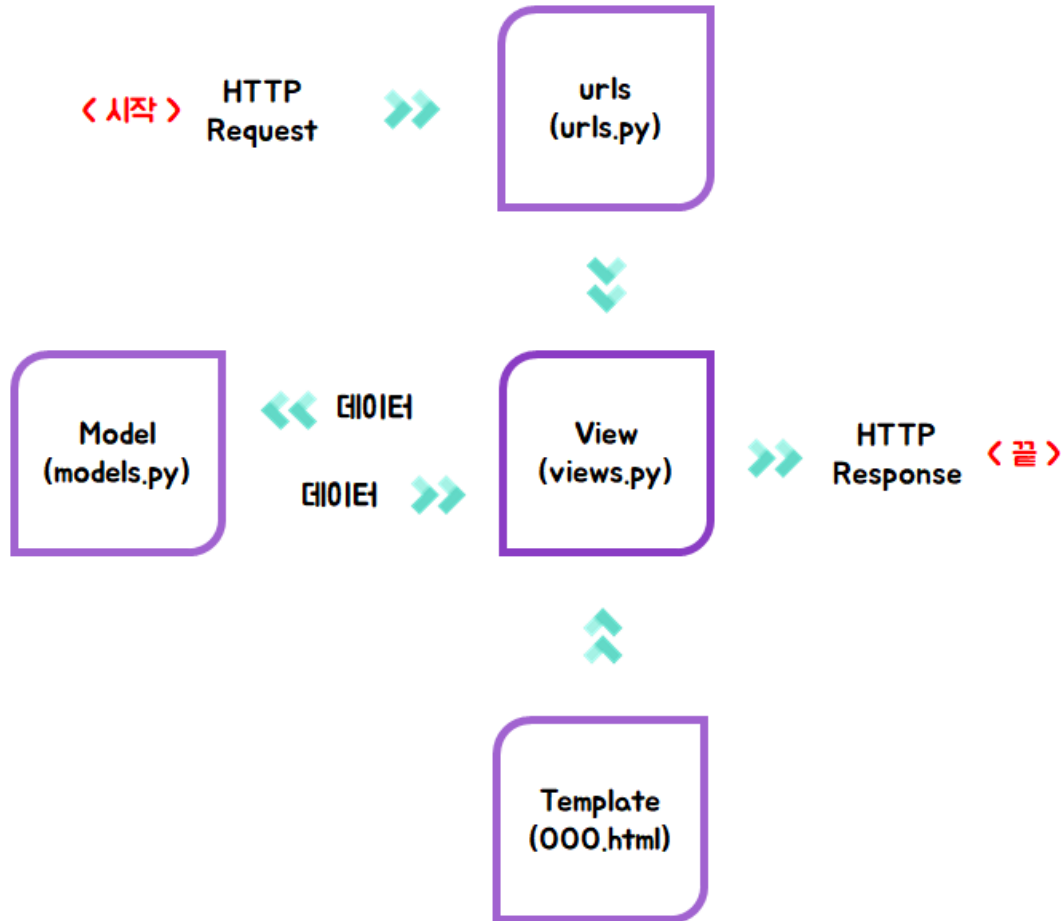


# Django MTV모델

## 장고 기본 구조



# Django 개발 흐름도



## < 시작 >

1. 웹서버에 http request이 들어옵니다.
2. urls에서 어떤 request인지 파악합니다.
3. view에서 파악한 request에 맞는 model과 template를 받습니다.
4. view는 template라는 뼈대에 model이라는 살을 붙입니다.
5. 완성된 html 파일을 http response를 통해 내보냅니다.



# Django 설치하기

# VSCODE 가상환경 구축하기

## 가상환경 구축하기

```
] python -m venv env    ← Virtual Env 가상환경 구축  
] .\env\Script\activate.bat  
](env) >
```

## 장고 설치하기

```
] (env) > pip install django  
] (env) > Django-admin --version  
4.0.6  <- 버전 확인
```

# Django 프로젝트 생성

# Django 프로젝트 생성하기

서비스를 개발할 프로젝트를 생성한다.(helloapp)

```
] (env) django-admin startproject helloapp .
```

서버 동작 구동하기

```
] (env) > python manage.py runserver  
Django version 4.0.6, using settings 'helloapp.settings'  
Starting development server at http://127.0.0.1:8000/  
Quit the server with CTRL-BREAK.
```

# Django 서버 동작상태 확인

django

View [release notes](#) for Django 4.0



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.



**Django Documentation**

Topics, references, & how-to's



**Tutorial: A Polling App**

Get started with Django



**Django Community**

Connect, get help, or contribute

모두 영어를 한글로

```
helloapp\settings.py
```

```
LANGUAGE_CODE = 'en-us'
```

```
TIME_ZONE = 'UTC'
```

를

```
LANGUAGE_CODE = 'ko-kr'
```

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



# Django 서버 동작상태 확인

django

Django 4.0 [릴리스 노트](#) 보기



성공적으로 설치되었습니다! 축하합니다!  
이 페이지는 어떤 URL도 지정되지 않았고, settings 파일에  
[DEBUG=True](#)가 설정되어 있을 때 표시됩니다.



