

Django 웹 프레임워크

- 맛집 공유 사이트

Mini-Project

장고 프레임워크 미니 프로젝트

◦ 맛집 공유 사이트 - 메인

맛집 공유 사이트 x +

← → ↻ ⓘ 127.0.0.1:8000

맛집 공유 사이트

made by Django

Category 상세내용을 보려면 클릭하세요. +

맛집 추가하기

Email 수신자와 제목, 인사말을 적어주세요.

수신자 콤마(,)로 구분해서 여러명에게 보낼 수 있습니다.

수신자를 적어주세요.

제목

제목을 적어주세요.

인사말

인사말을 적어주세요.

이메일 발송하기

Mini-Project

장고 프레임워크 미니 프로젝트

◦ 맛집 공유 사이트 – 맛집 추가하기

맛집 추가하기

127.0.0.1:8000/restaurantCreate/

맛집 추가하기

made by Django

카테고리

맛집 이름

관련 링크

상세 내용

장소 키워드

Mini-Project

장고 프레임워크 미니 프로젝트

◦ 맛집 공유 사이트 – 카테고리 추가하기



The screenshot shows a web browser window with the title '맛집 추가하기' and the URL '127.0.0.1:8000/categoryCreate/'. The page content is as follows:

카테고리 추가하기

made by Django

기본그룹

한식

중식

추가할 카테고리명을 입력하세요.

추가

홈으로

Mini-Project

장고 프레임워크 미니 프로젝트

◦ 맛집 공유 사이트 – 이메일 발송하기

맛집 공유 사이트

← → ↺ 127.0.0.1:8000

맛집 공유 사이트

made by Django

Category

상세내용을 보려면 클릭하세요.

+

한식

☐ 채근담

양식

☐ 알라프리마

일식

중식

맛집 추가하기

Email

수신자와 제목, 인사말을 적어주세요.

수신자

를마(.)로 구분해서 여러명에게 보낼 수 있습니다.

smcpacs@naver.com

제목

한식 채근담입니다.

인사말

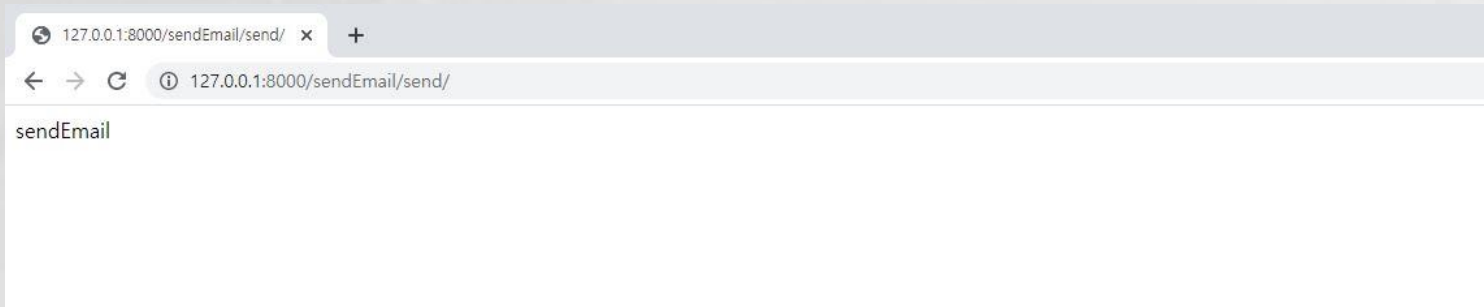
안녕하세요.
한식 채근담 맛집입니다.
즐겁니다.
다음에 꼭 오세요

이메일 발송하기

Mini-Project

장고 프레임워크 미니 프로젝트

- 맛집 공유 사이트 – 이메일 발송하기



장고에서의 애플리케이션 개발 방식

MVT 코딩 순서

- **프로젝트 뼈대 만들기** : 프로젝트 및 앱 개발에 필요한 디렉토리와 파일 생성
- **모델 코딩하기** : 테이블 관련 사항을 개발(models.py, admin.py 파일)
- **URLconf 코딩하기** : URL 및 뷰 매핑 관계를 정의(urls.py 파일)
- **템플릿 코딩하기** : 화면 UI 개발(templates/애플리케이션/디렉토리 하위의 *.html 파일들)
- **뷰 코딩하기** : 애플리케이션 로직 개발(views.py 파일)

프로젝트 뼈대 만들기

프로젝트 뼈대 만들기 순서 명령

- > `django-admin startproject RestaurantShare` // RestaurantShare 라는 프로젝트를 생성
- > `python manage.py startapp shareRes` // shareRes 라는 애플리케이션을 생성
- > `notepad settings.py` //설정 파일을 확인 및 수정
- > `python manage.py migrate` //데이터베이스에 기본 테이블을 생성
- > `python manage.py runserver` //현재까지 작업을 개발용 웹 서버로 확인

프로젝트 뼈대 만들기

프로젝트 생성

- ToDoList라는 프로젝트 만듦
- (base) C:\MyTest>django-admin startproject RestaurantShare

```
(base) C:\MyTest>django-admin startproject RestaurantShare  
(base) C:\MyTest>
```

프로젝트 뼈대 만들기

프로젝트 생성

o RestaurantShare 구성

```
(base) C:\MyTest>cd RestaurantShare

(base) C:\MyTest\RestaurantShare>dir
C 드라이브의 볼륨에는 이름이 없습니다.
볼륨 일련 번호: 421A-9EBA

C:\MyTest\RestaurantShare 디렉터리

2021-08-26 오후 01:57 <DIR> .
2021-08-26 오후 01:57 <DIR> ..
2021-08-26 오후 01:57      693 manage.py
2021-08-26 오후 01:57 <DIR> RestaurantShare
                1개 파일                693 바이트
                3개 디렉터리 18,689,232,896 바이트 남음
```

프로젝트 뼈대 만들기

애플리케이션 생성 – shareRes

- 프로젝트 루트 디렉토리 RestaurantShare으로 이동해서 shareRes라는 애플리케이션을 만드는 명령을 실행
- (base) C:\WMyTest>cd RestaurantShare
- (base) C:\WMyTest\RestaurantShare >python manage.py startapp shareRes

```
(base) C:\WMyTest\RestaurantShare>python manage.py startapp shareRes  
(base) C:\WMyTest\RestaurantShare>
```

프로젝트 뼈대 만들기

애플리케이션 생성 - shareRes

- shareRes라는 애플리케이션 폴더 확인
- (base) C:\MyTest\RestaurantShare\shareRes>dir

```
(base) C:\MyTest\RestaurantShare>cd shareRes
(base) C:\MyTest\RestaurantShare\shareRes>dir
C 드라이브의 볼륨에는 이름이 없습니다.
볼륨 일련 번호: 421A-9EBA

C:\MyTest\RestaurantShare\shareRes 디렉터리

2021-08-26 오후 07:05 <DIR>          .
2021-08-26 오후 07:05 <DIR>          ..
2021-08-26 오후 07:05          66 admin.py
2021-08-26 오후 07:05        154 apps.py
2021-08-26 오후 07:05 <DIR>          migrations
2021-08-26 오후 07:05          60 models.py
2021-08-26 오후 07:05          63 tests.py
2021-08-26 오후 07:05          66 views.py
2021-08-26 오후 07:05           0 __init__.py
                6개 파일              409 바이트
                3개 디렉터리 18,652,057,600 바이트 남음
```

