WAFFLE
STUDIO

Django에서의 MVC와 Class based view

Instructors:
김찬욱 @narayo9

2022.09.20 (화) 19:30

서울대학교 앱/웹개발 동아리 와플스튜디오

과제 0 review

과제 0 review: learned

놀랍게도 우리는 이런 것들을 모두 배웠습니다!

- Function based view와 **Class based view**
- **Models** and admin sites
- **Serializations**

사전준비

Docker 설치

- Postgresql docker 설치 및 실행

목차

1. SQL 기초
2. Models
3. Why class based views?
4. 블로그 API 만들어보기

1. SQL 기초

Why SQL?

Why SQL?

상태를 저장하기 위해 데이터베이스가 필요

State persist

- SQL. Redis. NoSQL....

읽을거리: Database should be ACID

위키백과

실습: sql w3 schools

링크

읽을거리: Some of The Most Important SQL Commands

- `SELECT` - extracts data from a database
- `UPDATE` - updates data in a database
- `DELETE` - deletes data from a database
- `INSERT INTO` - inserts new data into a database
- `CREATE DATABASE` - creates a new database
- `ALTER DATABASE` - modifies a database
- `CREATE TABLE` - creates a new table
- `ALTER TABLE` - modifies a table
- `DROP TABLE` - deletes a table

읽을거리: DQL / DML / DDL

링크

DQL: Data Query Language. (SELECT)

DML: Data Manipulation Language. (INSERT, UPDATE, DELETE, ...)

DDL: Data Definition Language(CREATE TABLE, ...)

실습: sql w3 schools

DML

- SQL Where
- SQL Update
- SQL Delete

DDL

- SQL Create Table

읽을거리: ORM

여러분은 이 모든 걸 이미 사용하셨어요!

링크: [What is an ORM, how does it work, and how should I use one?]

읽을거리: Django migrations

- 링크

- **migrate**, which is responsible for applying and unapplying migrations.
- **makemigrations**, which is responsible for creating new migrations based on the changes you have made to your models.