Django 문서

주제, 레퍼런스, & 입문참조하  
다



튜토리얼: 폴링 애플리  
케이션

Django와 함께 시작하기



Django 커뮤니티

연결하고, 도움을 받거나 기여  
하기

한글로 모두 바뀜

- 항상 어떤 정보를 수정하기 위하여서는  
settings.py 파일을 수정

# Django 앱 생성(디렉토리)

# Django 앱(디렉토리) 생성하기

새로운 페이지를 생성한다.(hello)

```
] (env) django-admin startapp hello
```

새로 생성한 페이지 실행하기

```
] (env) > python manage.py runserver  
Django version 4.0.6, using settings 'helloapp.settings'  
Starting development server at http://127.0.0.1:8000/  
Quit the server with CTRL-BREAK.
```

# Django 서버 동작상태 후 상태확인

http://127.0.0.1:8000/hello

Page not found (404)

Request Method: GET

Request URL: http://127.0.0.1:8000/hello

Using the URLconf defined in helloapp.urls, Django tried these URL patterns, in this order:

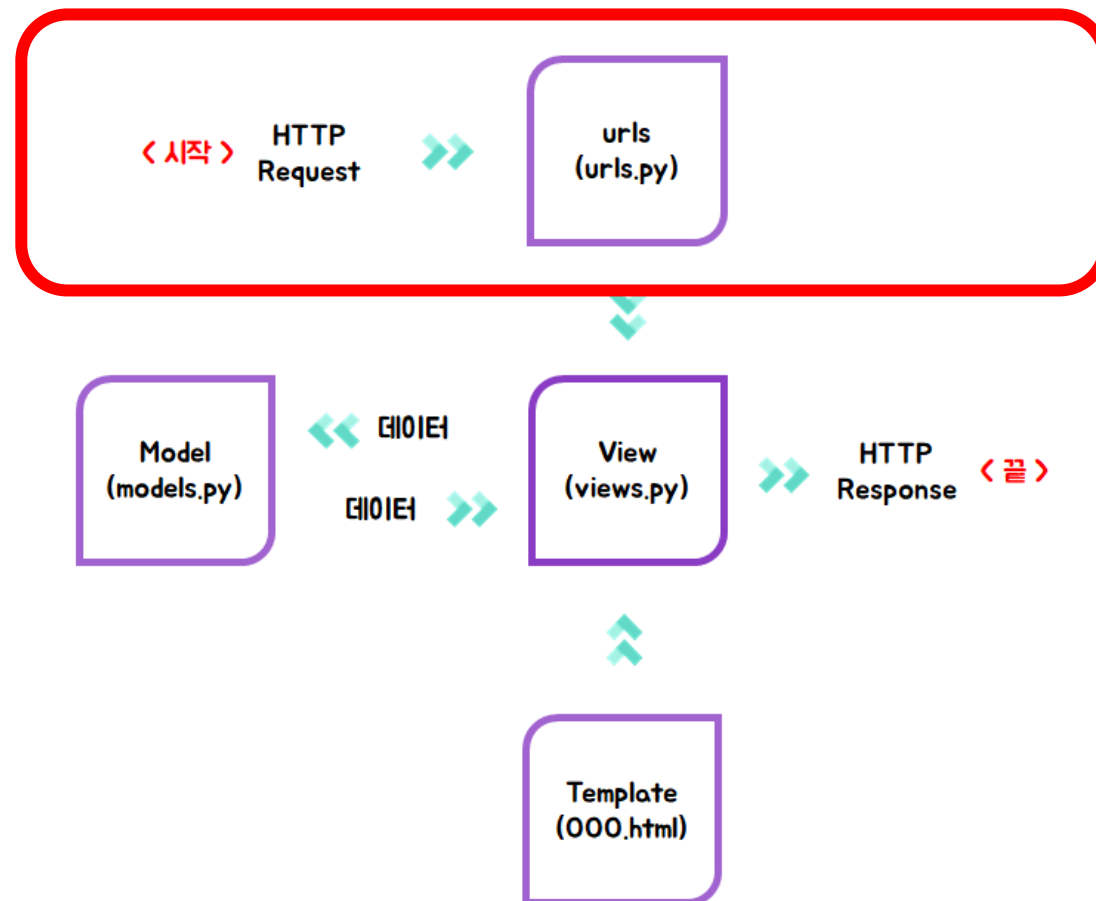
1. admin/

The current path, hello, didn't match any of these.

You're seeing this error because you have `DEBUG = True` in your Django settings file. Change that to `False`, and Django will display a standard 404 page.

