

## ■ 프로젝트 구조 생성

### ● mysite

```
Git 명령 프롬프트

C:\Wdev\Wdjango>mkdir mysite

C:\Wdev\Wdjango>cd mysite

C:\Wdev\Wdjango\mysite>django-admin startproject config .

C:\Wdev\Wdjango\mysite>
```

### ● 생성 결과

```
---mysite
|
|   manage.py
|
W---config
    asgi.py
    settings.py
    urls.py
    wsgi.py
    __init__.py
```

## ■ App 생성

### ● polls

C:\> 명령 프롬프트

```
C:\dev\django\mysite>django-admin startapp polls
```

```
C:\dev\django\mysite>
```

### ● 생성 결과

```
mysite
├── manage.py
├── config
│   ├── asgi.py
│   ├── settings.py
│   ├── urls.py
│   ├── wsgi.py
│   └── __init__.py
└── W─ polls
    ├── admin.py
    ├── apps.py
    ├── models.py
    ├── tests.py
    ├── views.py
    ├── __init__.py
    └── W─ migrations
        └── __init__.py
```

## ■ 프로젝트 구동 확인

### ● View

- polls/views.py

```
from django.http import HttpResponse

def index(request):
    return HttpResponse('Hello, world.')
```

### ● Routing

- config/urls.py

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

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

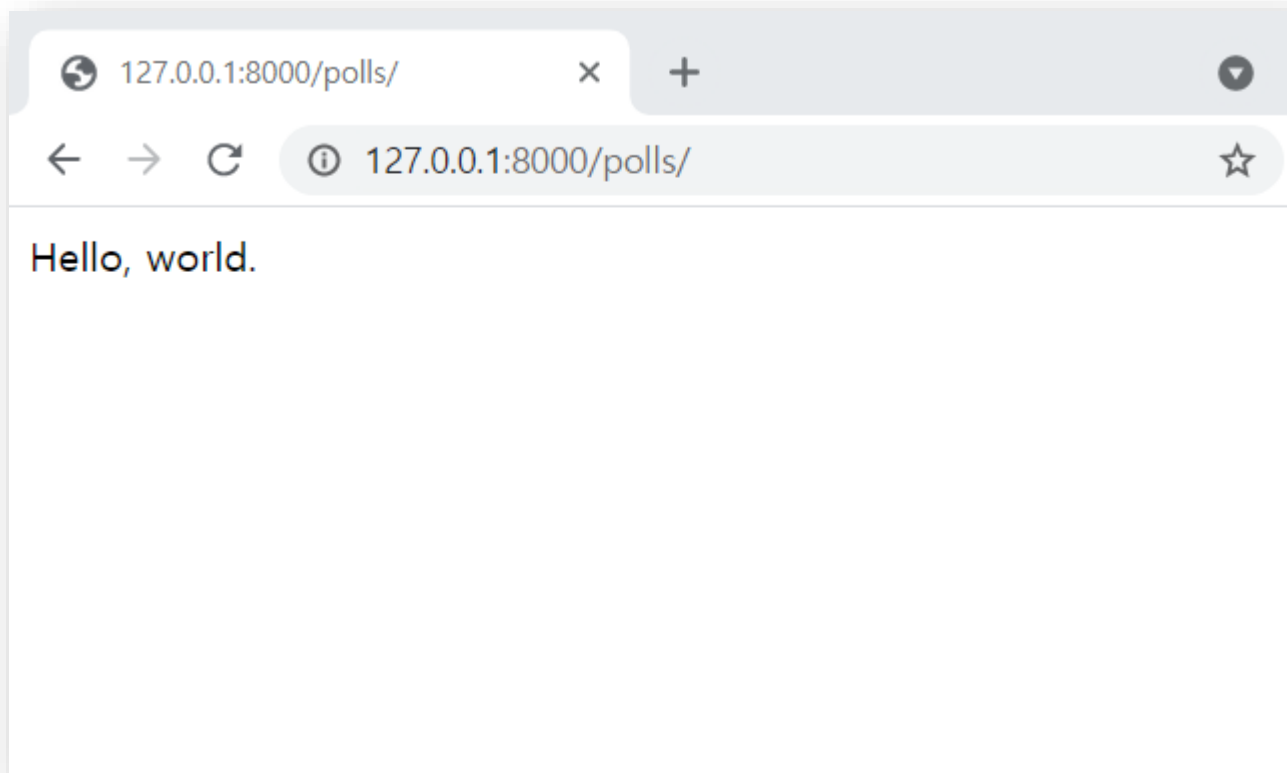
- polls/urls.py

```
from django.urls import path, include
from . import views

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

## ■ 프로젝트 구동 확인

- `python manage.py runserver`
- <http://127.0.0.1:8000/polls/>



## ■ 데이터 입력을 위한 준비

1. 앱 등록 - settings.py
2. 모델 작성 - models.py
3. 모델 적용 - makemigrations / migrate
4. 장고 웹 이용 - 데이터 입력 / 조회 / 수정 / 삭제

## ■ 앱 등록

### ● settings.py

```
INSTALLED_APPS = [  
    'polls',  
    'django.contrib.admin',  
    'django.contrib.auth',  
    'django.contrib.contenttypes',  
    'django.contrib.sessions',  
    'django.contrib.messages',  
    'django.contrib.staticfiles',  
]
```

## ■ 모델 작성

### ● models.py

```
from django.db import models

class Question(models.Model):
    subject = models.CharField(max_length=200)
    content = models.TextField()
    pub_date = models.DateTimeField()

class Answer(models.Model):
    question = models.ForeignKey(Question, on_delete=models.CASCADE)
    content = models.TextField()
    create_date = models.DateTimeField()
```

## ■ 모델 적용

### ● python manage.py makemigrations polls

```
명령 프롬프트

C:\dev\django\mysite>python manage.py makemigrations polls
Migrations for 'polls':
  polls\migrations\0001_initial.py
    - Create model Question
    - Create model Answer

C:\dev\django\mysite>
```

### ● python manage.py migrate

```
명령 프롬프트

C:\dev\django\mysite>python manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, polls, sessions
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying auth.0001_initial... 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 polls.0001_initial... OK
  Applying sessions.0001_initial... OK
```

테이블 (13)

- > 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
- > polls\_answer
- > polls\_question
- > sqlite\_sequence

polls\_answer

- id
- content
- create\_date
- question\_id

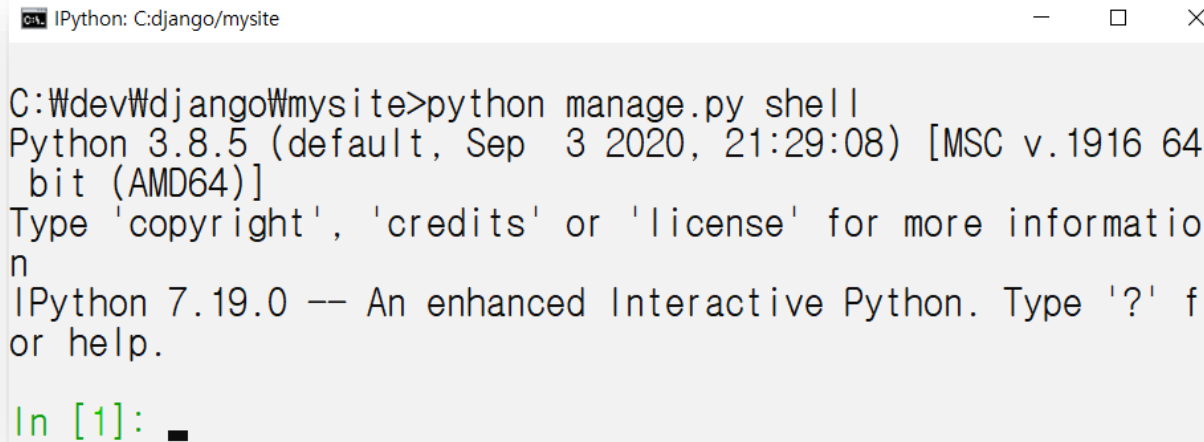
polls\_question

- id
- subject
- content
- pub\_date



## ■ 장고 쉘 이용

### ● python manage.py shell - 쉘 실행



```
IPython: C:\django\mysite

C:\Wdev\Wdjango\Wmysite>python manage.py shell
Python 3.8.5 (default, Sep 3 2020, 21:29:08) [MSC v.1916 64
bit (AMD64)]
Type 'copyright', 'credits' or 'license' for more informatio
n
IPython 7.19.0 -- An enhanced Interactive Python. Type '?' f
or help.