프로젝트 뼈대 만들기

애플리케이션 생성 – sendEmail

- 프로젝트 루트 디렉토리 RestaurantShare으로 이동해서 sendEmail라는 애플리케이션을 만드는 명령을 실행
- (base) C:\WMyTest>cd RestaurantShare
- (base) C:\WMyTest\RestaurantShare> python manage.py startapp sendEmail

```
(base) C:\WMyTest\RestaurantShare>python manage.py startapp sendEmail  
(base) C:\WMyTest\RestaurantShare>
```

프로젝트 뼈대 만들기

애플리케이션 생성 - sendEmail

- sendEmail라는 애플리케이션 폴더 확인
- (base) C:\MyTest\RestaurantShare\sendEmail>dir

```
(base) C:\MyTest\RestaurantShare>cd sendEmail
(base) C:\MyTest\RestaurantShare\sendEmail>dir
C 드라이브의 볼륨에는 이름이 없습니다.
볼륨 일련 번호: 421A-9EBA

C:\MyTest\RestaurantShare\sendEmail 디렉터리

2021-08-26 오후 07:13 <DIR>          .
2021-08-26 오후 07:13 <DIR>          ..
2021-08-26 오후 07:13                66 admin.py
2021-08-26 오후 07:13               156 apps.py
2021-08-26 오후 07:13 <DIR>          migrations
2021-08-26 오후 07:13               60 models.py
2021-08-26 오후 07:13               63 tests.py
2021-08-26 오후 07:13               66 views.py
2021-08-26 오후 07:13                0 __init__.py
                6개 파일                411 바이트
                3개 디렉터리 18,648,989,696 바이트 남음
```

프로젝트 뼈대 만들기

애플리케이션 생성

- shareRes, sendEmail라는 두개의 애플리케이션 폴더 확인
- (base) C:\MyTest\RestaurantShare>dir

```
(base) C:\MyTest\RestaurantShare>dir
C 드라이브의 볼륨에는 이름이 없습니다.
볼륨 일련 번호: 421A-9EBA

C:\MyTest\RestaurantShare 디렉터리

2021-08-26 오후 07:13 <DIR> .
2021-08-26 오후 07:13 <DIR> ..
2021-08-26 오후 01:57 693 manage.py
2021-08-26 오후 07:05 <DIR> RestaurantShare
2021-08-26 오후 07:13 <DIR> sendEmail
2021-08-26 오후 07:05 <DIR> shareRes
                1개 파일                693 바이트
                5개 디렉터리 18,649,366,528 바이트 남음
```


프로젝트 뼈대 만들기

프로젝트 설정 파일 변경

- 프로젝트에 필요한 설정 값들은 **settings.py** 파일에 지정
- settings.py 파일은 프로젝트의 전반적인 사항들을 설정해주는 곳으로, 루트 디렉토리를 포함한 각종 디렉토리의 위치, 로그의 형식, 프로젝트에 포함된 애플리케이션의 이름 등이 지정되어 있음

```
(base) C:\MyTest\RestaurantShare\RestaurantShare>dir
C 드라이브의 볼륨에는 이름이 없습니다.
볼륨 일련 번호: 421A-9EBA

C:\MyTest\RestaurantShare\RestaurantShare 디렉터리

2021-08-26 오후 07:05 <DIR>          .
2021-08-26 오후 07:05 <DIR>          ..
2021-08-26 오후 01:57             423 asgi.py
2021-08-26 오후 01:57          3,391 settings.py
2021-08-26 오후 01:57             778 urls.py
2021-08-26 오후 01:57             423 wsgi.py
2021-08-26 오후 01:57              0 __init__.py
2021-08-26 오후 07:05 <DIR>          __pycache__
                5개 파일             5,015 바이트
                3개 디렉터리 18,649,784,320 바이트 남음

(base) C:\MyTest\RestaurantShare\RestaurantShare>notepad settings.py
```


프로젝트 뼈대 만들기

프로젝트 설정 파일 변경

- ALLOWED_HOSTS 항목을 적절하게 지정
 - `ALLOWED_HOSTS = ['192.168.35.61', 'localhost', '127.0.0.1']`
- 애플리케이션들은 모두 설정 파일에 등록
(shareRes, sendEmail, 애플리케이션도 등록)
 - `INSTALLED_APPS = [~, 'shareRes', 'sendEmail',]`
- 프로젝트에 사용할 데이터베이스 엔진
 - 장고는 디폴트로 SQLite3 데이터베이스 엔진을 사용하도록 설정
- 타임존 지정(기본은 세계표준시(UTC)로 되어 있음. 한국시간을 변경)
 - `#TIME_ZONE = 'UTC'`
 - `TIME_ZONE = 'Asia/Seoul'`

프로젝트 뼈대 만들기

기본 테이블 생성

- migrate 명령은 데이터베이스에 변경사항이 있을 때 반영해주는 명령
- (base) C:\MyTest\ToDoList>python manage.py migrate

```
(base) C:\MyTest\RestaurantShare>python manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, sessions
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying auth.0001_initial... OK
  Applying admin.0001_initial... OK
  Applying admin.0002_logentry_remove_auto_add... OK
  Applying admin.0003_logentry_add_action_flag_choices... OK
  Applying contenttypes.0002_remove_content_type_name... OK
  Applying auth.0002_alter_permission_name_max_length... OK
  Applying auth.0003_alter_user_email_max_length... OK
  Applying auth.0004_alter_user_username_opts... OK
  Applying auth.0005_alter_user_last_login_null... OK
  Applying auth.0006_require_contenttypes_0002... OK
  Applying auth.0007_alter_validators_add_error_messages... OK
  Applying auth.0008_alter_user_username_max_length... OK
  Applying auth.0009_alter_user_last_name_max_length... OK
  Applying auth.0010_alter_group_name_max_length... OK
  Applying auth.0011_update_proxy_permissions... OK
  Applying auth.0012_alter_user_first_name_max_length... OK
  Applying sessions.0001_initial... OK
```

프로젝트 뼈대 만들기

기본 테이블 생성

- o (base) C:\MyTest\RestaurantShare>dir

```
(base) C:\MyTest\RestaurantShare>dir
C 드라이브의 볼륨에는 이름이 없습니다.
볼륨 일련 번호: 421A-9EBA

C:\MyTest\RestaurantShare 디렉터리

2021-08-26 오후 07:23 <DIR> .
2021-08-26 오후 07:23 <DIR> ..
2021-08-26 오후 07:23      131,072 db.sqlite3
2021-08-26 오후 01:57       693 manage.py
2021-08-26 오후 07:05 <DIR> RestaurantShare
2021-08-26 오후 07:23 <DIR> sendEmail
2021-08-26 오후 07:23 <DIR> shareRes
                2개 파일              131,765 바이트
                5개 디렉터리  18,647,285,760 바이트 남음
```

프로젝트 뼈대 만들기

작업 확인하기

- 프로젝트의 뼈대에 해당하는 프로젝트 디렉토리, 애플리케이션 디렉토리를 비롯해 관련 파일들 그리고 사용자 및 그룹 테이블을 만들었음
- 확인을 위해서 웹 서버를 실행하고, 그 웹 서버에 접속
- 장고에서는 개발 과정 도중에 현재 상태를 확인해볼 수 있도록 **runserver**라고 하는 간단한 테스트용 웹 서버를 제공

프로젝트 뼈대 만들기

작업 확인하기

- 웹 서버를 실행하기 위해 하나의 창에서 작업해도 되지만 runserver 용으로 별도의 cmd창을 열어 사용하는 것이 편리
- (base) C:\MyTest\RestaurantShare>python manage.py runserver 127.0.0.1:8000
- (base) C:\MyTest\RestaurantShare>python manage.py runserver
- (base) C:\MyTest\RestaurantShare>python manage.py runserver 0:8000