<에러가 나오는 이유>

Urls.py에 정보가 없어서 불러오지 못하여 발생하는 에러임

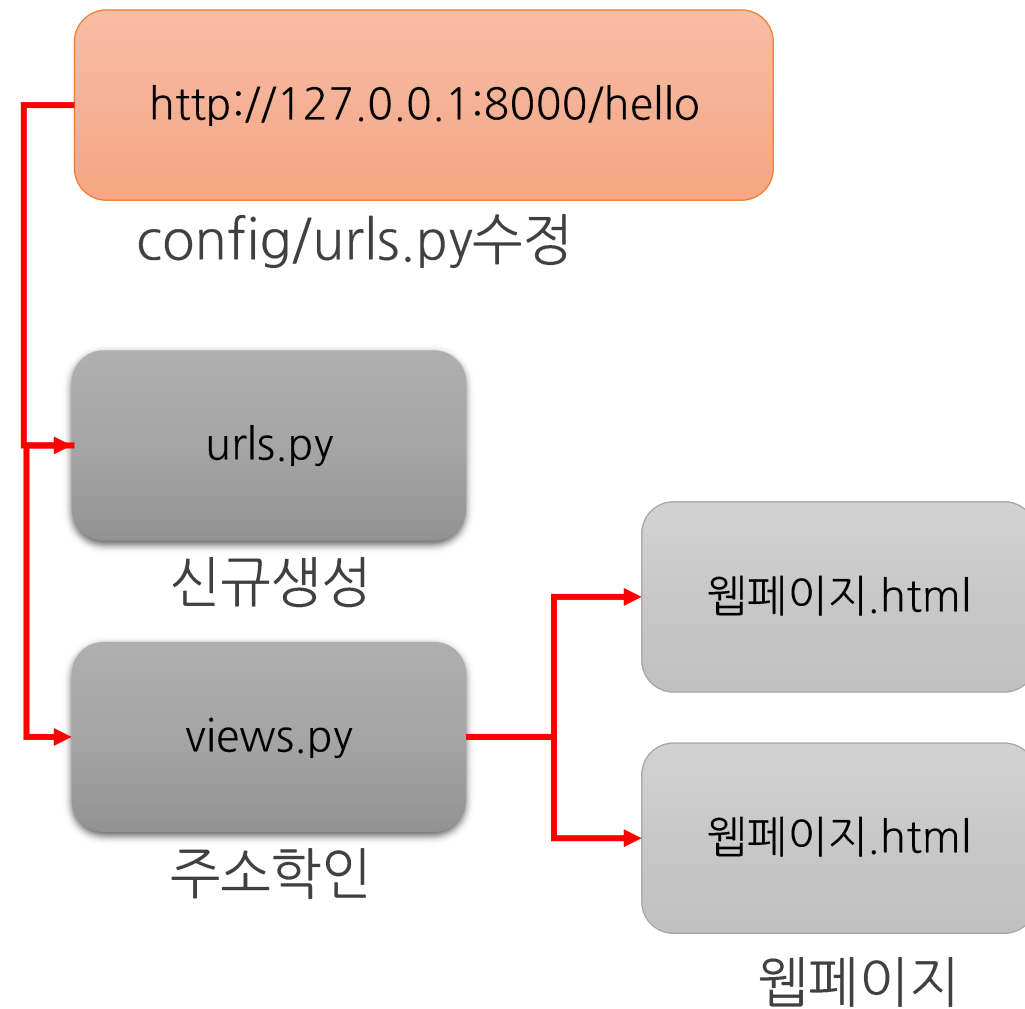


# 새로운 앱(디렉토리) urls 경로 설정

```
18
19 urlpatterns = [
20     path('admin/', admin.site.urls),
21 ]
22
```

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

```
urlpatterns = [
    path('admin/', admin.site.urls),
    path('hello/', include('hello.urls')),
]
```





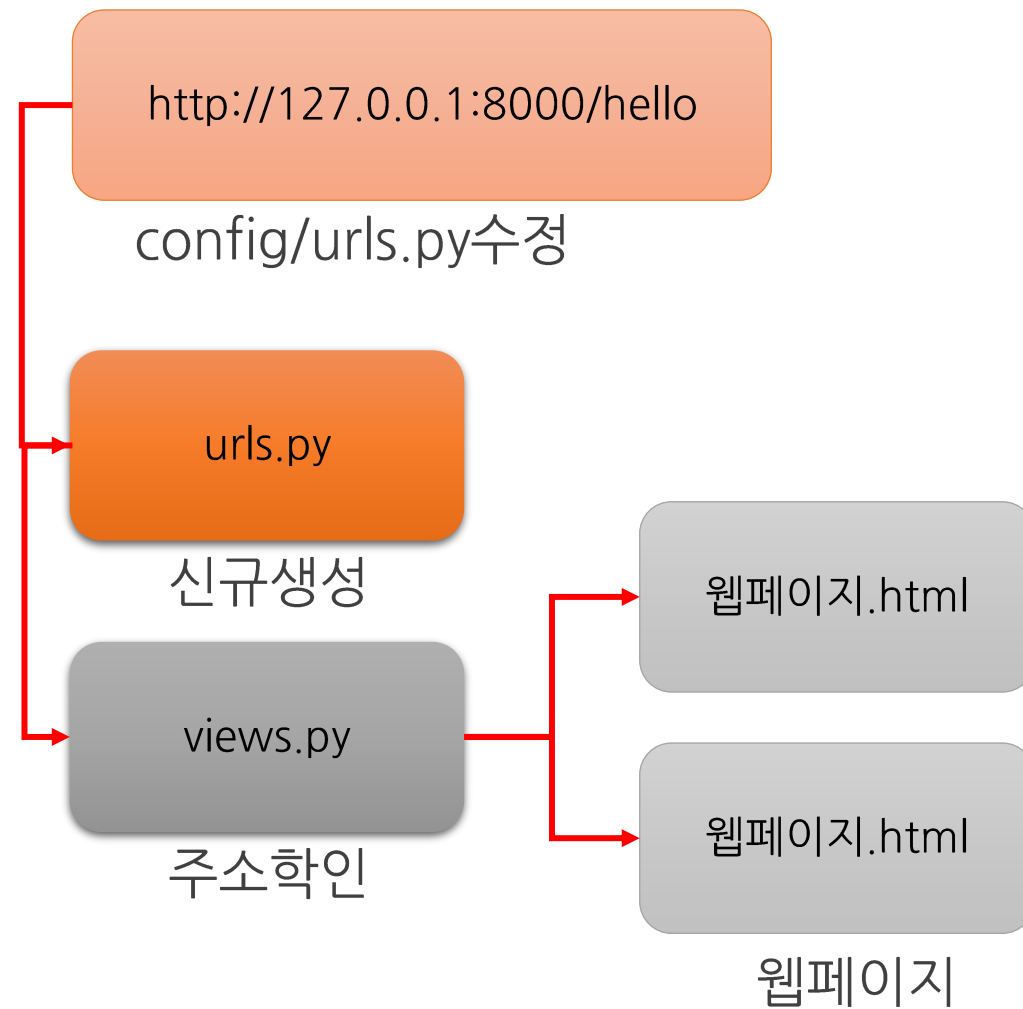
# 새로운 앱(디렉토리) urls.py 생성

hello/urls.py 신규생성

```
from django.urls import path
```

```
from . import views
```

```
urlpatterns = [  
    path("", views.index, name='index'),  
]
```