In [1]: _
```

### ● 모델 클래스 가져오기 / 조회 기능 테스트



```
IPython: C:\django\mysite

In [1]: from polls.models import Question, Answer

In [2]: Question.objects.all()
Out[2]: <QuerySet []>
```

```
from polls.models import Question, Answer

Question.objects.all()
```

## 장고 쉘 이용

### ● 모델 객체 생성 / 데이터 입력 및 저장

```
IPython: C:\django\mysite
In [5]: q = Question(subject='프로그래밍', content='컴파일
...: 언어가 아닌 것은?', pub_date=timezone.now())

In [6]: q.save()
```

[save 메소드 사용]  
① 객체 생성 (데이터 입력)  
② 데이터 저장

```
from django.utils import timezone

q = Question(subject='프로그래밍',
              content='컴파일 언어가 아닌 것은?', pub_date=timezone.now())
q.save()
```

```
IPython: C:\django\mysite
In [8]: Question.objects.create(
...:     subject='하드웨어', content='컴퓨터 장치가 아닌
...:     것은?',
...:     pub_date=timezone.now())
Out[8]: <Question: Question object (2)>
```

[매니저 속성 사용]  
① 객체 생성 (데이터 입력)  
+ 데이터 저장

```
Question.objects.create(
    subject='하드웨어', content='컴퓨터 장치가 아닌 것은?',
    pub_date=timezone.now())
```

	id	subject	content	pub_date
	...	필터	필터	필터
1	1	프로그래밍	컴파일 언어가 아닌 것은?	2021-06-02 01:12:29.135702
2	2	하드웨어	컴퓨터 장치가 아닌 것은?	2021-06-02 01:17:44.124509

## 장고 웹 이용

### 데이터 전체 조회

```
IPython: C:\django\mysite
In [10]: q = Question.objects.all()

In [11]: q
Out[11]: <QuerySet [<Question: Question object (1)>, <Question: Question object (2)>]>

In [12]: q[0].subject
Out[12]: '프로그래밍'
```

```
q = Question.objects.all()

q

q[0].subject
```

※ 모델 데이터 조회 결과 변경 → **\_\_str\_\_ 메소드 오버라이딩**

```
class Question(models.Model):
    subject = models.CharField(max_length=200)
    content = models.TextField()
    pub_date = models.DateTimeField()

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

```
IPython: C:\django\mysite
In [1]: from polls.models import Question, Answer

In [2]: Question.objects.all()
Out[2]: <QuerySet [<Question: 프로그래밍>, <Question: 하드웨어>]>
```

## 장고 웹 이용

### 데이터 조회 (1개)

```
IPython: C:\django\mysite
In [4]: Question.objects.get(id=1)
Out[4]: <Question: 프로그래밍>

In [5]: Question.objects.get(pk=1)
Out[5]: <Question: 프로그래밍>
```

`Question.objects.get(id=1)`

`Question.objects.get(pk=1)`

※ `get()` 메소드는 조건에 맞는 데이터가 없을 경우 오류 발생

```
IPython: C:\django\mysite
In [6]: Question.objects.get(id=3)

DoesNotExist                                Traceback (most recent call last)
<ipython-input-6-e5a0cb5352d9> in <module>
----> 1 Question.objects.get(id=3)

434         if not num:
--> 435             raise self.model.DoesNotExist(
436                 "%s matching query does not exist."
%
437                 self.model._meta.object_name

DoesNotExist: Question matching query does not exist.
```

## ■ 장고 웹 이용

### ● 데이터 조회 (N개)

```
IPython: C:\django\mysite
In [7]: Question.objects.filter(id=1)
Out[7]: <QuerySet [<Question: 프로그래밍>]>

In [8]: Question.objects.filter(pk=1)
Out[8]: <QuerySet [<Question: 프로그래밍>]>
```

`Question.objects.filter(id=1)`

`Question.objects.filter(pk=1)`

※ filter() 메소드는 조건에 맞는 데이터가 없을 경우 빈 QuerySet 반환

```
IPython: C:\django\mysite
In [9]: Question.objects.filter(id=3)
Out[9]: <QuerySet []>
```

<https://docs.djangoproject.com/ko/3.2/topics/db/queries/>

## 장고 웹 이용

- 데이터 수정 : 데이터 조회 후 속성의 값을 변경한 후 저장
  - 수정용 데이터 입력

```
IPython: C:\django\mysite
In [20]: Question.objects.create(
...:     subject='테스트제목', content='테스트내용',
...:     pub_date=timezone.now())
Out[20]: <Question: 테스트제목>
```

```
Question.objects.create(
    subject='테스트제목', content='테스트내용',
    pub_date=timezone.now())
```

	id	subject	content	pub_date
	...	필터	필터	필터
1	1	프로그래밍	컴파일 언어가 아닌 것은?	2021-06-02 01:12:29.135702
2	2	하드웨어	컴퓨터 장치가 아닌 것은?	2021-06-02 01:17:44.124509
3	3	테스트제목	테스트내용	2021-06-02 04:05:52.843590

## 장고 웹 이용

- 데이터 수정 : 데이터 조회 후 속성의 값을 변경한 후 저장
  - 수정용 데이터 조회 후 값을 변경하여 저장

IPython: C:\django\mysite

```
In [35]: q = Question.objects.get(subject='테스트제목')
```

```
In [36]: q.subject = '제목수정'
```

```
In [37]: q.save()
```

```
q = Question.objects.get(subject='테스트제목')
q.subject = '제목수정'
q.save()
```

	id	subject	content	pub_date
	...	필터	필터	필터
1	1	프로그래밍	컴파일 언어가 아닌 것은?	2021-06-02 01:12:29.135702
2	2	하드웨어	컴퓨터 장치가 아닌 것은?	2021-06-02 01:17:44.124509
3	3	제목수정	테스트내용	2021-06-02 04:05:52.843590

## 장고 웹 이용

### ● 데이터 삭제 : 데이터 조회 후 삭제

- 삭제용 데이터 조회 후 삭제

```
IPython: C:\django\mysite
In [38]: q = Question.objects.get(id=3)

In [39]: q.delete()
Out[39]: (1, {'polls.Question': 1})
```

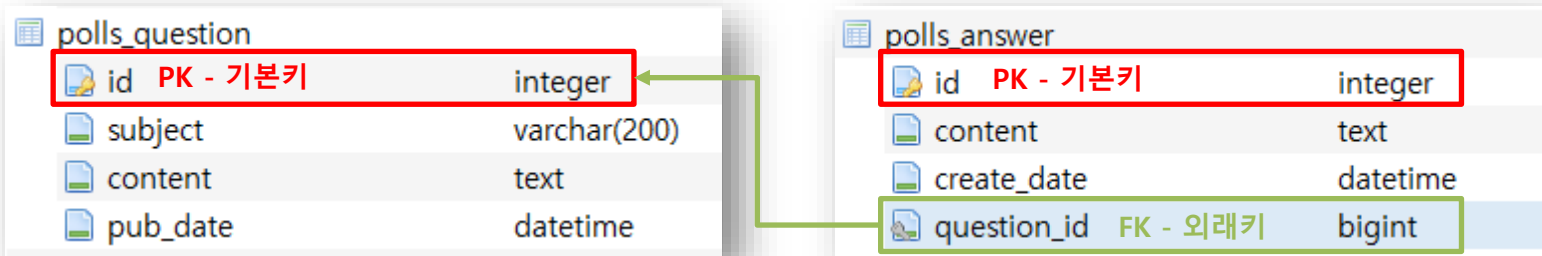
```
q = Question.objects.get(id=3)
q.delete()
```

	id	subject	content	pub_date
	...	필터	필터	필터
1	1	프로그래밍	컴파일 언어가 아닌 것은?	2021-06-02 01:12:29.135702
2	2	하드웨어	컴퓨터 장치가 아닌 것은?	2021-06-02 01:17:44.124509



## 장고 웹 이용

### ● Question - Answer 관계



- Answer 모델의 Foreign Key로 Question이 연결되어 있으므로

**Answer 모델의 데이터를 입력할 때는 Question 모델 데이터 필요**

```
class Question(models.Model):
    subject = models.CharField(max_length=200)
    content = models.TextField()
    pub_date = models.DateTimeField()

class Answer(models.Model):
    question = models.ForeignKey(Question, on_delete=models.CASCADE)
    content = models.TextField()
    create_date = models.DateTimeField()
```

