**< 애플리케이션 개발하기 - Model 코딩 (Ubuntu) >**

**※ 모델 작업은 데이터베이스에 테이블을 생성하는 작업**

**※ 명령어 실행 순서**

vim models.py // 테이블을 정의함

vim admins.py // 정의된 테이블이 Admin 화면에 보이게 함

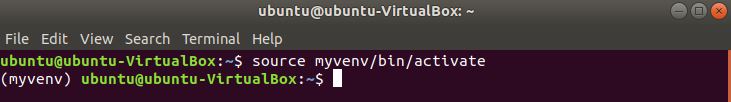
python manage.py makemigrations // 데이터베이스에 변경이 필요한 사항을 추출함

python manage.py migrate // 데이터베이스에 변경사항을 반영함

python manage.py runserver // 현재까지 작업을 개발용 웹 서버로 확인함

**※ 모든 작업은 가상환경(virtualenv)안에서 진행됌**

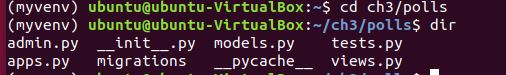
리눅스 --> source myvenv/bin/activate



윈도우 --> myvenv\Scripts\activate

**※ 앞에서(2주차 보고서 참고) 생성한 polls 안에서 작업**

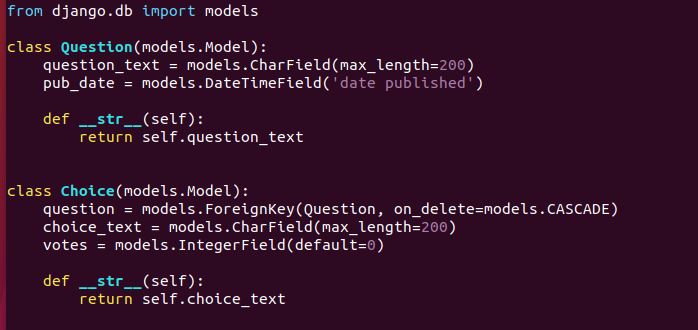
$ cd ch3/polls



**1. 테이블 정의**

polls 애플리케이션은 Question과 Choice 두 개의 테이블이 필요하며 테이블은 models.py 파일에 정의한다.

$ vim models.py



-> 처음 models.py 파일을 열면 '# create your models here.' 이라는 문장이 나오는데, 삭제후 위와 같이 입력하면 된다. (admin.py 파일도 동일하게 + 오타주의)

장고에서는 테이블을 하나의 클래스로 정의하고, 테이블의 컬럼은 클래스의 변수(속성)로 매핑한다. 테이블 클래스는 django.db.models.Model 클래스를 상속받아 정의하고, 각 클래스 변수의 타입도 장고에서 미리 정의된 필드 클래스를 사용한다.

Question 테이블 컬럼과 클래스 변수 간의 매핑

|  |  |  |  |
| --- | --- | --- | --- |
| **테이블 컬럼명** | **컬럼 타입** | **장고의 클래스 변수** | **장고의 필드 클래스** |
| id | integet | (id) | (PK는 장고에서 자동 생성해줌) |
| question\_text | varchar(200) | question\_text | models.CharField(max\_length=200) |
| pub\_date | datetime | pub\_date | models.DateTimeField('date published') |

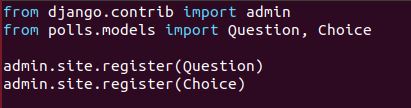
Choice 테이블 컬럼과 클래스 변수 간의 매핑

|  |  |  |  |
| --- | --- | --- | --- |
| **컬럼명** | **타입** | **장고의 클래스 변수** | **장고의 필드 클래스** |
| id | integet | (id) | (PK는 장고에서 자동 생성해줌) |
| choice\_text | varchar(200) | choice\_text | models.CharField(max.length=200) |
| votes | integer | votes | models.IntegerField(default=0) |
| question\_id | integer | question | models.ForeighKey(Question) |

**2. Admin 사이트에 테이블 반영**

Admin 사이트에 접속해보면 장고에서 기본적으로 제공하는 Users, Groups 테이블만 보인다. models.py 파일에서 정의한 테이블도 Admin 사이트에 보이도록 admin.py 파일에 등록해준다.