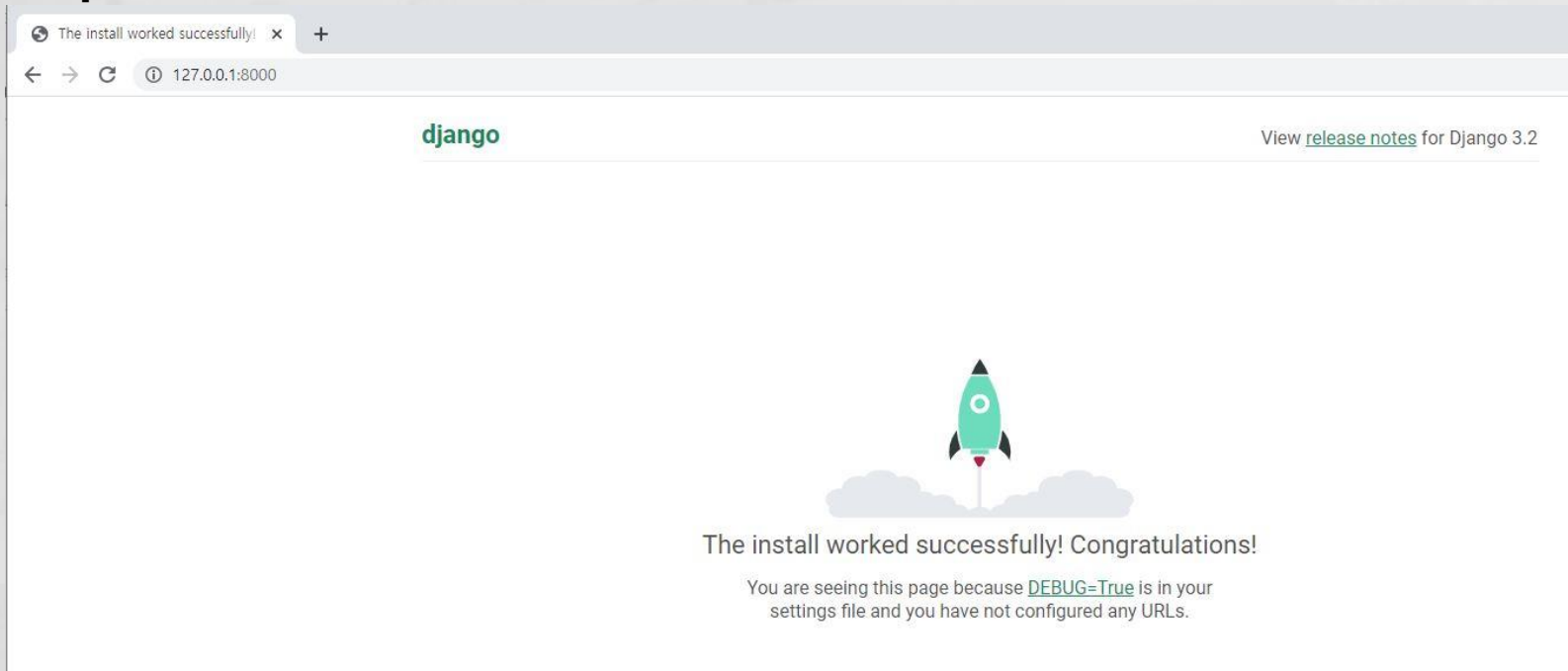
```
(base) C:\MyTest\RestaurantShare>python manage.py runserver 127.0.0.1:8000
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
August 26, 2021 - 19:26:54
Django version 3.2.6, using settings 'RestaurantShare.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

프로젝트 뼈대 만들기

작업 확인하기

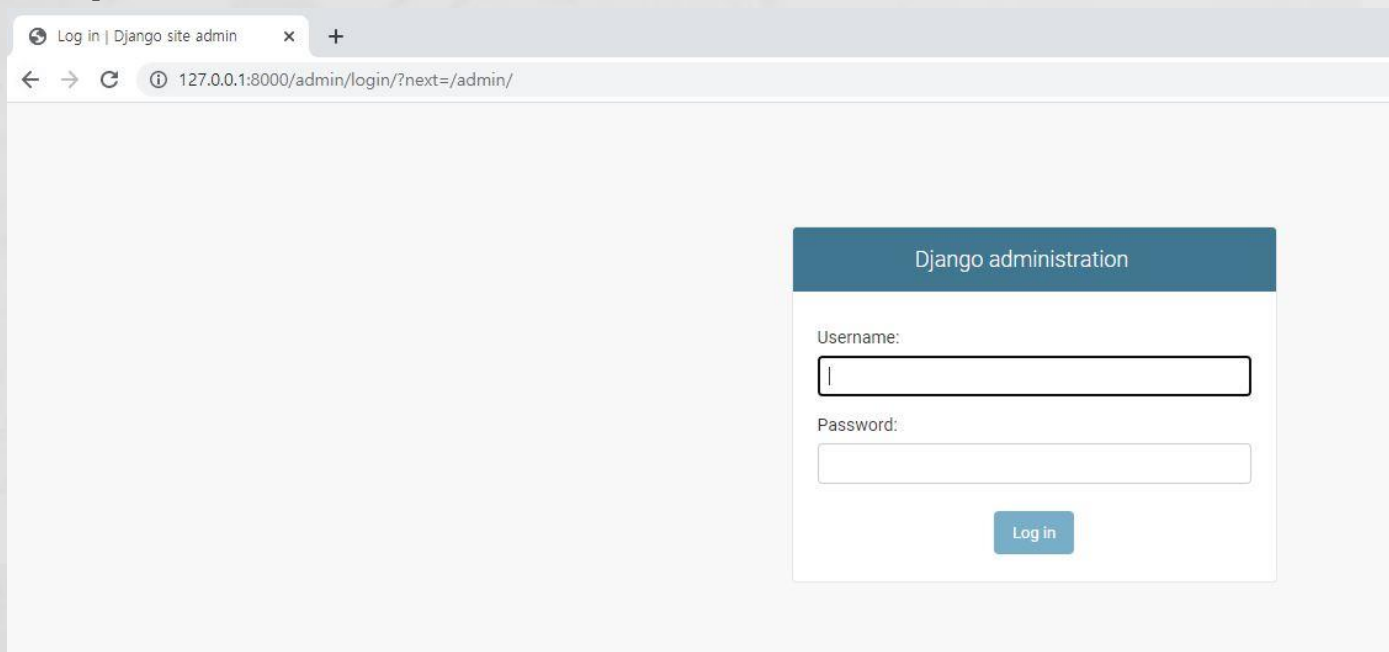
- runserver가 정상적으로 실행되었다면, 웹 브라우저를 열고 IP 주소는 runserver가 동작하는 서버 IP주소를 입력
- `http://127.0.0.1:8000/`



프로젝트 뼈대 만들기

Admin 사이트 접속

- 기본적으로 제공하는 Admin 사이트에 접속해서 테이블이 생성 확인
- 브라우저의 주소창에 IP 주소와 포트번호 동일, URL경로만 /admin 추가
- `http://127.0.0.1:8000/admin`



프로젝트 뼈대 만들기

Admin 사이트 접속

- 로그인하려면 username, password를 넣어야 되는데 아직 생성하지 않았음
- Admin 사이트에 로그인하기 위한 관리자(슈퍼유저)를 생성
- (base) C:\MyTest\RestaurantShare>python manage.py createsuperuser

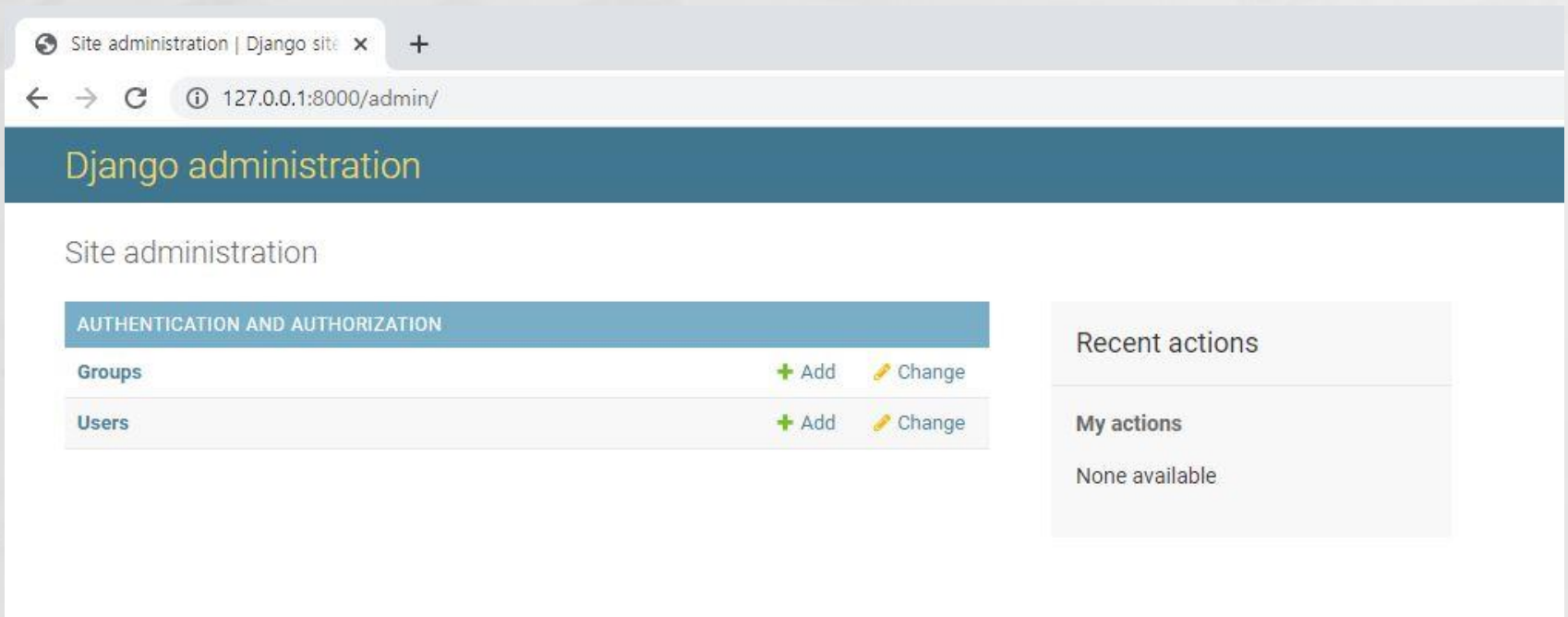
```
(base) C:\MyTest\RestaurantShare>python manage.py createsuperuser
Username (leave blank to use 'bbobbee'): smcpacs
Email address: smcpacs@naver.com
Password:
Password (again):
Superuser created successfully.