## 장고 웹 이용

### Answer 모델 데이터 입력

- Question 모델 데이터 생성 또는 조회 후 Answer 모델 데이터 입력

```
IPython: C:\django\mysite
In [7]: q=Question.objects.get(id=1)
In [8]: a=Answer(question=q,content='C',create_date=timezone.now())
In [9]: a.save()
```

```
IPython: C:\django\mysite
In [10]: Answer.objects.create(question=q,content='C#',create_date=
...:  imezone.now())
Out[10]: <Answer: Answer object (2)>

In [11]: Answer.objects.create(question=q,content='Python',create_da
...:  te=timezone.now())
...: Answer.objects.create(question=q,content='Kotlin',create_da
...:  te=timezone.now())
Out[11]: <Answer: Answer object (4)>
```

	id	content	create_date	question_id
...	필터	필터	필터	
1	1	C	2021-06-1...	1
2	2	C#	2021-06-1...	1
3	3	Python	2021-06-1...	1
4	4	Kotlin	2021-06-1...	1

```
q = Question.objects.get(id=1)
a = Answer(question=q, content='C', create_date=timezone.now())
a.save()
```

```
Answer.objects.create(question=q, content='C#', create_date=timezone.now())
Answer.objects.create(question=q, content='Python', create_date=timezone.now())
Answer.objects.create(question=q, content='Kotlin', create_date=timezone.now())
```

## ■ 장고 쉘 이용

### ● Answer 모델 데이터 조회

- id가 1번인 Answer 모델 데이터 조회

```
IPython: C:\django\mysite
In [57]: a = Answer.objects.get(id=1)
In [58]: a
Out[58]: <Answer: Answer object (1)>
```

```
a = Answer.objects.get(id=1)
a
```

- Answer 모델의 question 으로 Question 모델 데이터 조회

```
IPython: C:\django\mysite
In [4]: a.question
Out[4]: <Question: 프로그래밍>
```

```
a.question
```

➔ 답변에 해당하는 질문 조회 (1개)

## ■ 장고 쉘 이용

### ● Question 모델 데이터 조회

- id가 1번인 Question 모델 데이터 조회

```
IPython: C:\django\mysite
In [5]: q = Question.objects.get(id=1)

In [6]: q
Out[6]: <Question: 프로그래밍>
```

```
q = Question.objects.get(id=1)

q
```

- 연결모델명\_set 으로 Answer 모델 데이터 조회

```
IPython: C:\django\mysite
In [7]: a_list = q.answer_set.all()

In [8]: a_list
Out[8]: <QuerySet [<Answer: Answer object (1)>, <Answer: Answer object (2)>, <Answer: Answer object (3)>, <Answer: Answer object (4)>]>
```

```
a_list = q.answer_set.all()

a_list
```

➔ 질문에 등록되어 있는 답변 조회 (0 ~ N개)

## ■ 장고 쉘 이용

### ● Question / Answer 모델 데이터 조회

- 내용 중 "언어" 라는 단어가 포함되어 있는 질문 조회

```
In [9]: Question.objects.filter(content__contains='언어')  
Out[9]: <QuerySet [<Question: 프로그래밍>]>
```

- id가 1인 질문에 등록된 답변 중 "C" 가 포함되어 있는 답변 조회

```
In [9]: Question.objects.filter(content__contains='언어')  
Out[9]: <QuerySet [<Question: 프로그래밍>]>  
  
In [10]: q = Question.objects.get(id=1)  
  
In [11]: q.answer_set.filter(content__contains='C')  
Out[11]: <QuerySet [<Answer: Answer object (1)>, <Answer: Answer object (2)>]>
```

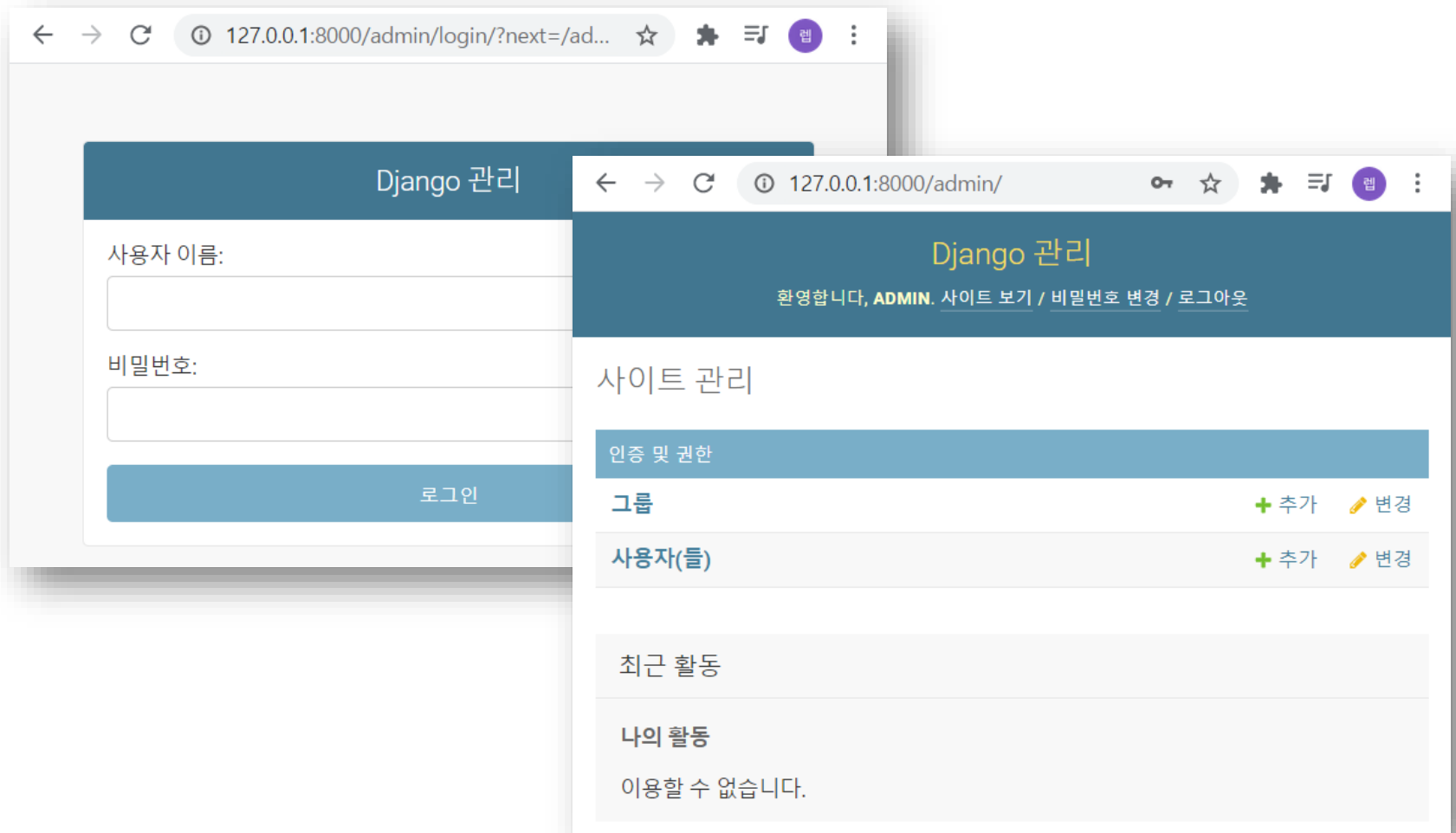
- 내용이 "Python" 인 답변의 질문 제목 조회

```
In [12]: a = Answer.objects.get(content='Python')  
  
In [13]: a.question.subject  
Out[13]: '프로그래밍'
```