실습: Pycharm으로 확인해보기

Startproject: waffle blog

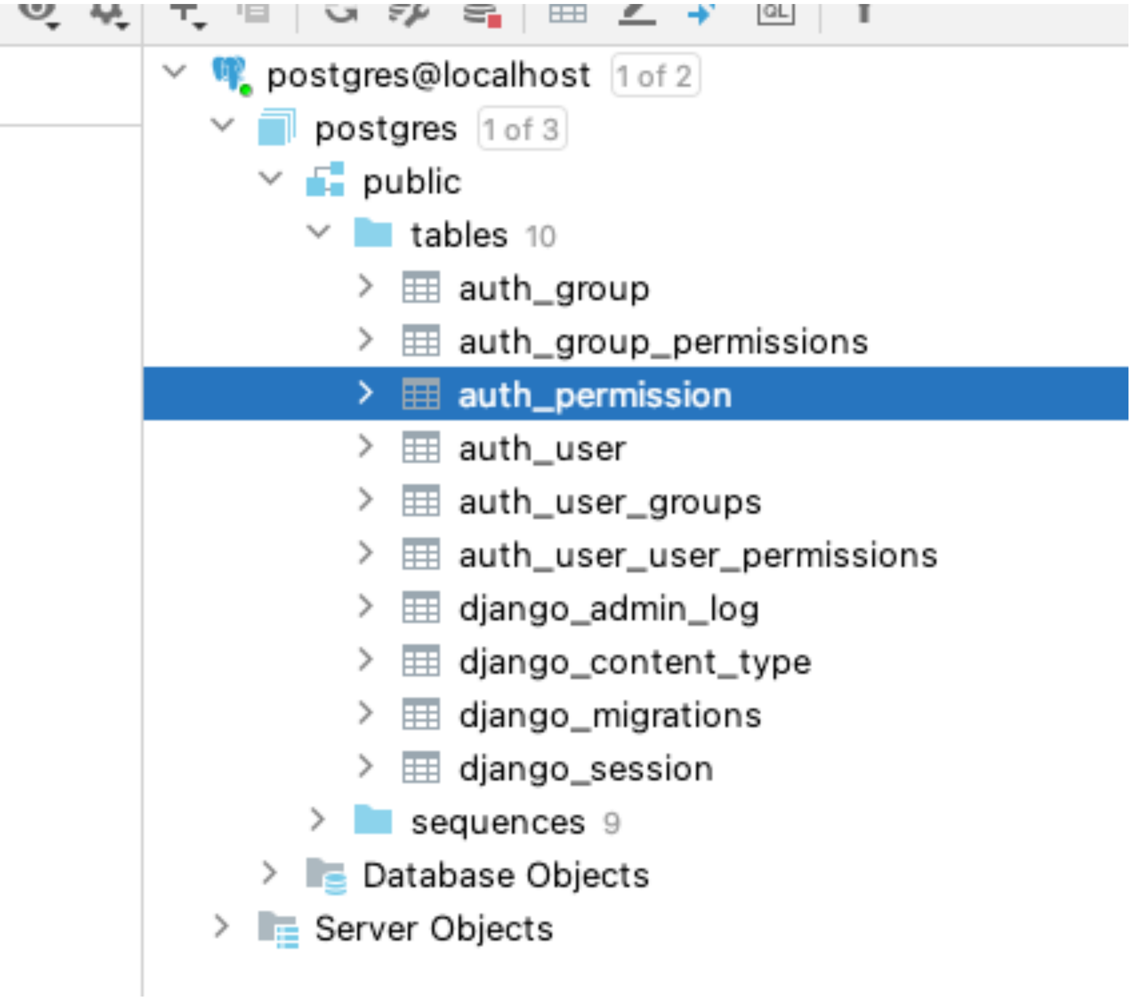
Do Migrate

psycopg2 말고 psycopg2-binary 설치할 것!

세팅 참고

Apple Sillicon에서 알려진 이슈

실습: Pycharm으로 확인해보기



postgres@localhost 1 of 2
postgres 1 of 3
public
tables 10
auth_group
auth_group_permissions
auth_permission
auth_user
auth_user_groups
auth_user_user_permissions
django_admin_log
django_content_type
django_migrations
django_session
sequences 9
Database Objects
Server Objects

2. Models

참고: Django models

정의

A **model** is the single, definitive source of information about your data. It contains the essential fields and behaviors of the data you're storing.
Generally, each model maps to a single database table.

Settings 수정하기

Do this by editing your settings file and changing the **INSTALLED_APPS** setting to add the name of the module that contains your models.py.

실습: Post modeling

Blog Post를 모델링 해보아요

- Settings 수정하기
- Migration 확인 후 db table 변경 확인하기

Field options

null

If True, Django will store empty values as NULL in the database. Default is False.

blank

If True, the field is allowed to be blank. Default is False. Note that this is different than null. null is purely database-related, whereas blank is validation-related. If a field has blank=True, form validation will allow entry of an empty value. If a field has blank=False, the field will be required.

CharField null=True, blank=True

이 둘은 무슨 차이일까요?

```
class Post(models.Model):
    title = models.CharField(max_length=100, blank=True)
```

```
class Post(models.Model):
    title = models.CharField(max_length=100, null=True)
```

CharField null=True, blank=True

이러면 어떤 일이 일어날까요?

```
class Post(models.Model):
    title = models.CharField(max_length=100, null=True, blank=True)
```

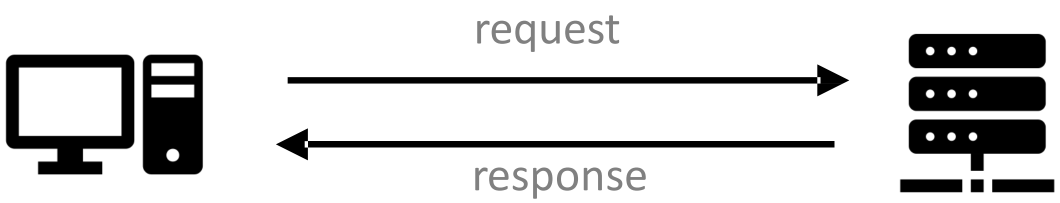
읽을거리: model best practices

구글에 검색하는 꿀팁: ~ best practices

링크

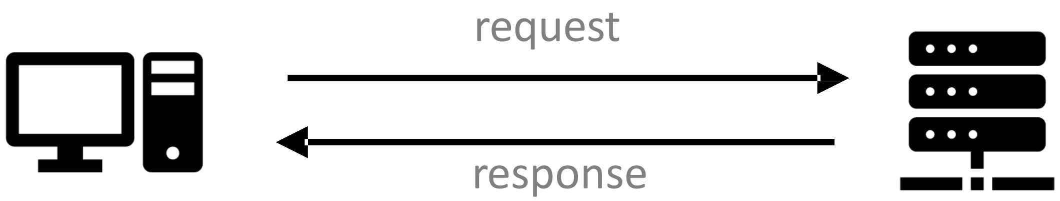
3. Why class based views?

Recap: 서버란 무엇일까요



가장 간단한: Function based view

```python
def index(request):
    return HttpResponse("Hello, world. You're at the p.")
```



실습: Function based views로 ViewSet 만들기

읽을거리: FBV vs CBV

Quora 질답

4. 블로그 API 만들기

Q&A