(base) C:\MyTest\RestaurantShare>
```


프로젝트 뼈대 만들기

Admin 사이트 접속

- `http://127.0.0.1:8000/admin`
- 로그인 창에서 `username`, `password` 입력



프로젝트 뼈대 만들기

Admin 사이트 접속

◦ 사용자 확인

Select user to change | Django x +

127.0.0.1:8000/admin/auth/user/

Django administration

Home > Authentication and Authorization > Users

AUTHENTICATION AND AUTHORIZATION

- Groups + Add
- Users + Add**

Select user to change

Q [] Search

Action: [] Go 0 of 1 selected

<input type="checkbox"/>	USERNAME	EMAIL ADDRESS	FIRST NAME	LAST NAME	STAFF STATUS
<input type="checkbox"/>	smcpacs	smcpacs@naver.com			✓

1 user

프로젝트 뼈대 만들기

Admin 사이트 접속

- Admin 사이트에서 Users와 Groups 테이블을 포함하여 새롭게 만들 테이블에 대한 데이터의 입력, 변경, 삭제 등의 작업을 할 수 있음
- Admin 화면에서 기본적으로 Users와 Groups 테이블이 보이는 것은 이미 settings.py 파일에 django.contrib.auth 애플리케이션이 등록되어 있기 때문
- 즉 장고에서 기본으로 제공하는 auth 앱에 Users와 Groups 테이블이 미리 정의

프로젝트 뼈대 만들기

골격 생성

- 애플리케이션을 MVT 패턴 방식으로 개발할 수 있도록 골격 생성
- (base) C:\WMyTest>tree /F RestaurantShare

```
C:\WMYTEST\RESTAURANTSHARE
db.sqlite3
manage.py

RestaurantShare
├── asgi.py
├── settings.py
├── urls.py
├── wsgi.py
├── __init__.py
├── __pycache__
│   ├── settings.cpython-38.pyc
│   ├── urls.cpython-38.pyc
│   ├── wsgi.cpython-38.pyc
│   └── __init__.cpython-38.pyc
└── shareRes
    ├── admin.py
    ├── apps.py
    ├── models.py
    ├── tests.py
    ├── views.py
    ├── __init__.py
    ├── migrations
    │   ├── __init__.py
    │   ├── __pycache__
    │   │   └── __init__.cpython-38.pyc
    │   └── __pycache__
    │       ├── admin.cpython-38.pyc
    │       ├── apps.cpython-38.pyc
    │       ├── models.cpython-38.pyc
    │       └── __init__.cpython-38.pyc
    └── sendEmail
        ├── admin.py
        ├── apps.py
        ├── models.py
        ├── tests.py
        ├── views.py
        ├── __init__.py
        ├── migrations
        │   ├── __init__.py
        │   ├── __pycache__
        │   │   └── __init__.cpython-38.pyc
        │   └── __pycache__
        │       ├── admin.cpython-38.pyc
        │       ├── apps.cpython-38.pyc
        │       ├── models.cpython-38.pyc
        │       └── __init__.cpython-38.pyc
```

애플리케이션 개발하기 – Model 코딩

Model 코딩

- 모델 작업은 데이터베이스에 테이블을 생성하는 작업
- >notepad models.py // 테이블을 정의함
- >notepad admins.py // 정의된 테이블이 Admin 화면에 보이게 함
- >python manage.py makemigrations // 데이터베이스에 변경이 필요한 사항을 추출함
- >python manage.py migrate // 데이터베이스에 변경사항을 반영함
- >python manage.py runserver // 현재까지 작업을 개발용 웹 서버로 확인함

애플리케이션 개발하기 – Model 코딩

테이블 정의

- shareRes 애플리케이션은 Category, Restaurant 테이블이 필요
- 테이블은 **models.py** 파일에 정의

테이블(Category) 컬럼명	컬럼 타입
category_name	Varchar(100)
Id	Integer

테이블(Restaurant) 컬럼명	컬럼 타입
restaurant_name	Varchar(100)
restaurant_link	Varchar(500)
restaurant_content	Text
restaurant_keyword	Varchar(50)
category	ForeignKey
id	Integer

애플리케이션 개발하기 – Model 코딩

테이블 정의

- 테이블은 **models.py** 파일에 정의

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

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