## ■ 장고 Admin

### ● 관리자 사이트

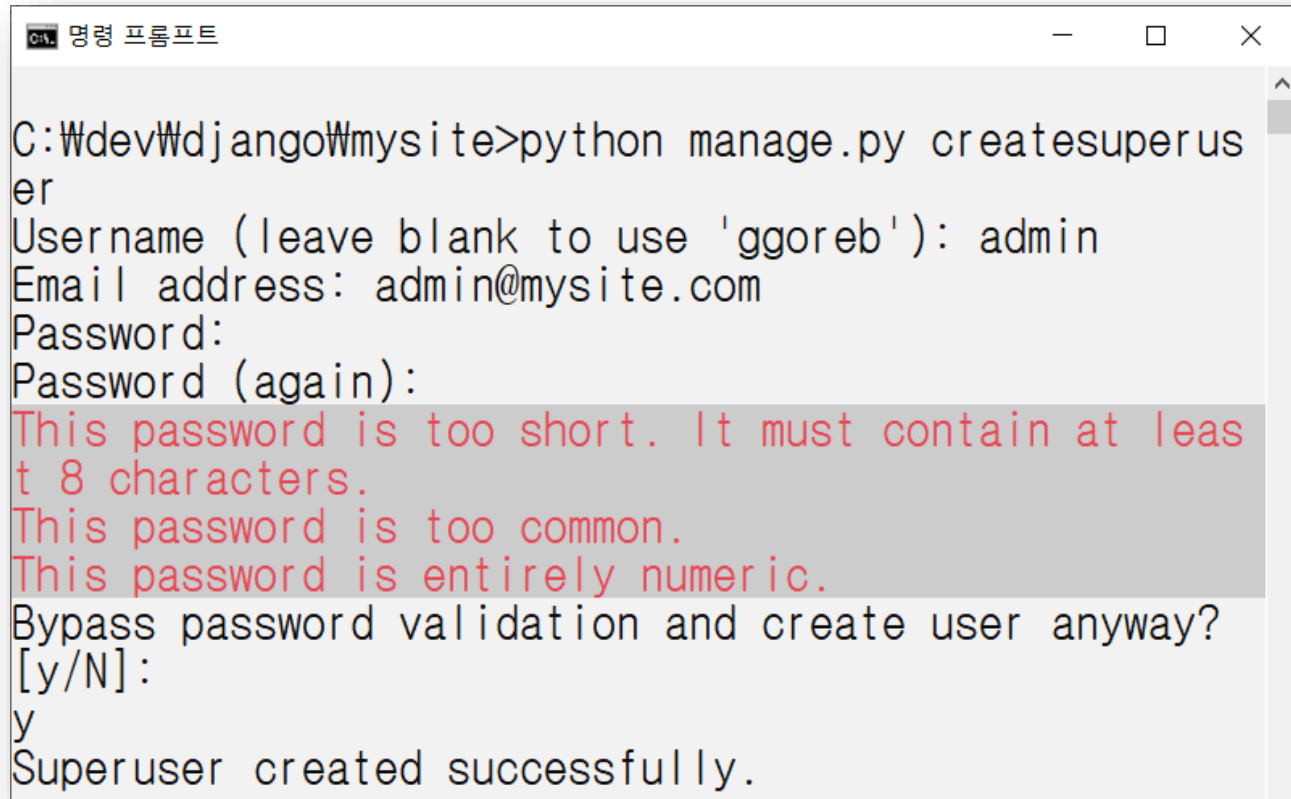
- 프로젝트를 생성하면 자동으로 관리자 사이트가 같이 생성됨
- 관리자 계정을 만들면 장고가 제공해주는 다양한 기능을 활용 할 수 있음



## ■ 장고 Admin

### ● 관리자 계정 생성하기

- `python manage.py createsuperuser`

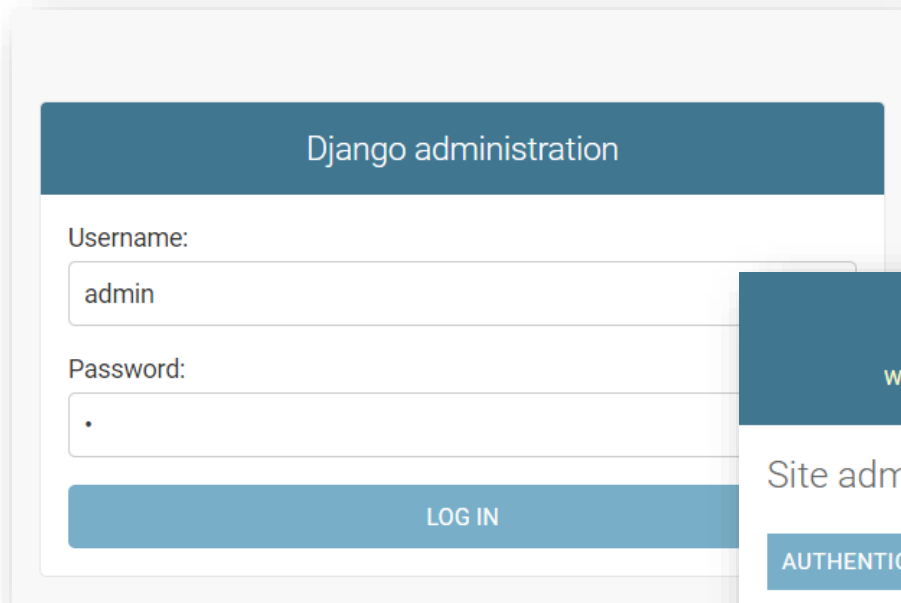


```
C:\dev\django\mysite>python manage.py createsuperuser
Username (leave blank to use 'ggoreb'): admin
Email address: admin@mysite.com
Password:
Password (again):
This password is too short. It must contain at least 8 characters.
This password is too common.
This password is entirely numeric.
Bypass password validation and create user anyway?
[y/N]:
y
Superuser created successfully.
```

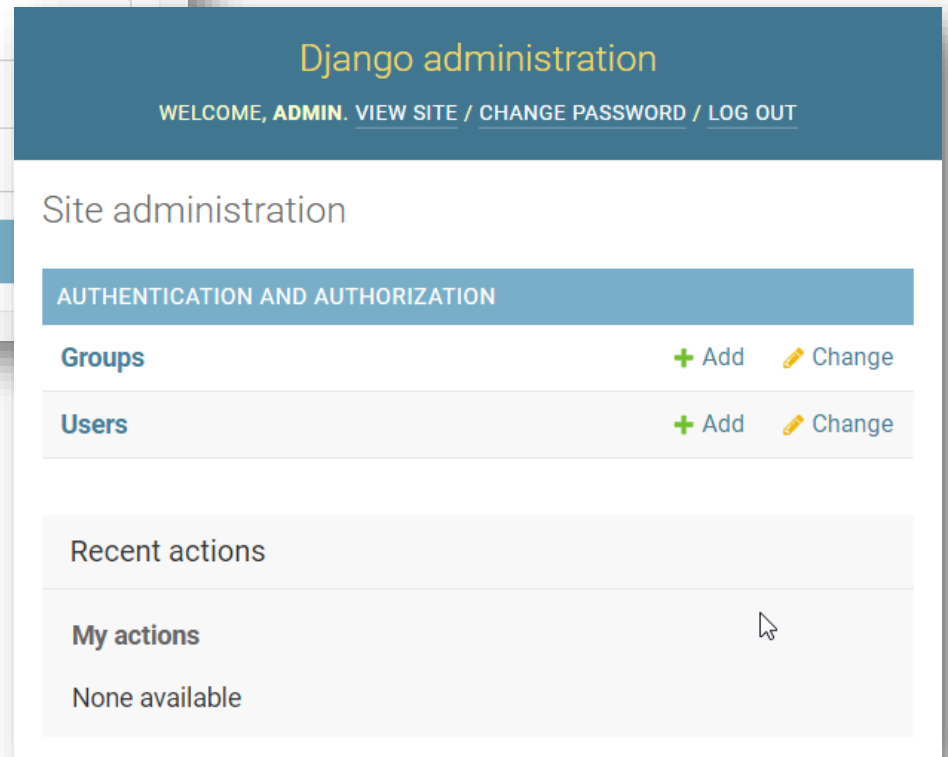
## ■ 장고 Admin

### ● 관리자 사이트 접속하기

- <http://127.0.0.1:8000/admin/> (또는 <http://localhost:8000/admin/>)



The image shows the Django administration login page. It has a dark blue header with the text "Django administration". Below the header, there are two input fields: "Username:" with the value "admin" and "Password:" with a single dot. At the bottom, there is a blue button labeled "LOG IN".