# 새로운 앱(디렉토리) views.py 수정

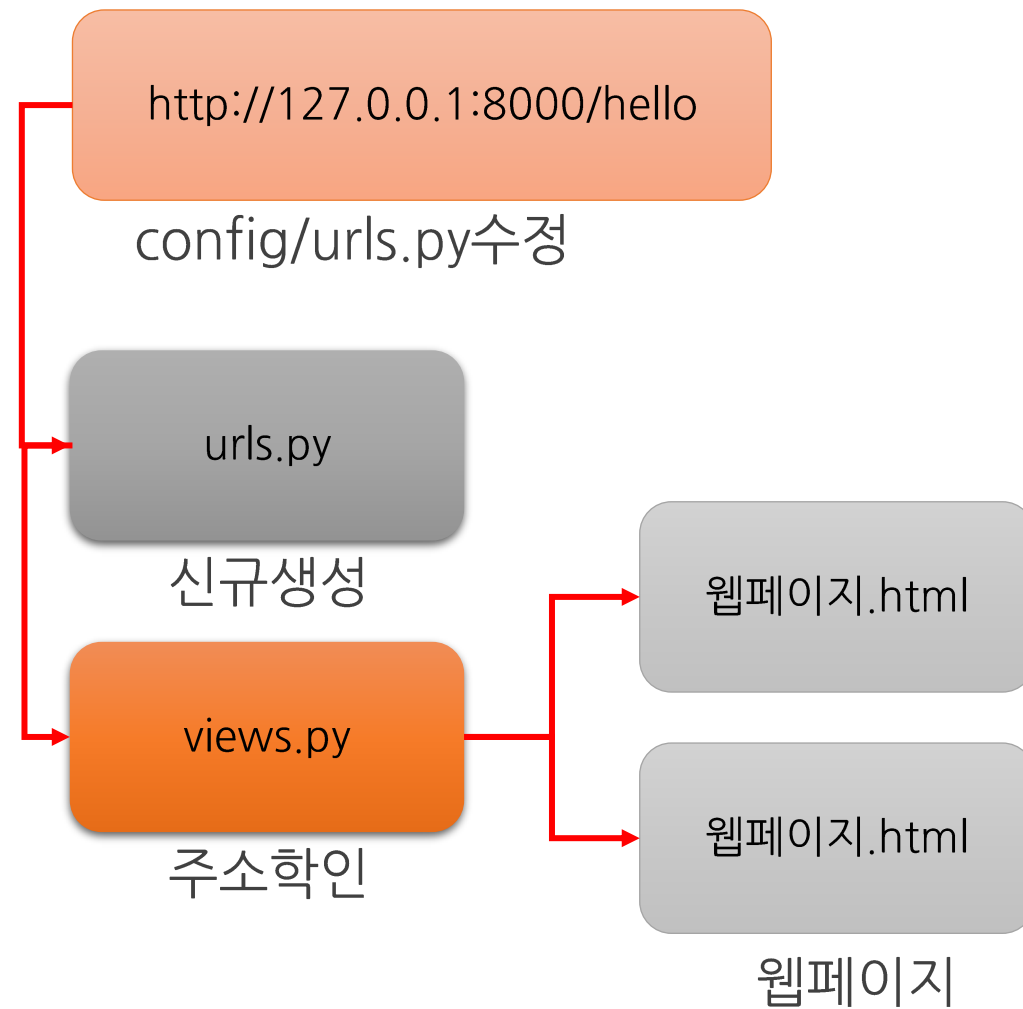
## hello/views.py 수정

```
from django.shortcuts import render
```

```
# Create your views here.
```

```
from django.http import HttpResponse
```

```
def index(request):  
    return HttpResponse("안녕하세요.  
hello 페이지 입니다.")
```



# Django 데이터베이스 연결

# 데이터베이스 생성 : helloweb

게시판 DB 생성( DB명 : hellowebdb, id :helloweb, pw : 1234 )

# 데이터베이스 생성

```
create database hellowebdb;
```

# 사용자 생성

```
create user 'helloweb'@'localhost' identified by '1234';
```

# 사용자 권한 생성

```
grant all privileges on hellowebdb.* to 'helloweb'@'localhost';
```

# 사용자 권한 적용

```
flush privileges;
```

# 데이터베이스 생성 : helloweb

MySQL 8.0 Command Line Client

```
Enter password: *****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 18
Server version: 8.0.30 MySQL Community Server - GPL

Copyright (c) 2000, 2022, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> create database hellowebdb;
Query OK, 1 row affected (0.02 sec)

mysql> create user 'helloweb'@'localhost' identified by '1234';
Query OK, 0 rows affected (0.01 sec)

mysql> grant all privileges on hellowebdb.* to 'helloweb'@'localhost';
Query OK, 0 rows affected (0.02 sec)

mysql> flush privileges;
Query OK, 0 rows affected (0.02 sec)

mysql> _
```



# 데이터베이스 접속 : helloweb workbench

## Setup New Connection

Connection Name:

Connection Method:

Standard (TCP/IP)

Parameters

SSL

Advanced

Hostname:

127.0.0.1

Port:

3306

Username:

root

Password:

Store in Vault ...