```
class Category(models.Model):
```

```
    category_name = models.CharField(max_length = 100)
```

```
class Restaurant(models.Model):
```

```
    category = models.ForeignKey(Category, on_delete=models.SET_DEFAULT, default=3)
```

```
    restaurant_name = models.CharField(max_length = 100) #맛집 이름
```

```
    restaurant_link = models.CharField(max_length = 500) #맛집 URL
```

```
    restaurant_content = models.TextField() # 맛집 설명
```

```
    restaurant_keyword = models.CharField(max_length = 50) #키워드
```

애플리케이션 개발하기 – Model 코딩

Admin 사이트에 테이블 반영

- models.py 파일에서 정의한 테이블도 admin 사이트에 보이도록 등록
- admin.py 파일에 등록

```
from django.contrib import admin

# Register your models here.
from shareRes.models import Category, Restaurant

admin.site.register(Category)
admin.site.register(Restaurant)
```

- models.py 모듈에서 정의한 toDo클래스를 임포트하고 , admin.site.register() 함수를 사용하여 임포트 한 클래스를 Admin 사이트에 등록

애플리케이션 개발하기 – Model 코딩

데이터베이스 변경사항 반영

- 테이블의 신규 생성, 테이블의 정의 변경 등 데이터베이스에 변경이 필요한 사항이 있으면, 이를 데이터베이스에 실제로 반영해주는 작업
- 변경사항을 데이터베이스에 반영
- (base) C:\MyTest\RestaurantShare>python manage.py makemigrations

```
(base) C:\MyTest\RestaurantShare>python manage.py makemigrations
Migrations for 'shareRes':
  shareRes\migrations\0001_initial.py
    - Create model Category
    - Create model Restaurant
```

애플리케이션 개발하기 – Model 코딩

데이터베이스 변경사항 반영

- makemigrations 명령에 의해 shareRes/migrations 디렉토리 하위에 마이그레이션 파일들이 생김

```
C:\MyTest\RestaurantShare\shareRes 디렉토리

2021-08-26 오후 07:23 <DIR> .
2021-08-26 오후 07:23 <DIR> ..
2021-08-26 오후 07:57      182 admin.py
2021-08-26 오후 07:05     154 apps.py
2021-08-26 오후 08:00 <DIR> migrations
2021-08-26 오후 07:50     623 models.py
2021-08-26 오후 07:05      63 tests.py
2021-08-26 오후 07:05      66 views.py
2021-08-26 오후 07:05       0 __init__.py
2021-08-26 오후 07:57 <DIR> __pycache__
                6개 파일                1,088 바이트
                4개 디렉토리 18,627,801,088 바이트 남음
```

애플리케이션 개발하기 – Model 코딩

데이터베이스 변경사항 반영

- 마이그레이션 파일들을 이용해 migrate 명령으로 데이터베이스에 테이블을 생성
- (base) C:\MyTest\RestaurantShare>python manage.py migrate

```
(base) C:\MyTest\RestaurantShare\shareRes>cd ..  
  
(base) C:\MyTest\RestaurantShare>python manage.py migrate  
Operations to perform:  
  Apply all migrations: admin, auth, contenttypes, sessions, shareRes  
Running migrations:  
  Applying shareRes.0001_initial... OK
```

애플리케이션 개발하기 – Model 코딩

작업 확인하기

- 웹 서버를 실행하기 위해 하나의 창에서 작업해도 되지만 runserver 용으로 별도의 cmd창을 열어 사용하는 것이 편리
- (base) C:\MyTest\RestaurantShare>python manage.py runserver 127.0.0.1:8000
- (base) C:\MyTest\RestaurantShare>python manage.py runserver
- (base) C:\MyTest\RestaurantShare>python manage.py runserver 0:8000

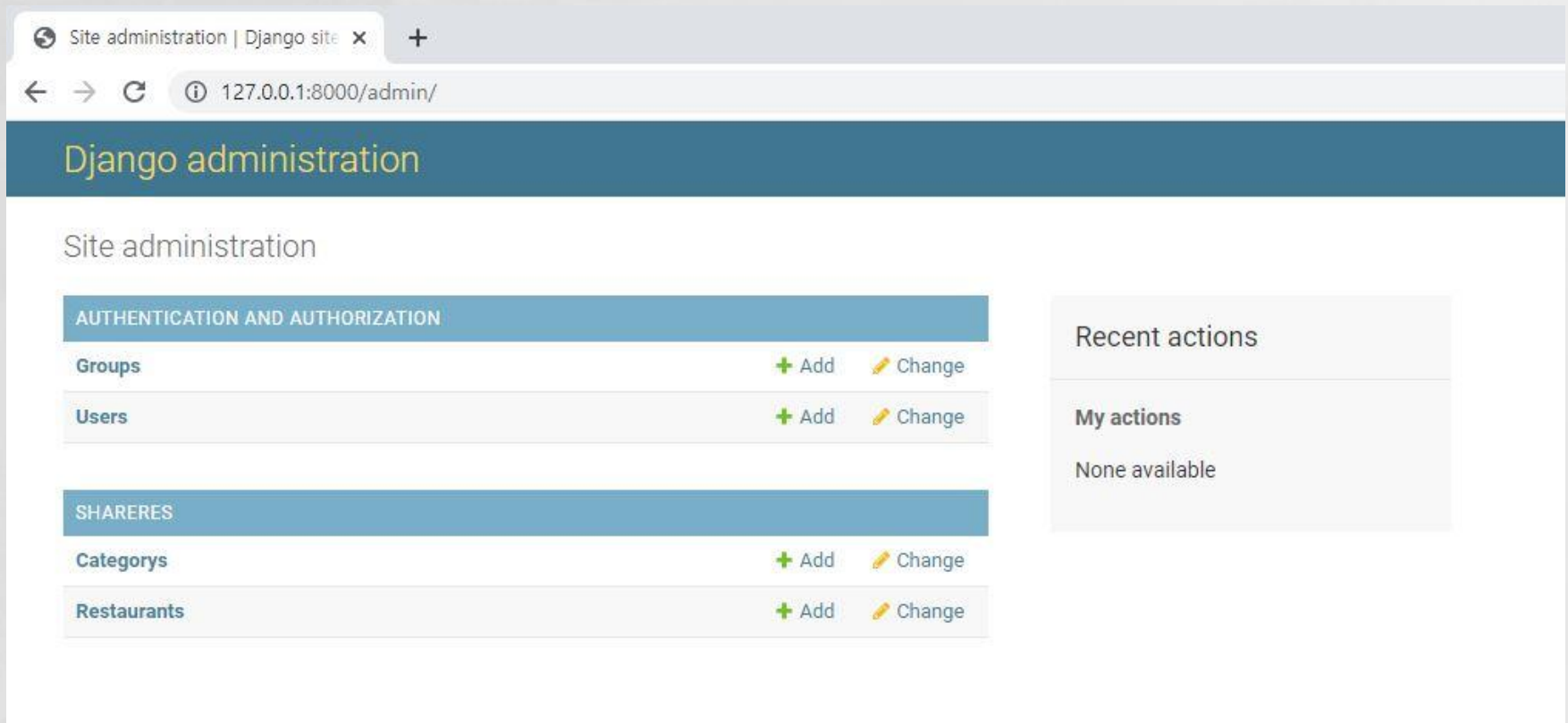
```
(base) C:\MyTest\RestaurantShare>python manage.py runserver 127.0.0.1:8000
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
August 26, 2021 - 19:26:54
Django version 3.2.6, using settings 'RestaurantShare.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