The image shows the Django administration dashboard after a successful login. The header is dark blue with the text "Django administration" in yellow. Below the header, there is a navigation bar with the text "WELCOME, ADMIN. VIEW SITE / CHANGE PASSWORD / LOG OUT". The main content area is white and contains the following sections:

- Site administration**
  - AUTHENTICATION AND AUTHORIZATION**
    - Groups** with links "+ Add" and "Change"
    - Users** with links "+ Add" and "Change"
  - Recent actions**
  - My actions** with a mouse cursor icon
  - None available**

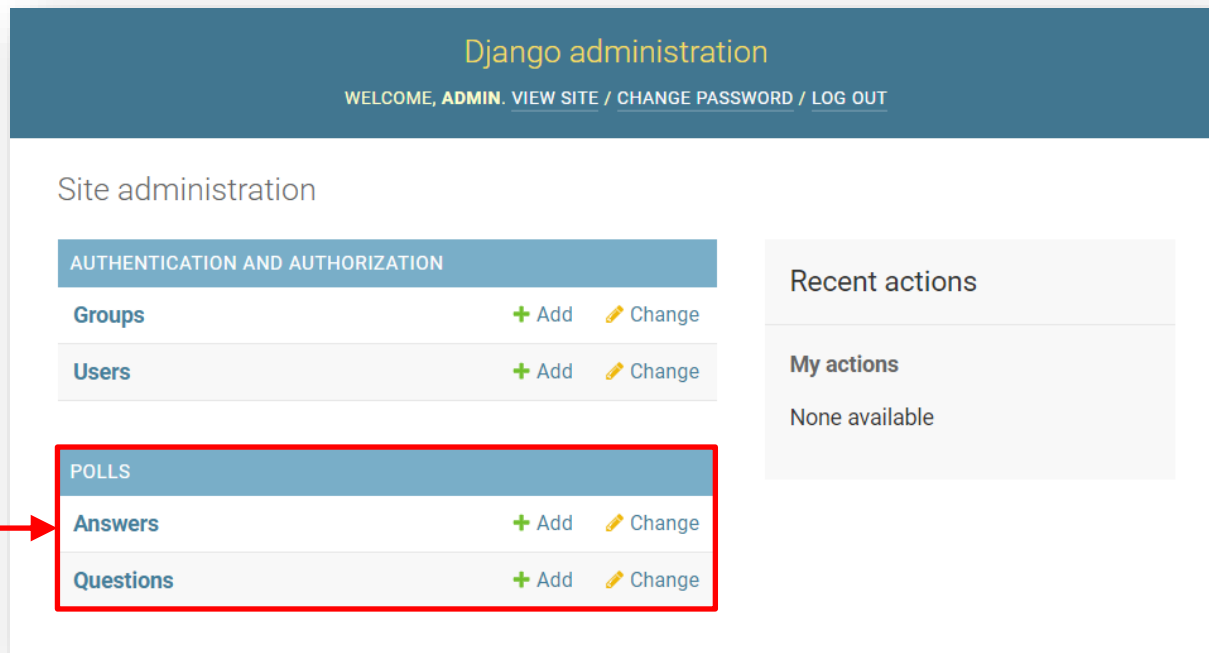


## ■ 장고 Admin

- 관리자 사이트에서 데이터를 제어 할 수 있도록 등록
  - polls/admin.py

```
from django.contrib import admin
from .models import Question, Answer
```

```
admin.site.register(Question)
admin.site.register(Answer)
```



## 장고 Admin

### ● 검색 기능 추가하기

- polls/admin.py

```
from django.contrib import admin
from .models import Question, Answer

class QuestionAdmin(admin.ModelAdmin):
    search_fields = ['subject']

admin.site.register(Question, QuestionAdmin)
admin.site.register(Answer)
```

Home > Polls > Questions

Select question to change

ADD QUESTION +

Action:  Go

0 of 2 selected

<input type="checkbox"/>	QUESTION
<input type="checkbox"/>	하드웨어
<input type="checkbox"/>	프로그래밍

2 questions

Home > Polls > Questions

Select question to change

ADD QUESTION +

Search

Action:  Go

0 of 2 selected

<input type="checkbox"/>	QUESTION
<input type="checkbox"/>	하드웨어
<input type="checkbox"/>	프로그래밍

2 questions

## ■ 질문 목록 조회 구현하기

- Question 모델 데이터를 작성한 날짜의 역순으로 조회
  - polls/views.py

```
from .models import Question

def index(request):
    question_list = Question.objects.order_by('-pub_date')
    result = [ q.subject for q in question_list ]
    return HttpResponse( str(result) )
```

← → ↻ ⓘ 127.0.0.1:8000/polls/ ☆

['하드웨어', '프로그래밍']

## ■ 질문 목록 조회 구현하기

### ● render로 화면 출력하기 : 템플릿 사용

- mysite/templates 디렉토리 생성 및 settings.py 설정

> config

> polls

> templates

≡ db.sqlite3

🔗 manage.py

```
TEMPLATES = [  
    {  
        'BACKEND': 'django.template.backends.django.DjangoTemplates',  
        'DIRS': [ BASE_DIR / 'templates' ],  
        'APP_DIRS': True,  
        'OPTIONS': {  
            ....  
        },  
    },  
]
```

## ■ 질문 목록 조회 구현하기

### ● render로 화면 출력하기 : 템플릿 사용

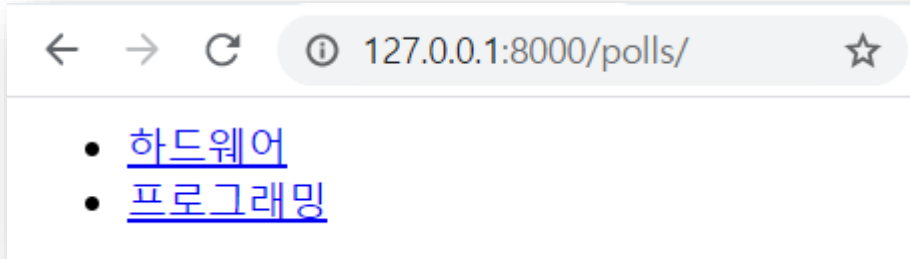
- mysite/templates/polls/question\_list.html

```
{% if question_list %}
    <ul>
        {% for question in question_list %}
            <li>
                <a href="/polls/{{ question.id }}/">
                    {{ question.subject }}
                </a>
            </li>
        {% endfor %}
    </ul>
{% else %}
    <p>질문이 없습니다.</p>
{% endif %}
```

## ■ 질문 목록 조회 구현하기

### ● render로 화면 출력하기 : 템플릿 사용

- <http://127.0.0.1:8000/polls/> (또는 <http://localhost:8000/polls/>)



### ● 템플릿 디렉토리

- 앱 하위에 있는 templates 디렉토리를 기본으로 사용  
ex) mysite/**polls**/templates, tutorial/**firstapp**/templates 등
- 앱이 여러 개로 구성되는 경우에 공통 템플릿을 사용할 때는  
앱 하위가 아닌 하나의 별도 디렉토리로 관리하는 것이 편리함  
ex) **mysite**/templates, **tutorial**/templates 등

## ■ 질문 상세 조회 구현하기

### ● Question 모델 데이터 중 PK(id) 로 조회

- polls/urls.py

```
from django.urls import path, include
from . import views

urlpatterns = [
    path('', views.index),
    path('<int:question_id>/', views.detail),
]
```

- config/urls.py

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

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

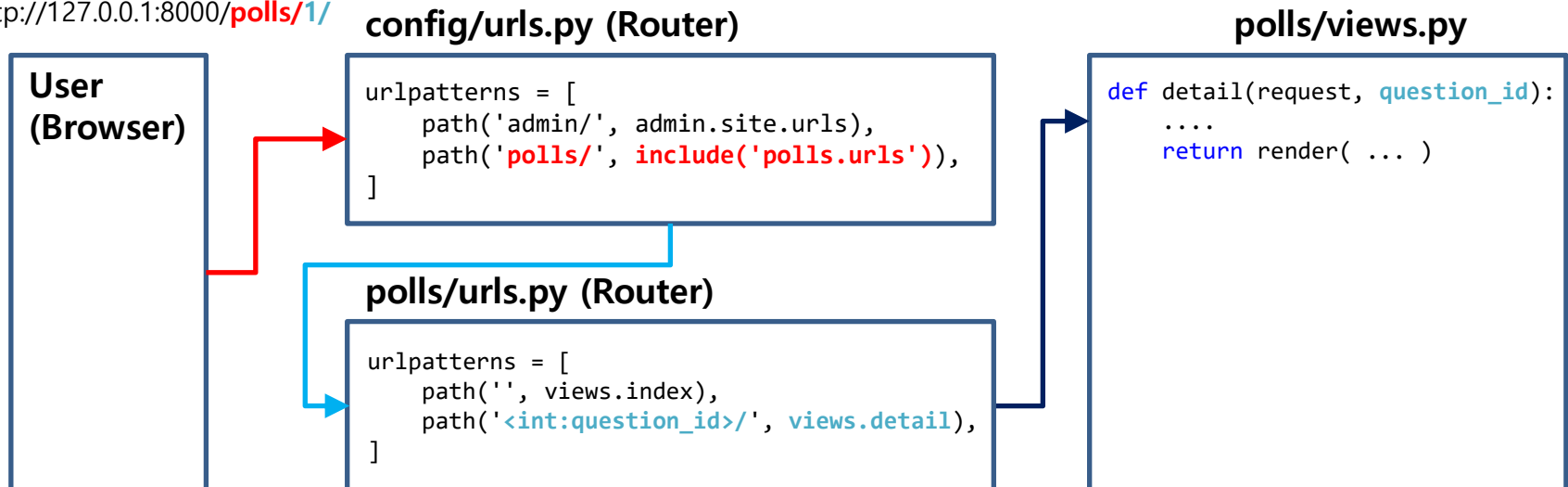
## ■ 질문 상세 조회 구현하기

### ● Question 모델 데이터 중 PK(id) 로 조회

- polls/views.py

```
def detail(request, question_id):  
    question = Question.objects.get(id=question_id)  
    context = {  
        'question': question  
    }  
    return render(  
        request, 'polls/question_detail.html', context)
```

