Django

라이브러리 : 단순 활용가능한 도구들의 집합

프레임워크 : 소프트웨어의 특정 문제를 해결하기 위해서 상호 협력하는 클래스와 인터페이스의 집합

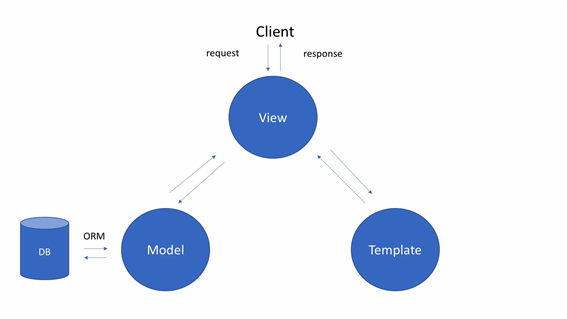
UI, DB연동, URL파싱, 세션 관리, 관리자페이지, Request파싱, 등등..

Django가 제공하는 것 : 폼, 개발 프로세스, 관리자, 보안 등

ㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡ

1. Django의 구성

모델 템플릿 뷰 (MTV 패턴)



view : 서버의 개념, 요청에 대한 응답을 하는 곳

template : view에서 response로 쓰이는 HTML 등

model : modeling을 통해 만들어짐, 추상적 개념, DB에 테이블 형태로 만들기 위한 설계

render 통해 template을 response로 view에게 전달

client ↔ view ↔ template & model ↔ DB (SQL 언어 사용, ORM (Object Relational Mapping -> 파이썬과 SQL 사이의 통역 역할))

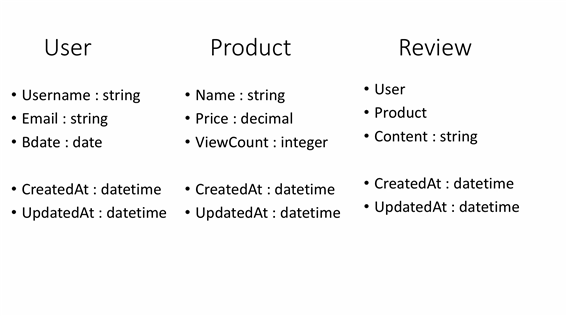
\* User라는 모델을 만들어 migrate 하여 UserTable을 DB에 저장!

객체가 언제 생성됐는지 언제 수정됐는지 중요! (created\_at / updated\_at)

모델의 각 컬럼이 어떤 데이터 타입인지가 중요!

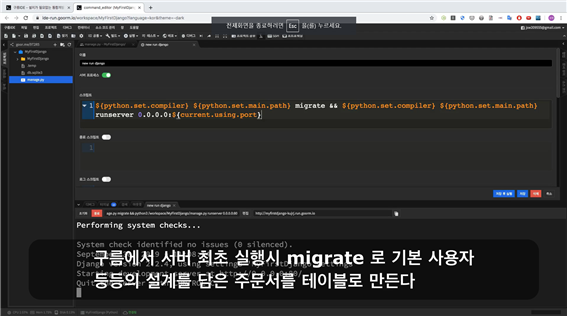
relation : model 간의 상호작용

모델링 과정에선 relation이 필요.



ㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡ

\* 모델링을 통한 테이블 생성



1. models.py에서 모델링을 한다

2. python manage.py makemigrations로 주문서를 만듬. (migration : 생성 될 테이블 정보를 담은 주문서)

3. python manage.py migrate로 주문서 내역대로 테이블을 생성한다.

ctrl + c로 서버 종료 가능

MyFirstDjango 파일에서

setting.py : Django의 설정을 담당하는 부분

urls.py :

1. 모델링을 하기 위해선 app 생성!

python manage.py startapp 앱이름

admin.py -> 관리자 페이지

views.py : request를 받아서 response를 줌

Wsgi.py : web server gateway interface) 웹서버와 장고 어플리케이션을 연결하는 pipeline

Manage.py : 다양한 것을 장고에서 실현시킴

App을 만들 때 중요한 부분

settings.py -> INSTALLED\_APPS에서 ‘앱이름’이 설치 되었음을 장고에게 알려주어야 함

(settings.py 에서 INSTALLED\_APPS 에 'posts' 추가하기!)