Clear

Default Schema:

## hellowebdb 데이터베이스 접속

Connection Name :

hellowebdb

Username:

helloweb

Password:

1234

# Django 데이터베이스 연결

# Django와 mysql(mariadb) 연결하기

(1) pymysql을 설치 <= mysql의 연동을 위하여 필요

```
] (env) pip install pymysql
```

# Django와 mysql(mariadb) 연결하기

데이터베이스 설정 helloapp / settings.py

```
DATABASES = {  
    'default': {  
        'ENGINE': 'django.db.backends.mysql',  
        'NAME': 'hellowebdb',  
        'USER': 'helloweb',  
        'PASSWORD': '1234',  
        'HOST': '127.0.0.1',  
        'PORT': 3306,  
    }  
}
```

# Django에 test1 앱 생성

새로운 페이지를 생성한다.(test1)

```
] (env) django-admin startapp test1
```

새로 생성한 페이지 실행하기

```
] (env) > python manage.py runserver
```

Django version 4.0.6, using settings 'helloapp.settings'  
Starting development server at <http://127.0.0.1:8000/>  
Quit the server with CTRL-BREAK.

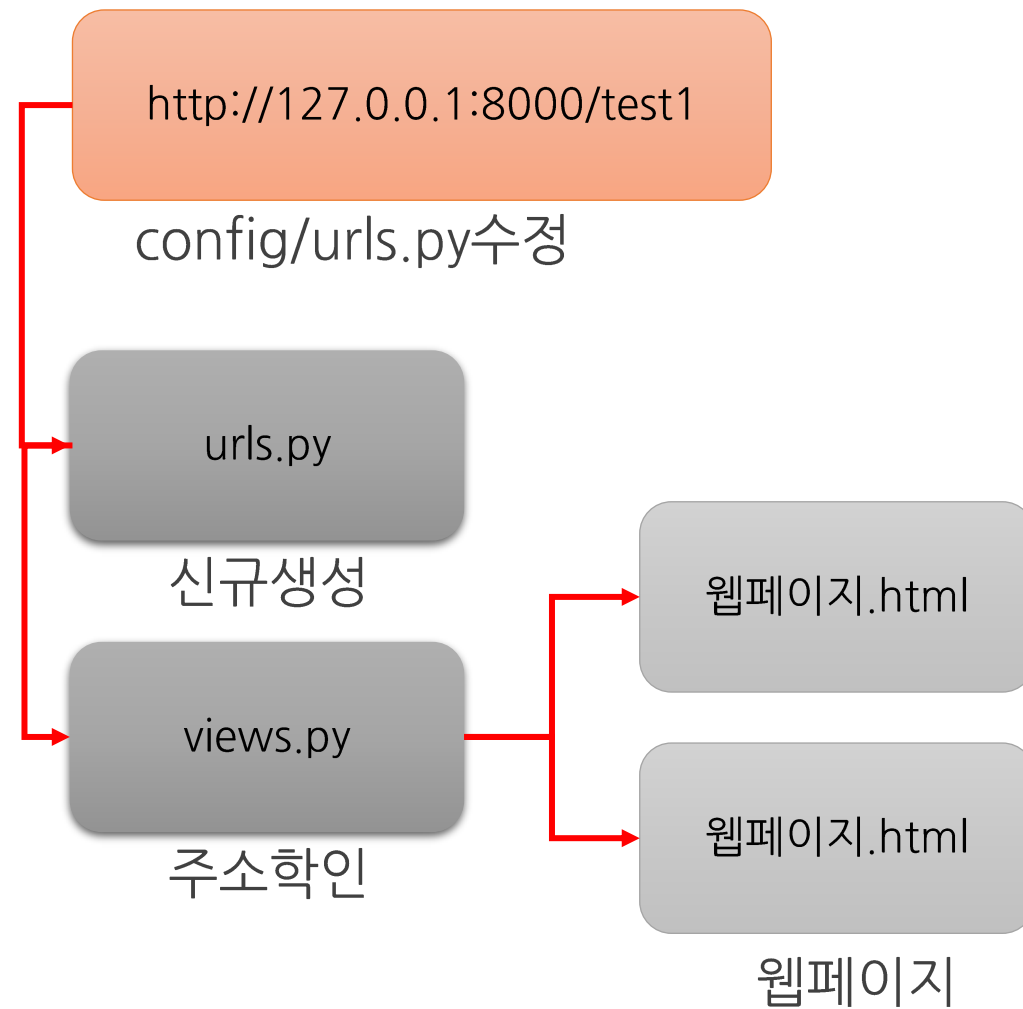


# 새로운 앱(디렉토리) urls 경로 설정

```
18
19 urlpatterns = [
20     path('admin/', admin.site.urls),
21 ]
22
```

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

```
urlpatterns = [
    path('admin/', admin.site.urls),
    ...
    path('test1/', include('test1.urls')),
]
```