애플리케이션 개발하기 – Model 코딩

작업 확인

- Category, Restaurants 테이블을 만든 모델이 정상적으로 등록



애플리케이션 개발하기 –View 및 Template

URLconf 코딩

- URLconf 설계 따르면, Admin사이트까지 포함해서 URL과 뷰가 필요
- `urls.py(C:\MyTest\RestaurantShare\RestaurantShare)` 파일에 코딩
- `urlpatterns = [~ path('', include('shareRes.urls')),
path('sendEmail/', include('sendEmail.urls')),]`

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

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


애플리케이션 개발하기 -View 및 Template

URLconf 코딩 - shareRes

- urls.py(C:\MyTest\RestaurantShare\shareRes) 파일에 코딩

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

    path('restaurantDetail/delete',views.Delete_restaurant, name='resDelete'),
    path('restaurantDetail/<str:res_id>',views.restaurantDetail, name='resDetailPage'), # http://localhost:8000/restaurantDetail/<str:res_id>
    path('restaurantDetail/updatePage/update',views.Update_restaurant, name='resUpdate'),
    path('restaurantDetail/updatePage/<str:res_id>',views.restaurantUpdate, name='resUpdatePage'),

    path('restaurantCreate/', views.restaurantCreate, name='resCreatePage'), #http://localhost:8000/restaurantCreate/
    path('restaurantCreate/create',views.Create_restaurant,name='resCreate'), ##http://localhost:8000/restaurantCreate/create

    path('categoryCreate/',views.categoryCreate, name='cateCreatePage'),
    path('categoryCreate/create',views.Create_category, name='cateCreate'),
    path('categoryCreate/delete',views.Delete_category, name='cateDelete'),
]
```

애플리케이션 개발하기 -View 및 Template

URLconf 코딩 - shareRes

- urls.py(C:\MyTest\RestaurantShare\sendEmail) 파일에 코딩

```
urlpatterns = [  
    path('send/', views.sendEmail) #http://localhost:8080/seneEmail/send/ 호출시 매칭  
]
```


애플리케이션 개발하기 –View 및 Template

뷰 함수 index() 및 템플릿 작성

- 뷰함수와 템플릿은 서로에게 영향을 미치기 때문에 보통 같이 작업
- 내용을 구현하기 위해 템플릿 파일 index.html

```
<body>
  <div class="container">
    <div class="header">
      <div class="jumbotron">
        <h1>맛집 공유 사이트</h1>
        <p>made by Django</p>
      </div>
    </div>
    <div class="content">
      <div class="row">
        <form action="/sendEmail/send/" method="POST" onsubmit="return emailCheckForm();"> {% csrf_token %}
          <div class="restaurantInfoDiv ">
            <div class="row">
              <div class="infoHeaderDiv">
                <h1>Category <small class="subTitle">상세내용을 보려면 클릭하세요.</small></h1>
              </div>
              <div class="categoryAddDiv">
                <a href="categoryCreate/" class="categoryAddBtn btn btn-info" role="button"> + </a>
              </div>
            </div>
          </div>
        </form>
      </div>
    </div>
  </div>
```

애플리케이션 개발하기 –View 및 Template

뷰 함수 `index()` 및 템플릿 작성

- 뷰 코딩을 위해 `views.py`에 `index()` 함수 정의

```
def index(request):  
    categories = Category.objects.all()  
    restaurants = Restaurant.objects.all()  
    content = {'categories': categories, 'restaurants': restaurants}  
    return render(request, 'shareRes/index.html', content)  
# return HttpResponse("index")  
# return render(request, 'shareRes/index.html')
```

애플리케이션 개발하기 –View 및 Template

뷰 함수 restaurantDetail() 및 템플릿 작성

- 뷰함수와 템플릿은 서로에게 영향을 미치기 때문에 보통 같이 작업
- 내용을 구현하기 위해 템플릿 파일 restaurantDetail.html

```
<body>
  <div class="container">
    <div class="header">
      <div class="jumbotron">
        <h1>맛집 상세보기</h1>
        <p>made by Django</p>
      </div>
    </div>
    <div class="content">
      <div class="inputDiv">
        <div class="input-group">
          <span class="input-group-addon" id="sizing-addon2">카테고리</span>
          <input id="resCategory" name="resCategory" type="text" class="form-control" disabled
            value="{{restaurant.category.category_name}}">
        </div>
        <div class="input-group">
          <span class="input-group-addon" id="sizing-addon2">맛집 이름</span>
```

애플리케이션 개발하기 –View 및 Template

뷰 함수 restaurantDetail() 및 템플릿 작성

- 뷰 코딩을 위해 views.py에 restaurantDetail() 함수 정의

```
def restaurantDetail(request,res_id) :  
    restaurant = Restaurant.objects.get(id = res_id)  
    content = {'restaurant': restaurant}  
    # return HttpResponse("restaurantDetail")  
    return render(request, 'shareRes/restaurantDetail.html', content)
```

애플리케이션 개발하기 –View 및 Template

뷰 함수 restaurantCreate() 및 템플릿 작성

- 뷰함수와 템플릿은 서로에게 영향을 미치기 때문에 보통 같이 작업
- 내용을 구현하기 위해 템플릿 파일 restaurantCreate.html

```
<body>
  <div class="container">
    <div class="header">
      <div class="jumbotron">
        <h1>맛집 추가하기</h1>
        <p>made by Django</p>
      </div>
    </div>
    <div class="content">
      <form action="./create" method="POST" onsubmit="return checkFrom();">{% csrf_token %}
        <div class="inputDiv">
          <div class="input-group">
            <span class="input-group-addon" id="sizing-addon2">카테고리</span>
            <select id="resCategory" name="resCategory" class="resCategory" size="1" required autofocus>
              {% for category in categories %}
                {% if category.id == 3 %}
                  <option value="{{category.id}}" selected>{{category.category_name}}</option>
                {% else %}
                  <option value="{{category.id}}">{{category.category_name}}</option>
                {% endif %}
              {% endfor %}
            </select>
          </div>
        </div>
      </form>
    </div>
  </div>
```


애플리케이션 개발하기 –View 및 Template

뷰 함수 restaurantCreate() 및 템플릿 작성

- 뷰 코딩을 위해 views.py에 restaurantCreate() 함수 정의

```
def restaurantCreate(request) :  
    categories = Category.objects.all()  
    content = {'categories': categories}  
    # return HttpResponse("restaurantCreate")  
    return render(request,'shareRes/restaurantCreate.html',content)
```

애플리케이션 개발하기 –View 및 Template

뷰 함수 restaurantUpdate() 및 템플릿 작성

- 뷰함수와 템플릿은 서로에게 영향을 미치기 때문에 보통 같이 작업
- 내용을 구현하기 위해 템플릿 파일 restaurantUpdate.html

```
<body>
  <div class="container">
    <div class="header">
      <div class="jumbotron">
        <h1>맛집 수정하기</h1>
        <p>made by Django</p>
      </div>
    </div>
    <div class="content">
      <form action="/update" method="POST" onsubmit="return checkFrom();">{% csrf_token %}
        <div class="inputDiv">
          <div class="input-group">
            <span class="input-group-addon" id="sizing-addon2">카테고리</span>
            <select id="resCategory" name="resCategory" class="resCategory" size="1" required autofocus>
              {% for category in categories %}
                {% if category == restaurant.category %}
                  <option value="{{category.id}}" selected>{{category.category_name}}</option>
                {% else %}
                  <option value="{{category.id}}">{{category.category_name}}</option>
                {% endif %}
              {% endfor %}
            </select>
          </div>
        </div>
      </form>
    </div>
  </div>
```

애플리케이션 개발하기 –View 및 Template

뷰 함수 restaurantUpdate() 및 템플릿 작성

- 뷰 코딩을 위해 views.py에 restaurantUpdate() 함수 정의