http://127.0.0.1:8000/polls/1/





## ■ 질문 상세 조회 구현하기

- Question 모델 데이터 중 PK(id) 로 조회

- mysite/templates/polls/question\_detail.html

```
<h1>{{ question.subject }}</h1>
```

```
<div>
```

```
    {{ question.content }}
```

```
</div>
```



## ■ 오류 화면 처리

● <http://127.0.0.1:8000/polls/100/> 접속

- id가 100인 데이터를 조회할 수 없음

```
DoesNotExist at /polls/100/
Question matching query does not exist.

Request Method: GET
Request URL: http://127.0.0.1:8000/polls/100/
Django Version: 3.2.3
Exception Type: DoesNotExist
Exception Value: Question matching query does not exist.
Exception Location: C:\Users\GGoreb\miniconda3\lib\site-packages\django\db\models\query.py, line 435, in get
Python Executable: C:\Users\GGoreb\miniconda3\python.exe
Python Version: 3.8.5
Python Path: [C:\dev\django\mysite, C:\Users\GGoreb\miniconda3\python38.zip, C:\Users\GGoreb\miniconda3\DLLs, C:\Users\GGoreb\miniconda3\lib, C:\Users\GGoreb\miniconda3, C:\Users\GGoreb\miniconda3\lib\site-packages, C:\Users\GGoreb\miniconda3\lib\site-packages\win32, C:\Users\GGoreb\miniconda3\lib\site-packages\win32\lib, C:\Users\GGoreb\miniconda3\lib\site-packages\Pythonwin\]

Server time: Thu, 03 Jun 2021 10:15:45 +0000

Traceback Switch to copy-and-paste view
O:\Users\GGoreb\miniconda3\lib\site-packages\django\core\handlers\exception.py, line 47, in inner
47, response = get_response(request)
    Local vars

O:\Users\GGoreb\miniconda3\lib\site-packages\django\core\handlers\base.py, line 181, in _get_response
181, response = wrapped_callback(request, *callback_args,
    **callback_kwargs)
    Local vars

O:\dev\django\mysite\polls\views.py, line 13, in detail
13, question = Question.objects.get(id=question_id)
    Local vars

O:\Users\GGoreb\miniconda3\lib\site-packages\django\db\models\manager.py, line 85, in manager_method
85, return getattr(self.get_queryset(), name)(*args, **kwargs)
    Local vars
```

DoesNotExist 오류 발생

오류와 관련된 코드가 노출되면

보안에 문제가 되므로 중요한 정보가

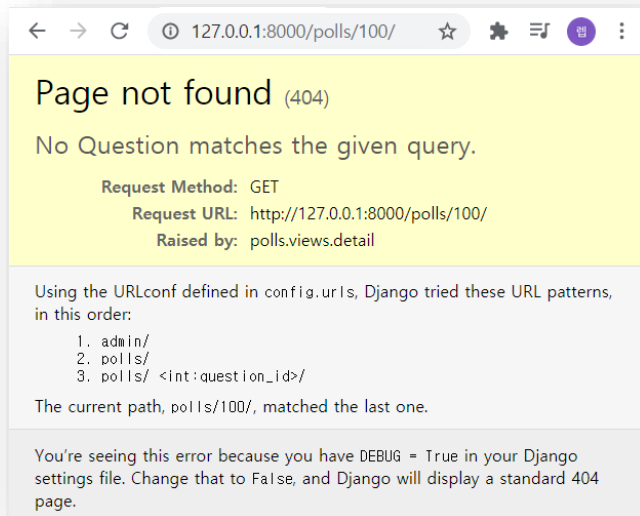
표현되지 않도록 처리

## ■ 오류 화면 처리

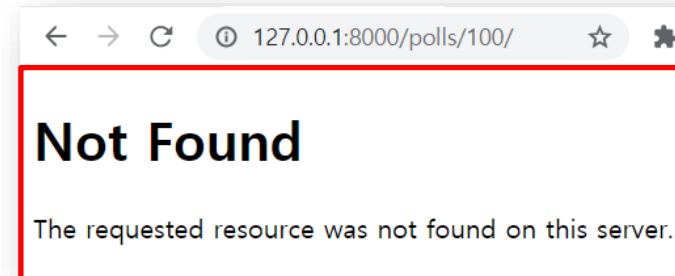
- get\_object\_or\_404 또는 get\_list\_or\_404 활용
  - polls/views.py

```
from django.shortcuts import render, get_object_or_404

def detail(request, question_id):
    question = get_object_or_404(Question, id=question_id)
    context = {
        'question': question
    }
    return render(
        request, 'polls/question_detail.html', context)
```



※ 실제 서비스 운영 시에는 DEBUG 모드를 False로 변경



## ■ URL에 이름을 지정하고 사용하기

### ● URL 하드코딩 문제 해결하기

- URL의 규칙이 변경되는 경우 URL이 쓰여진 모든 코드를 수정해야 됨
- URL마다 name을 지정하여 하드코딩 문제 해결

#### polls/urls.py

```
from django.urls import path, include
from . import views

urlpatterns = [
    path('', views.index),
    path('<int:question_id>', views.detail),
]
```

#### polls/urls.py

```
from django.urls import path, include
from . import views

urlpatterns = [
    path('', views.index, name='index'),
    path('<int:question_id>', views.detail, name='detail'),
]
```

#### mysite/templates/polls/question\_list.html

```
{% if question_list %}
<ul>
{% for question in question_list %}
<li>
<a href="/polls/{% question.id %}/">
{{ question.subject }}
</a>
</li>
{% endfor %}
</ul>
{% else %}
<p>질문이 없습니다.</p>
{% endif %}
```



#### mysite/templates/polls/question\_list.html

```
{% if question_list %}
<ul>
{% for question in question_list %}
<li>
<a href="{% url 'detail' question.id %}">
{{ question.subject }}
</a>
</li>
{% endfor %}
</ul>
{% else %}
<p>질문이 없습니다.</p>
{% endif %}
```

## ■ URL에 이름을 지정하고 사용하기

### ● URL 네임스페이스 사용하기

- 프로젝트에 앱이 여러 개 인 경우 URL name이 중복 될 수도 있음
- **App 마다 namespace을 지정**하여 name 중복 문제 해결

polls/urls.py

```
from django.urls import path, include
from . import views

app_name = 'polls'

urlpatterns = [
    path('', views.index, name='index'),
    path('<int:question_id>/', views.detail, name='detail'),
]
```

mysite/templates/polls/question\_list.html

```
{% for question in question_list %}
    <li>
        <a href="{% url 'polls:detail' question.id %}">
            {{ question.subject }}
        </a>
    </li>
{% endfor %}
```

## ■ 답변 등록 기능 만들기

### ● 질문 상세 조회 템플릿에 답변 등록 버튼 만들기

- mysite/templates/polls/question\_detail.html

```
<h1>{{ question.subject }}</h1>

<div>
    {{ question.content }}
</div>

<form
    action="{% url 'polls:answer_create' question.id %}" method="post">

    {% csrf_token %}

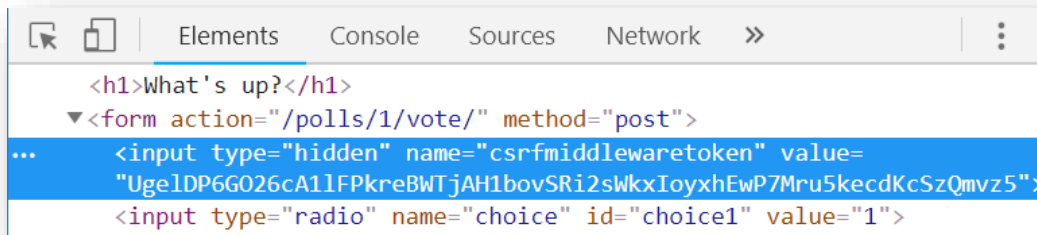
    <textarea name="content" id="content" rows="15"></textarea>
    <input type="submit" value="답변 등록">
</form>
```