# 새로운 앱(디렉토리) views.py 수정

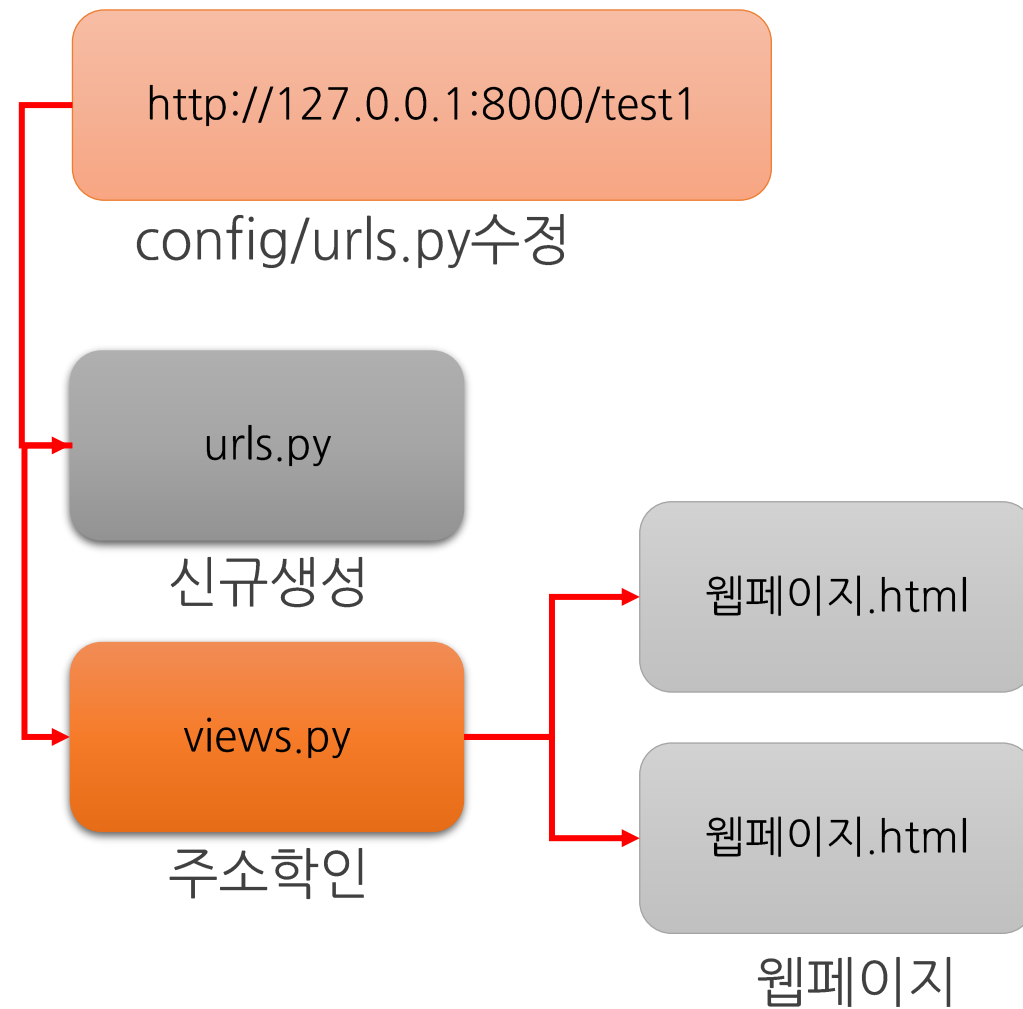
## test1/views.py 수정

```
from django.shortcuts import render
```

```
# Create your views here.
```

```
from django.http import HttpResponse
```

```
def index(request):  
    return HttpResponse("안녕하세요.  
test1 페이지 입니다.")
```



# 새로운 앱(디렉토리) models.py 수정

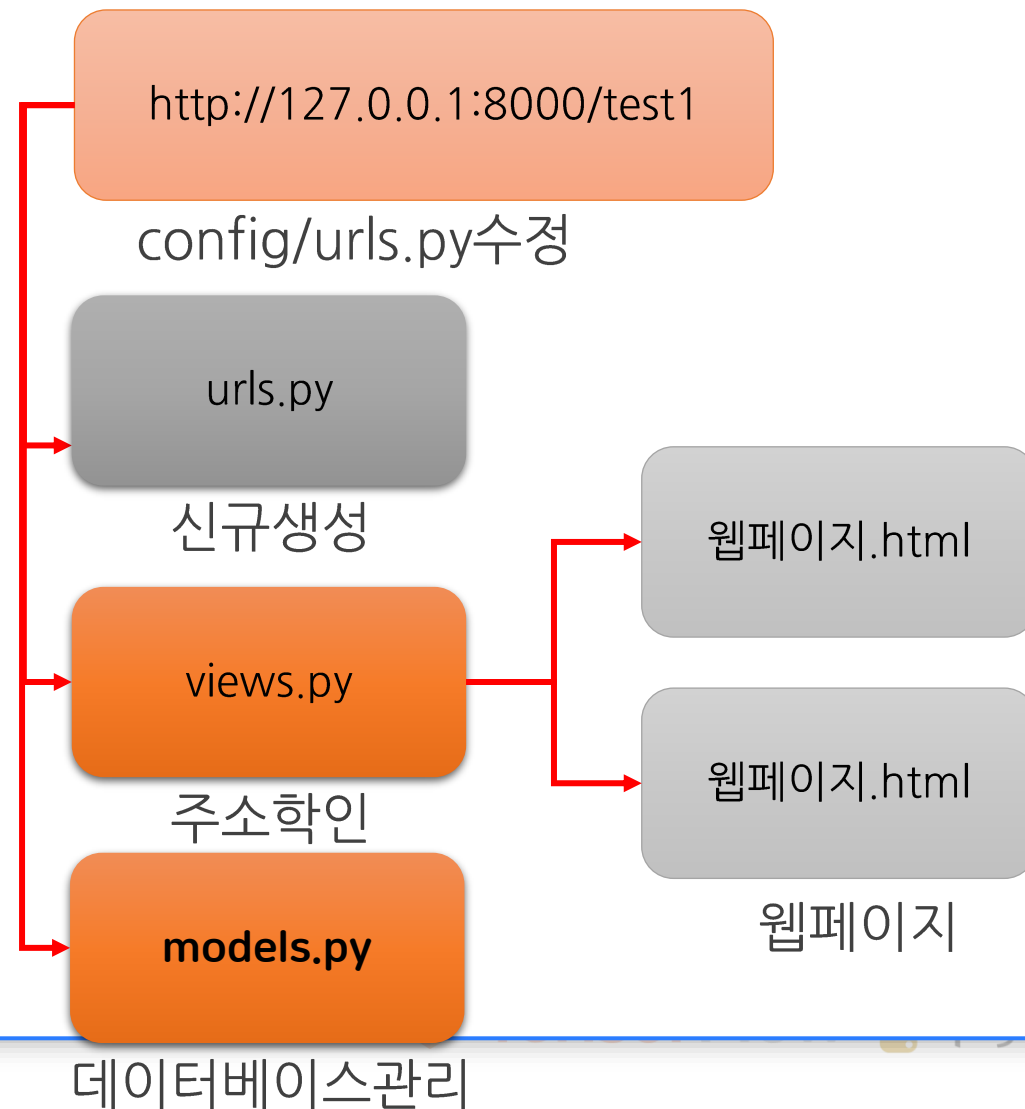
## test1/models.py 수정

```
from django.db import models
```

```
# Create your models here.
```

```
class Test1(models.Model):  
    idx = models.AutoField(primary_key=True)  
    titles = models.CharField(max_length=200)  
    bodys = models.TextField()  
    dates = models.DateTimeField()
```

```
def __str__(self):  
    return self.titles
```