```
def restaurantUpdate(request, res_id):  
    categories = Category.objects.all()  
    restaurant = Restaurant.objects.get(id = res_id)  
    # print(restaurant)  
    content = {'categories': categories, 'restaurant': restaurant}  
    return render(request, 'shareRes/restaurantUpdate.html', content)
```


애플리케이션 개발하기 –View 및 Template

뷰 함수 categoryCreate() 및 템플릿 작성

- 뷰함수와 템플릿은 서로에게 영향을 미치기 때문에 보통 같이 작업
- 내용을 구현하기 위해 템플릿 파일 categoryCreate.html

```
<body>
  <div class="container">
    <div class="header">
      <div class="jumbotron">
        <h1>카테고리 추가하기</h1>
        <p>made by Django</p>
      </div>
    </div>
    <div class="content">

      <div class="inputDiv">

        {% for category in categories %}
        {% if category.id == 3%}
          <div class="input-group">
            <span class="input-group-addon" style="border:1px solid #ccc; border-radius: 4px;">{{ category.categor
          </div>
        {% endif %}

      </div>
    </div>
  </div>
</body>
```

애플리케이션 개발하기 –View 및 Template

뷰 함수 categoryCreate() 및 템플릿 작성

- 뷰 코딩을 위해 views.py에 categoryCreate() 함수 정의

```
def categoryCreate(request) : #기존 카테고리 출력
    categories = Category.objects.all()
    content = {'categories': categories}
    # return HttpResponse("categoryCreate")
    return render(request, 'shareRes/categoryCreate.html',content)
```

애플리케이션 개발하기 –View 및 Template

뷰 코딩을 위한 기능 작성

- 뷰 코딩을 위해 views.py에 기능 함수 정의

```
def Delete_restaurant(request):  
    res_id = request.POST['resId']  
    restaurant = Restaurant.objects.get(id = res_id)  
    restaurant.delete()  
    return HttpResponseRedirect(reverse('index'))
```

애플리케이션 개발하기 –View 및 Template

뷰 코딩을 위한 기능 작성

- 뷰 코딩을 위해 `views.py`에 기능 함수 정의

```
def Update_restaurant(request):
    print("여기 왔나요?")
    resId = request.POST['resId']
    change_category_id = request.POST['resCategory']
    change_category = Category.objects.get(id = change_category_id)
    change_name = request.POST['resTitle']
    change_link = request.POST['resLink']
    change_content = request.POST['resContent']
    change_keyword = request.POST['resLoc']
    before_restaurant = Restaurant.objects.get(id = resId)
    before_restaurant.category = change_category
    before_restaurant.restaurant_name = change_name
    before_restaurant.restaurant_link = change_link
    before_restaurant.restaurant_content = change_content
    before_restaurant.restaurant_keyword = change_keyword
    before_restaurant.save()
    return HttpResponseRedirect(reverse('resDetailPage', kwargs={'res_id':resId}))
```

애플리케이션 개발하기 –View 및 Template

뷰 코딩을 위한 기능 작성

- 뷰 코딩을 위해 `views.py`에 기능 함수 정의

```
def Create_restaurant(request) :  
    category_id = request.POST['resCategory']  
    category = Category.objects.get(id = category_id)  
    name = request.POST['resTitle']  
    link = request.POST['resLink']  
    content = request.POST['resContent']  
    keyword = request.POST['resLoc']  
    new_res = Restaurant(category = category, restaurant_name = name, restaurant_link = link,  
                        restaurant_content = content, restaurant_keyword = keyword)  
    new_res.save()  
    return HttpResponseRedirect(reverse('index'))
```


애플리케이션 개발하기 –View 및 Template

뷰 코딩을 위한 기능 작성

- 뷰 코딩을 위해 `views.py`에 기능 함수 정의

```
def Create_category(request): # 카테고리 새로 추가하기
    category_name = request.POST['categoryName']
    new_category = Category(category_name = category_name)
    new_category.save()
    return HttpResponseRedirect(reverse('index'))
# return HttpResponse("여기서 category Create 기능을 구현할거야.")
```

애플리케이션 개발하기 –View 및 Template

뷰 코딩을 위한 기능 작성

- 뷰 코딩을 위해 `views.py`에 기능 함수 정의

```
def Delete_category(request):  
    category_id = request.POST['categoryId']  
    delete_category = Category.objects.get(id = category_id)  
    delete_category.delete()  
    return HttpResponseRedirect(reverse('cateCreatePage'))
```

애플리케이션 개발하기 –View 및 Template

작업 확인하기

- 웹 서버를 실행하기 위해 하나의 창에서 작업해도 되지만 runserver 용으로 별도의 cmd창을 열어 사용하는 것이 편리
- (base) C:\MyTest\RestaurantShare>python manage.py runserver
127.0.0.1:8000
- (base) C:\MyTest\RestaurantShare>python manage.py runserver
- (base) C:\MyTest\RestaurantShare>python manage.py runserver 0:8000

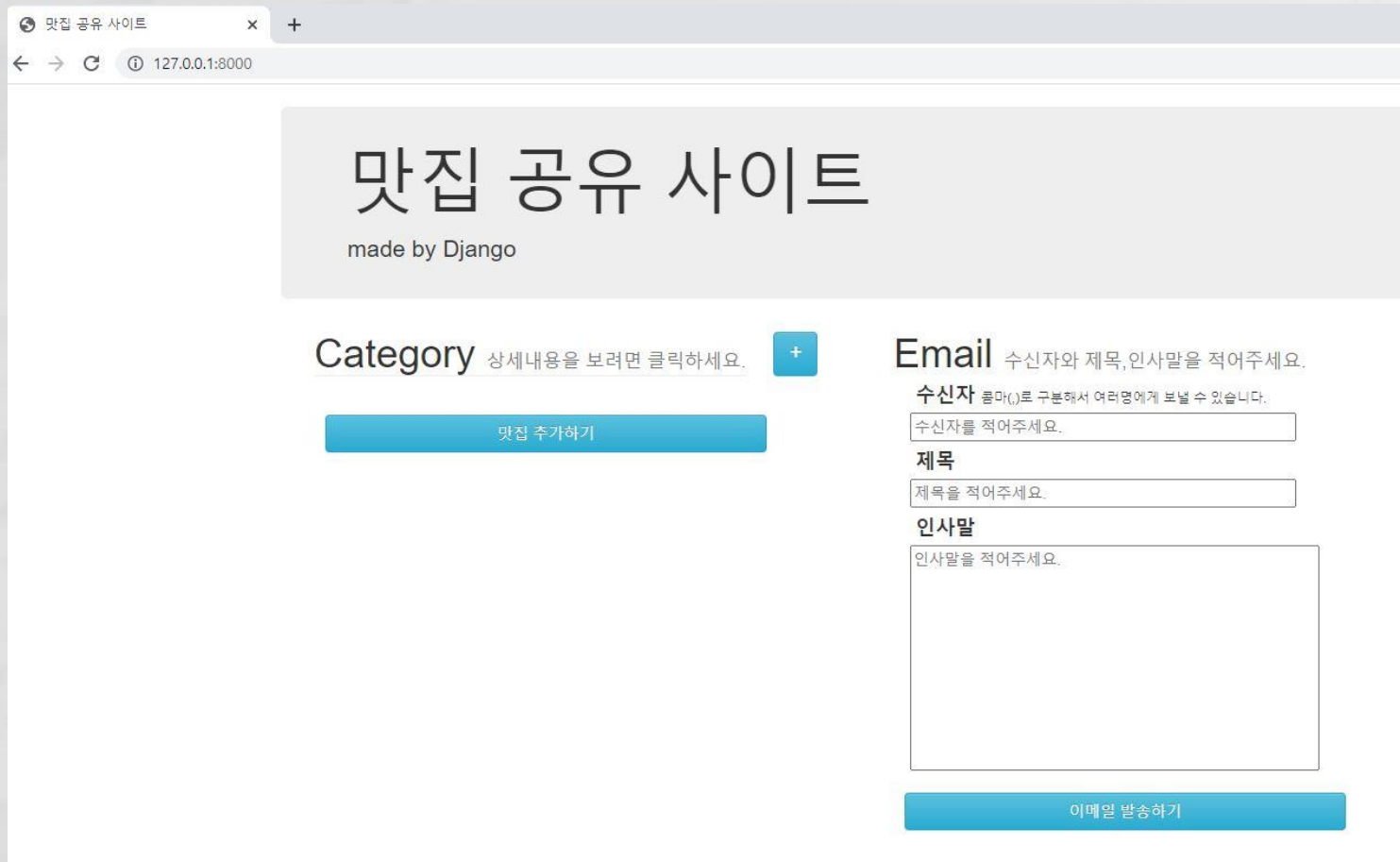
```
(base) C:\MyTest\RestaurantShare>python manage.py runserver 127.0.0.1:8000
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
August 26, 2021 - 19:26:54
Django version 3.2.6, using settings 'RestaurantShare.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```