## ■ 답변 등록 기능 만들기

### ● CSRF (Cross Site Request Forgery)

- 사이트 간 요청 위조
- <http://ggoreb.com/article/1> 과 같은 URL의 패턴을 분석하여 일반적인 방법으로 접근 할 수 없는 페이지에 접근하는 등 취약점을 공격하는 방법
- referer (referrer의 오타, 요청 전 페이지) 정보를 확인하거나 토큰(랜덤값)을 발급하여 정상적인 접근인지 확인
- 장고에서는 템플릿의 태그 기능을 이용하여 토큰을 발급하는 기능 제공

{% csrf\_token %}



```
<h1>What's up?</h1>
▼ <form action="/polls/1/vote/" method="post">
... <input type="hidden" name="csrfmiddlewaretoken" value=
    "Uge1DP6G026cA1lFPkreBWTjAH1bovSRi2sWkxIoyxEwP7Mru5kecdKcSzQmvz5">
    <input type="radio" name="choice" id="choice1" value="1">
```

## ■ 답변 등록 기능 만들기

### ● 답변 등록을 위한 URL 매핑 등록하기

- polls/urls.py

```
from django.urls import path, include
from . import views

app_name = 'polls'

urlpatterns = [
    path('', views.index, name='index'),
    path('<int:question_id>/', views.detail, name='detail'),
    path(
        'answer/create/<int:question_id>/', views.answer_create,
        name='answer_create'),
]
```



## ■ 답변 등록 기능 만들기

### ● 답변 등록을 위한 URL 매핑 등록하기

- polls/views.py

```
from django.utils import timezone
# from .models import Answer

def answer_create(request, question_id):
    question = get_object_or_404(Question, pk=question_id)
    question.answer_set.create(
        content=request.POST.get('content'), create_date=timezone.now())

    # answer = Answer(
    #     question=question, content=request.POST.get('content'),
    #     create_date=timezone.now())
    # answer.save()
```

## ■ 답변 등록 기능 만들기

### ● 답변 등록 후 상세 화면으로 이동하기

- polls/views.py

```
from django.shortcuts import render, get_object_or_404, redirect

def answer_create(request, question_id):
    question = get_object_or_404(Question, pk=question_id)
    question.answer_set.create(
        content=request.POST.get('content'), create_date=timezone.now())

    return redirect('polls:detail', question_id=question.id)

# return redirect('/polls/%s/' % question_id)
```

## ■ 답변 등록 기능 만들기

### ● 등록된 답변 표시하기

- mysite/templates/polls/question\_detail.html

```
<h1>{{ question.subject }}</h1>

<div>
    {{ question.content }}
</div>

<h5> {{ question.answer_set.count }}개의 답변이 있습니다.</h5>

<ul>
    {% for answer in question.answer_set.all %}
    <li>{{ answer.content }}</li>
    {% endfor %}
</ul>

...
```

## ■ 답변 등록 기능 만들기

### ● 결과 확인

/polls/2/ (답변 등록)

# 하드웨어

컴퓨터 장치가 아닌 것은?

0개의 답변이 있습니다.

의 자

답변 등록

/polls/2/ (답변 확인)

# 하드웨어

컴퓨터 장치가 아닌 것은?

1개의 답변이 있습니다.

- 의 자

답변 등록

## ■ 화면 꾸미기

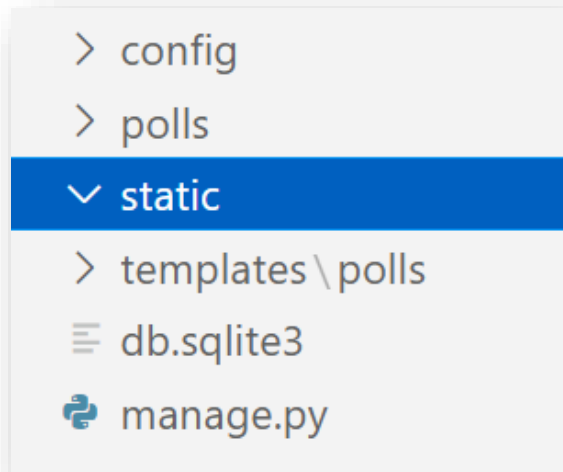
### ● 설정 파일에 static 디렉토리 위치 추가하기

- mysite/settings.py

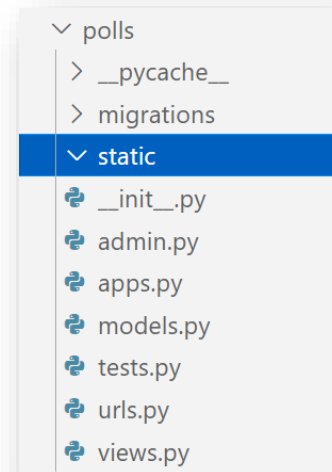
```
# 기본 주소
STATIC_URL = '/static/'

# 기본 경로인 [App]/static과 아래 경로 함께 사용
STATICFILES_DIRS = [
    BASE_DIR / 'static'
]
```

- static 디렉토리 생성



또는



## ■ 화면 꾸미기

### ● 스타일시트 작성 및 적용하기

- mysite/static/style.css

```
textarea {  
    width: 100%;  
}  
  
input[type=submit] {  
    margin-top: 10px;  
}
```

- mysite/templates/polls/question\_detail.html

```
{% load static %}  
<link rel="stylesheet" type="text/css" href="{% static 'style.css' %}">  
  
<h1>{{ question.subject }}</h1>  
  
....
```

## ■ 화면 꾸미기

### ● 결과 확인

/polls/2/

# 하드웨어

컴퓨터 장치가 아닌 것은?

1개의 답변이 있습니다.

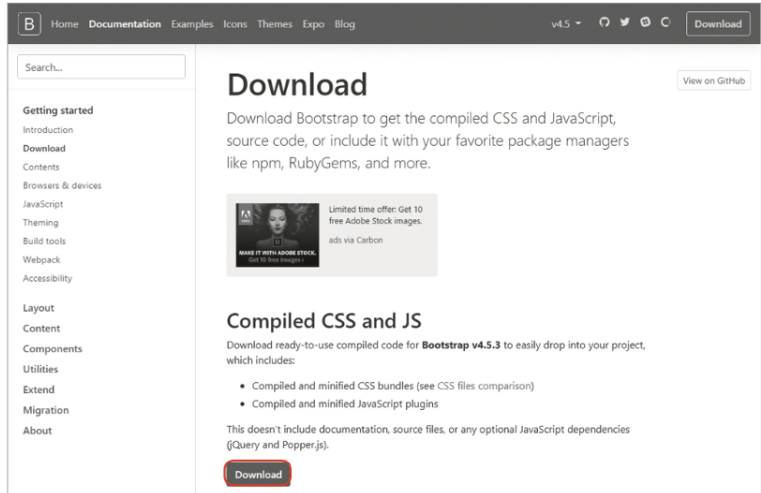
- 의자

답변 등록

# ■ Bootstrap 적용하기

## ● 부트스트랩 파일 다운로드

<http://getbootstrap.com>



### ▶ 부트스트랩 4.5.3 버전 다운로드

: [getbootstrap.com/docs/4.5/getting-started/download](http://getbootstrap.com/docs/4.5/getting-started/download)

### ▶ 내려받은 파일의 압축을 해제한 후 bootstrap.min.css 파일만 복사해서 mysite/static 디렉터리에 저장

#### • 압축 파일에 있는 bootstrap.min.css 경로

bootstrap-4.5.3-dist.zip/bootstrap-4.5.3-dist/css/bootstrap.min.css

#### • 복사할 경로

C:/projects/mysite/static/bootstrap.min.css



## ■ Bootstrap 적용하기

### ● 질문 목록 템플릿에 부트스트랩 적용하기

- `mysite/templates/polls/question_list.html`

```
{% load static %}

<link rel="stylesheet" type="text/css"
      href="{% static 'bootstrap.min.css' %}">

<div class="container my-3">
  <table class="table">
    <thead>
      <tr class="thead-dark">
        <th>번호</th>
        <th>제목</th>
        <th>작성일시</th>
      </tr>
    </thead>
```

## ■ Bootstrap 적용하기

### ● 질문 목록 템플릿에 부트스트랩 적용하기

- mysite/templates/polls/question\_list.html

```
<tbody>
  {% if question_list %}
  {% for question in question_list %}
  <tr>
    <td>{{ forloop.counter }}</td>
    <td>
      <a href="{% url 'polls:detail' question.id %}">
        {{ question.subject }}
      </a>
    </td>
    <td>{{ question.pub_date }}</td>
  </tr>
  {% endfor %}
  {% else %}
  <tr>
    <td colspan="3">질문이 없습니다.</td>
  </tr>
  {% endif %}
</tbody>
</table>
</div>
```

번호	제목	작성일시
1	<a href="#">하드웨어</a>	June 14, 2021, 9:36 a.m.
2	<a href="#">프로그래밍</a>	June 14, 2021, 9:36 a.m.