# Django 관리권한 설정하기

helloapp / settings.py

```
ALLOWED_HOSTS = ['*']
```

```
INSTALLED_APPS = [
```

```
...
```

```
'django.contrib.staticfiles',
```

```
    'app1',
```

```
    'hello',
```

```
    'test1',
```

```
]
```

# Django 데이터베이스 갱신하기

데이터베이스를 갱신한다.

```
] (env) python manage.py makemigrations test1
```

```
] (env) python manage.py migrate
```

...

```
Applying auth.0011_update_proxy_permissions... OK
```

```
Applying auth.0012_alter_user_first_name_max_length... OK
```

```
Applying sessions.0001_initial... OK
```

# Django 관리자 생성하기

# Django 관리자 생성하기

```
] (env) python manage.py createsuperuser
```

사용자 이름 (leave blank to use 'matalcross.ai'): admin

이메일 주소:

Password:

Password (again):

<http://127.0.0.1:8000/admin>



# Django 관리자에 test1 데이터베이스 연결

## test1 / admin.py

```
from django.contrib import admin  
from . models import Test1
```

```
# Register your models here.  
admin.site.register(Test1)
```

### Django 관리

환영합니다, **ADMIN**. [사이트 보기](#) / [비밀번호 변경](#) / [로그아웃](#)

#### 사이트 관리

TEST1

Test1s

+ 추가    ✎ 변경

인증 및 권한

그룹

+ 추가    ✎ 변경

사용자(들)

+ 추가    ✎ 변경

#### 최근 활동

나의 활동

이용할 수 없습니다.

# Django 외부 template 사용

# Django 외부 template(html,css,icon) 사용

helloapp / settings.py <= templates 경로설정

```
import os
TEMPLATES = [
    {
        'BACKEND':
        'django.template.backends.django.DjangoTemplates',
        'DIRS': [],

        'DIRS': [os.path.join(BASE_DIR, 'templates')],
    },
]
```

=> 경로를 만들어준다.

# Django 외부 test1.html 만들기

helloapp / templates / test1.html

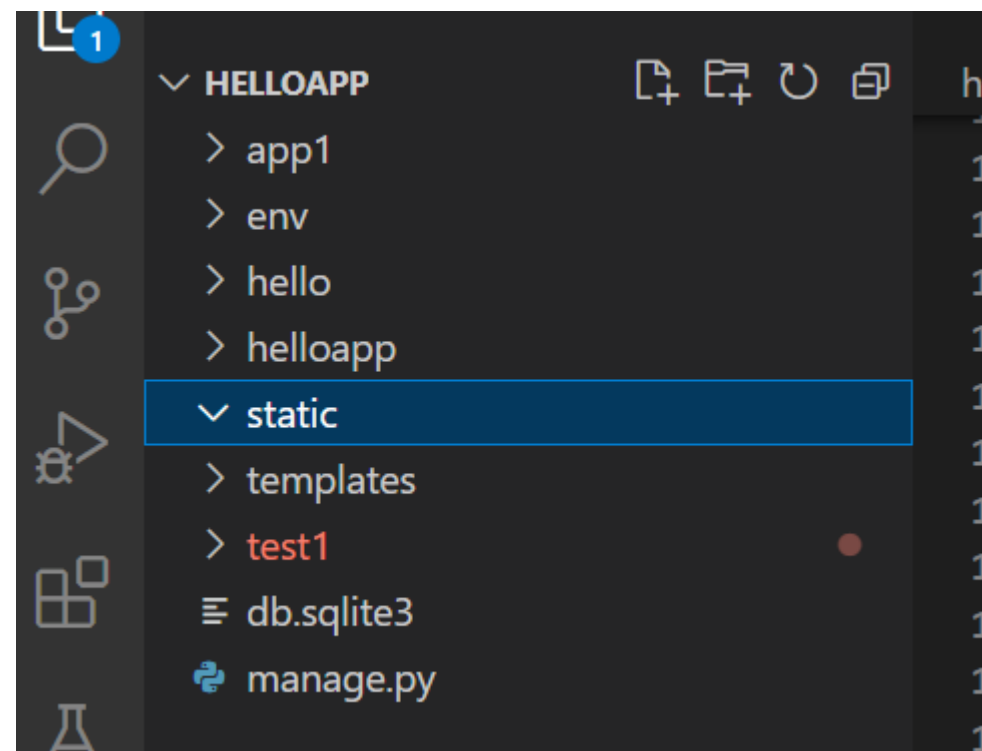
```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width,
initial-scale=1.0">
  <title>데이터베이스 테스트</title>
</head>
<body>
```

```
<h1>데이터베이스 테스트입니다</h1>
```