애플리케이션 개발하기 -View 및 Template

작업 확인하기

- 웹 브라우저에서 `http://127.0.0.1:8000`



맛집 공유 사이트

← → ↻ ⓘ 127.0.0.1:8000

맛집 공유 사이트

made by Django

Category 상세내용을 보려면 클릭하세요. +

맛집 추가하기

Email 수신자와 제목, 인사말을 적어주세요.

수신자 수신자(,)로 구분해서 여러명에게 보낼 수 있습니다.

수신자를 적어주세요.

제목

제목을 적어주세요.

인사말

인사말을 적어주세요.

이메일 발송하기

애플리케이션 개발하기 –View 및 Template

작업 확인하기

- 웹 브라우저에서 `http://127.0.0.1:8000`

맛집 추가하기

made by Django

카테고리

맛집 이름

관련 링크

상세 내용

장소 키워드

애플리케이션 개발하기 -View 및 Template

작업 확인하기

- 웹 브라우저에서 `http://127.0.0.1:8000`

The screenshot shows a web browser window with the address bar displaying `http://127.0.0.1:8000`. The page title is "맛집 공유 사이트" (Favorite Place Sharing Site). The main heading is "맛집 공유 사이트" (Favorite Place Sharing Site) with the subtitle "made by Django".

The interface is divided into two main sections: "Category" and "Email".

Category Section:

- Header: "Category" with a subtext "상세내용을 보려면 클릭하세요." (Click to see details) and a "+" button.
- Buttons: "한식" (Korean Food) and "양식" (Western Food) are highlighted in blue.
- Form: Below "한식", there is a checkbox labeled "채근담" (Chaegeondam). Below "양식", there is a checkbox labeled "알라프리마" (Alafrima).
- Text: "일식" (Japanese Food) and "중식" (Chinese Food) are listed below the checkboxes.
- Button: A blue button labeled "맛집 추가하기" (Add Favorite Place) is at the bottom of the category section.

Email Section:

- Header: "Email" with a subtext "수신자와 제목, 인사말을 적어주세요." (Please enter the recipient, subject, and message).
- Form: There are three input fields: "수신자" (Recipient) with the value "smcpacs@naver.com", "제목" (Subject) with the value "한식 채근담입니다." (Korean Chaegeondam), and "인사말" (Message) with the value "안녕하세요. 한식 채근담 맛집입니다. 좋습니다. 다음에 꼭 오세요." (Hello. This is a Korean Chaegeondam restaurant. It's good. Please come next time).
- Button: A blue button labeled "이메일 발송하기" (Send Email) is at the bottom of the email section.

애플리케이션 개발하기 –View 및 Template

작업 확인하기

◦ 데이터 입력

The screenshot shows the Django administration interface in a web browser. The address bar displays the URL `127.0.0.1:8000/admin/shareRes/category/`. The page title is "Django administration". The breadcrumb navigation shows "Home > Shareres > Categorys".

On the left sidebar, there are two main sections:

- AUTHENTICATION AND AUTHORIZATION**
 - Groups [+ Add](#)
 - Users [+ Add](#)
- SHARERES**
 - Categorys [+ Add](#) (highlighted in yellow)
 - Restaurants [+ Add](#)

The main content area is titled "Select category to change". It features an "Action:" dropdown menu, a "Go" button, and a status "0 of 4 selected". Below this, there is a list of categories with checkboxes:

- ☐ CATEGORY
- ☐ Category object (7)
- ☐ Category object (6)
- ☐ Category object (2)
- ☐ Category object (1)

At the bottom of the list, it says "4 categorys".

애플리케이션 개발하기 -View 및 Template

작업 확인하기

- DB에서 데이터 확인
- (base) C:\MyTest\RestaurantShare>python manage.py dbshell

```
(base) C:\MyTest\RestaurantShare>python manage.py dbshell
SQLite version 3.33.0 2020-08-14 13:23:32
Enter ".help" for usage hints.
sqlite>
sqlite> .tables
auth_group                  django_admin_log
auth_group_permissions      django_content_type
auth_permission             django_migrations
auth_user                   django_session
auth_user_groups            shareRes_category
auth_user_user_permissions  shareRes_restaurant
sqlite>
```

```
sqlite> select * from shareRes_category;
1|한식
2|양식
3|중식
4|일식
5|미식

sqlite> select * from shareRes_restaurant;
1|채근담|https://www.chaageundaam.com/|채근담은 홍자성의 교전 '채근담'의 정신으로 음식을 구현
자연과 함께 유유자적하는 삶의 모습
몸과 정신을 함께 하는 음식에 초점을 맞춘 채식|대치점|1
2|알라프리마|https://allaprima.co.kr/|친구랑 생일이 일주일 차이라 자축하자며 찾아가게된 알라프리마
예약금은 10만원으로 좀 세더라구요 취소는 최소 5일전 가능..
음식을 굳이 해석하면서 먹기보다는 일단 맛있는게 중요하다고 생각하는 사람인데, 일식과 이탈리아의 퓨
변의하면서 갔어요
결론적으로는 돈이 하나도 안아까웠다는...!
아 이재료를 이렇게 요리할수있구나 이게 이런맛을 내네 메뉴 하나하나가 신선했고 그윽다고 그 새로움이
해. 부담이 없었어요
그동안 미술형 스타에 대한 감흥이 별로 없었는데 먹어보니까 와 이래서 별받는구나 싶었던...
다만 향이 강한 음식이 몇 있어서 사람에 따라 호불호가 살짝 있을수도 있겠다 싶었네요
생일이라 부탁드린 레터링도 너무 예뻐고 점책도 편안하니 좋아서 저는 대만족했습니다|논현동|2
sqlite>
```

sqlite> PRAGMA table_info(my_to_do_app_todo);

정리

정리

- 프로젝트 뼈대 만들기
- 애플리케이션 개발하기 - Model 코딩
- 애플리케이션 개발하기 - View 및 Template