$ vim admin.py



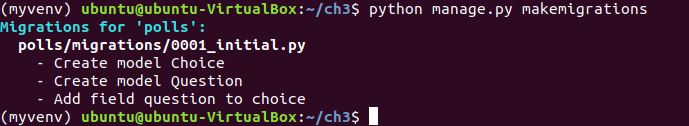
models.py 모듈에서 정의한 Question, Choice 클래스를 임포트하고, admin.site.register() 함수를 사용하여 임포트한 클래스를 Admin 사이트에 등록해주면 된다. 이와 같이 테이블을 새로 만들 때는 models.py와 admin.py 두 개의 파일을 함께 수정해야 한다.

**3. 데이터베이스 변경사항 반영**

테이블의 신규 생성, 테이블의 정의 변경 등 데이터베이스에 변경이 필요한 사항이 있으면, 이를 데이터베이스에 반영해주는 작업을 해야 한다.

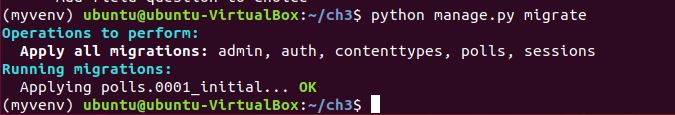
(ch3 디렉토리에서 작업)

$ python manage.py makemigrations



-> migrations은 테이블 및 필드의 생성, 삭제, 변경 등과 같이 데이터베이스에 대한 변경사항을 알려주는 정보다.

$ python manage.py migrate



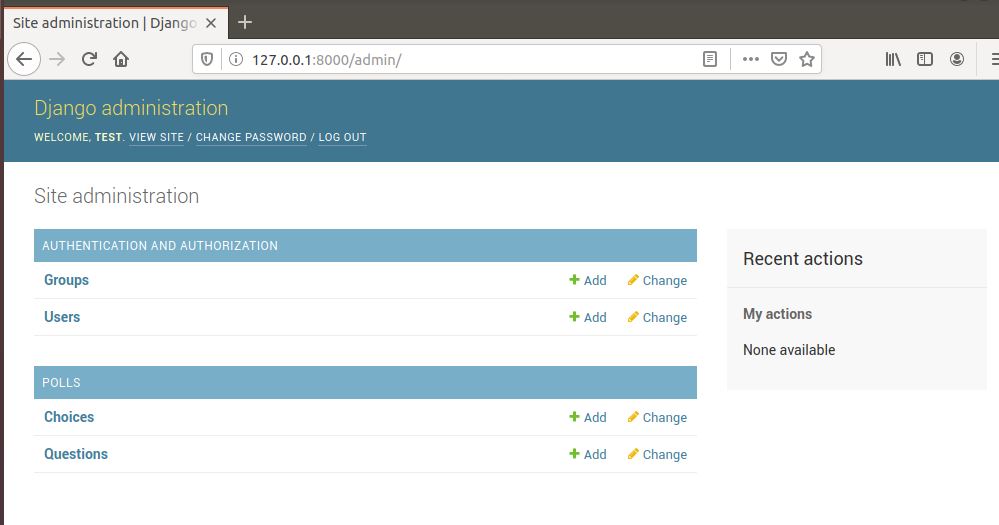
**4. 지금까지 작업 확인하기**

runserver가 실행되어 있지 않다면 먼저 실행한후 웹 브라우저 접속.

$ python manage.py runsever 0.0.0.0:8000



지금까지 작업 확인 -> http://127.0.0.1:8000/admin



-> 앞에 내용이 정상적으로 등록되어 있으면 Users, Groups 이외에 추가한 Question, Choice 테이블을 확인할 수 있다.