# Django 외부 template(html,css,icon) 사용

helloapp / settings.py <= static 경로설정

```
STATIC_URL = 'static/'  
STATIC_ROOT = os.path.join(BASE_DIR, 'staticfiles')  
STATICFILES_DIRS = (os.path.join(BASE_DIR, 'static'), )
```



# Django 외부 template 연결

# Django 외부 template 파일 변수 연결

helloapp / test1 / views.py 의 변수 외부 template에 연결

```
def index(request):  
    # return HttpResponse("안녕하세요. test1 페이지 입니다.")  
    test_msg = "외부변수가 잘 전달되는지 테스트 입니다."  
    return render(request, 'test1.html', {'message': test_msg})
```

test\_msg 변수를 message라는 외부의 인식하는 변수로 전달한다.

helloapp / templates / test1.html

```
<h1 class="text-center m-5">test1 데이터베이스 정보</h1>  
<h1 class="text-center m-5">{{message}}</h1>
```

# Django 외부 template 파일 연결

helloapp / templates / test1.html 수정

bootstrap 참고

```
<h1 class="text-center m-5">test1 데이터베이스 정보</h1>
```

```
<table class="table table-dark table-striped">
```

test1 데이터베이스 정보

외부변수가 잘 전달되는지 테스트 입니다.

| # | First | Last     | Handle |
|---|-------|----------|--------|
| 1 | Mark  | Otto     | @mdo   |
| 2 | Jacob | Thornton | @fat   |



# Django 외부 template에 DB데이터 연결

## helloapp / test1 / views.py 의 DB데이터 연결

```
from django.shortcuts import render  
from . models import Test1
```

```
def index(request):  
    # return HttpResponse("안녕하세요. test1 페이지 입니다.")  
  
    # test_msg = "외부변수가 잘 전달되는지 테스트 입니다."  
    # return render(request, 'test1.html', {'message': test_msg})
```

```
test1_data = Test1.objects.all()  
return render(request, 'test1.html', {'db_list': test1_data})
```

# Django 외부 template에 DB데이터 연결

helloapp / templates / test1.html

test1 데이터베이스 정보

| # | 순번             | 제목   | 내용   | 등록일자 |
|---|----------------|------|------|------|
| 1 | Mark           | Otto | @mdo | @mdo |
| * | 자료가 존재하지 않습니다. |      |      |      |

# Django 외부 template에 DB데이터 연결

helloapp / templates / test1.html

```
<table class="table table-dark table-striped">
  <thead>
    <tr>
      <th scope="col">#</th>
      <th scope="col">제목</th>
      <th scope="col">내용</th>
      <th scope="col">등록일자</th>
    </tr>
  </thead>
  <tbody>
    {% if db_list.count > 0 %}
    {% for item in db_list %}
    <tr>
      <th scope="row">{{item.idx}}</th>
      <td>{{item.titles}}</td>
      <td>{{item.bodys}}</td>
```

test1 데이터베이스 정보

| # | 제목      | 내용          | 등록일자                 |
|---|---------|-------------|----------------------|
| 1 | 테스트     | 테스트입니다.     | 2022년 7월 31일 7:08 오후 |
| 2 | 2번째 데이터 | 2번째 데이터입니다. | 2022년 7월 31일 7:08 오후 |

# 데이터베이스 테이블 생성



Table Name: test1

Schema: hellowebdb

Charset/Collation:





utf8

utf8\_bin

Engine:

InnoDB

Comments:

| Column Name   | Datatype     | PK                                  | NN                                  | UQ                                  | B                        | UN                       | ZF                       | AI                       | G                        | D                        |
|---|--------------|-------------------------------------|-------------------------------------|-------------------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|
|  idx     | INT          | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
|  titles | VARCHAR(200) | <input type="checkbox"/>            | <input checked="" type="checkbox"/> | <input type="checkbox"/>            | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
|  bodys | LONGTEXT     | <input type="checkbox"/>            | <input checked="" type="checkbox"/> | <input type="checkbox"/>            | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
|  dates | DATETIME     | <input type="checkbox"/>            | <input checked="" type="checkbox"/> | <input type="checkbox"/>            | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
|   |              | <input type="checkbox"/>            | <input type="checkbox"/>            | <input type="checkbox"/>            | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |

Online DDL  
Algorithm: Default Lock Type: Default

```
1 CREATE TABLE `hellowebdb`.`test1` (  
2   `idx` INT NOT NULL,  
3   `titles` VARCHAR(200) NOT NULL,  
4   `bodys` LONGTEXT NOT NULL,  
5   `dates` DATETIME NOT NULL,  
6   PRIMARY KEY (`idx`),  
7   UNIQUE INDEX `idx_UNIQUE` (`idx` ASC) VISIBLE)  
8 ENGINE = InnoDB  
9 DEFAULT CHARACTER SET = utf8;  
10
```

# 데이터베이스 테이블에 데이터 입력

## query에 입력

```
insert into test1 values(1, '테스트1', '테스트 내용입니다.', now())  
insert into test1 values(2, '서비스', '데이터베이스 테스트 입니다.',  
now())  
insert into test1 values(3, '서비스구축', '데이터를 저장하고 있습니다.',  
now())
```