Ex> model을 만들어 보자

## python manage.py startapp posts (App 생성하기)

## posts 앱 안의 models.py

from django.db import models

class Post(models.Model):

title = models.CharField(max\_length=200)

content = models.TextField()

view\_count = models.IntegerField(default=0)

created\_at = models.DateTimeField(auto\_now\_add=True)

updated\_at = models.DateTimeField(auto\_now=True)

#charField : 문자열 데이터 타입 / max\_length : 최대 길이

TextField : 1000자 정도의 문자열 데이터 타입

InterField : 정수형 데이터 타입(default=0 > 기본값 초기화)

DateTimeField : 시간 날짜 데이터 타입

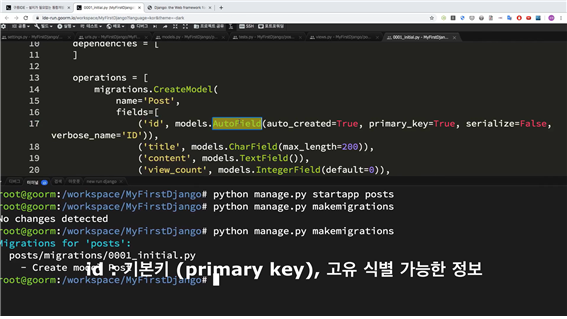
auto\_now\_add=True: 생성 당시 현재시간 저장

auto\_now=True : 생성, 수정될 때 현재시간 저장

2. 모델링 완료 후 테이블을 생성하기 위한 주문서로 만들기

python manage.py makemigrations

-> posts/migrations/0001\_initial.py 생성!



field에 id라는게 생성 됌

primary\_key -> 기본 키

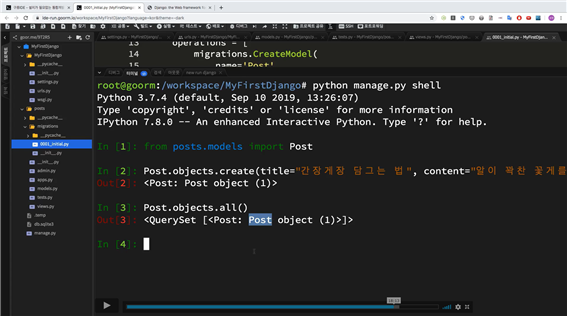
ex> posts 라는 블로그 생성 후 첫 게시글은 id=1 두 번째 게시글은 id=2,

한 테이블 안에서 동일한 키를 같는 객체는 있으면 안됨.

3. 이렇게 만든 주문서로 DB에 테이블로 만들어달라!

-> python manage.py migrate

DB에 테이블 형태로 post를 만듦.



\* 테이블 확인, 생성 명령어

ORM (SQL과 Python의 통역사)을 통해 확인 가능

python manage.py shell 입력

1. ㅏㄴ든 모델(클래스)를 가져와야 함 -> from posts.models import Post

2. Post.objects.create(title=”간장게장 담그는 법”, content=”알이 꽉찬 꽃게를~”)

Post.objects.all() -> 데이터셋을 보여줌

\* object의 attribute 찾기

->

post = Post.objects.get(id=1) # 1. 변수를 설정 후 서치

post.title -> 간장게장 담그는 법 (게시글 제목)

post.content -> 게시글 내용

post.view\_count -> 조회수

ㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡ

* 관리자 페이지 활용하기

만들어진 테이블을 관리자 페이지에서 확인하는 법

Admin.py에서 확인 가능하다.

서버 실행 시키기

-> 터미널에 python manage.py runserver (0.0.0.0:8000) 입력

(서버의 IP를 변경할경우 포트와 함께 입력)

-> http://~~~/admin 으로 들어감