## ■ Bootstrap 적용하기

### ● 질문 상세 템플릿에 부트스트랩 적용하기

- `mysite/templates/polls/question_detail.html`

```
{% load static %}

<link rel="stylesheet" type="text/css"
      href="{% static 'bootstrap.min.css' %}">

<div class="container my-3">
  <h2 class="border-bottom py-2">{{ question.subject }}</h2>
  <div class="card my-3">
    <div class="card-body">
      <div class="card-text" style="white-space: pre-line;">
        {{ question.content }}
      </div>
      <div class="d-flex justify-content-end">
        <div class="badge badge-light p-2">
          {{ question.pub_date }}
        </div>
      </div>
    </div>
  </div>
</div>
```

## ■ Bootstrap 적용하기

### ● 질문 상세 템플릿에 부트스트랩 적용하기

- mysite/templates/polls/question\_detail.html

```
<h5 class="border-bottom my-3 py-2">
    {{question.answer_set.count}}개의 답변이 있습니다.
</h5>
{% for answer in question.answer_set.all %}
<div class="card my-3">
    <div class="card-body">
        <div class="card-text" style="white-space: pre-line;">
            {{ answer.content }}
        </div>
        <div class="d-flex justify-content-end">
            <div class="badge badge-light p-2">
                {{ answer.create_date }}
            </div>
        </div>
    </div>
</div>
{% endfor %}
```

## ■ Bootstrap 적용하기

### ● 질문 상세 템플릿에 부트스트랩 적용하기

- `mysite/templates/polls/question_detail.html`

```
<form action="{% url 'polls:answer_create' question.id %}"
      method="post" class="my-3">
  {% csrf_token %}
  <div class="form-group">
    <textarea name="content" id="content"
      class="form-control" rows="10"></textarea>
  </div>
  <input type="submit" value="답변등록" class="btn btn-primary">
</form>
</div>
```

## ■ 연습문제

### ● 템플릿 상속 사용하기

- question\_list / question\_detail 의 중복 코드를 base.html로 분리

## ■ 질문 등록 기능 만들기

### ● 질문 등록 버튼 만들기

- mysite/templates/polls/question\_list.html

```
.....
<tr>
    <td colspan="3">질문이 없습니다.</td>
</tr>
{% endif %}
</tbody>
</table>
<a href="{% url 'polls:question_create' %}" class="btn btn-primary">
    질문 등록하기
</a>
</div>
```

## ■ 질문 등록 기능 만들기

### ● 질문 등록 URL 매핑

- polls/urls.py

```
from django.urls import path, include
from . import views

app_name = 'polls'

urlpatterns = [
    path('', views.index, name='index'),
    path('<int:question_id>/', views.detail, name='detail'),
    path(
        'answer/create/<int:question_id>/', views.answer_create,
        name='answer_create'),
    path(
        'question/create/', views.question_create,
        name='question_create'),
]
```



## ■ 질문 등록 기능 만들기

### ● 질문 등록 함수

- polls/views.py

```
from .forms import QuestionForm

def question_create(request):
    form = QuestionForm()
    return render(request, 'polls/question_form.html', {'form': form})
```

- polls/forms.py

```
from django import forms
from .models import Question

class QuestionForm(forms.ModelForm):
    class Meta:
        model = Question
        fields = ['subject', 'content']
```

## ■ 질문 등록 기능 만들기

### ● 질문 등록 템플릿

- mysite/templates/polls/question\_form.html

```
{% extends 'base.html' %}

{% block content %}
<div class="container">
  <h5 class="my-3 border-bottom pb-2">질문등록</h5>
  <form method="post" class="post-form my-3">
    {% csrf_token %}
    {{ form.as_p }}
    <button type="submit" class="btn btn-primary">저장하기</button>
  </form>
</div>
{% endblock %}
```

질문등록

Subject:

Content:

저장하기

## ■ 질문 등록 기능 만들기

### ● 입력된 질문 저장하기

- polls/views.py

```
from .forms import QuestionForm

def question_create(request):
    if request.method == 'POST':
        form = QuestionForm(request.POST)
        if form.is_valid():
            question = form.save(commit=False)
            question.create_date = timezone.now()
            question.save()
            return redirect('polls:index')
    else:
        form = QuestionForm()
        context = {'form': form}
        return render(request, 'polls/question_form.html', context)
```

## ■ 질문 등록 기능 만들기

### ● 결과 확인

질문등록

Subject:

Content:

궁금합니다

[저장하기](#)



번호	제목	작성일시
1	<a href="#">테스트</a>	June 15, 2021, 2:12 a.m.
2	<a href="#">하드웨어</a>	June 14, 2021, 9:36 a.m.
3	<a href="#">프로그래밍</a>	June 14, 2021, 9:36 a.m.

[질문 등록하기](#)

## ■ 질문 등록 기능 만들기

### ● 폼에 부트스트랩 적용하기

- polls/forms.py

```
class QuestionForm(forms.ModelForm):  
    class Meta:  
        model = Question  
        fields = ['subject', 'content']  
        widgets = {  
            'subject': forms.TextInput(attrs={'class': 'form-control'}),  
            'content': forms.Textarea(  
                attrs={'class': 'form-control', 'rows': 10}),  
        }  
}
```

The diagram illustrates the visual transformation of a Django form. On the left, a basic HTML form titled "질문등록" (Question Registration) is shown. It features a "Subject:" label followed by a simple text input field, and a "Content:" label followed by a large, empty text area. At the bottom left, there is a blue button labeled "저장하기" (Save). A large blue arrow points from this basic form to the right, where the same form is shown after applying Bootstrap styling. The styled version features a clean, modern layout with a light gray border and a white background. The "Subject:" label is in a smaller font, and the input field has a rounded rectangle and a light gray border. The "Content:" label is also in a smaller font, and the text area has a rounded rectangle and a light gray border. The "저장하기" button is now a blue button with white text, positioned at the bottom left of the form.

## ■ 질문 등록 기능 만들기

### ● label 속성 제어

- polls/forms.py

```
class QuestionForm(forms.ModelForm):
    class Meta:
        model = Question
        fields = ['subject', 'content']
        widgets = {
            'subject': forms.TextInput(attrs={'class': 'form-control'}),
            'content': forms.Textarea(
                attrs={'class': 'form-control', 'rows': 10}),
        }
        labels = {
            'subject': '제목',
            'content': '내용',
        }
```

질문등록

제목:

내용:

저장하기

## ■ 질문 등록 기능 만들기

### ● 수작업으로 폼 작성하기

- polls/forms.py

```
class QuestionForm(forms.ModelForm):
    class Meta:
        model = Question
        fields = ['subject', 'content']

        # widgets 항목 제거

        labels = {
            'subject': '제목',
            'content': '내용',
        }
```

## ■ 질문 등록 기능 만들기

### ● 수작업으로 폼 작성하기

- mysite/templates/polls/question\_form.html

```
{% extends 'base.html' %}

{% block content %}
<div class="container">
  <h5 class="my-3 border-bottom pb-2">질문등록</h5>
  <form method="post" class="post-form my-3">
    {% csrf_token %}

    <!-- 오류표시 Start -->
    {% if form.errors %}
    <div class="alert alert-danger" role="alert">
      {% for field in form %}
        {% if field.errors %}
          <strong>{{ field.label }}</strong>
          {{ field.errors }}
        {% endif %}
      {% endfor %}
    </div>
    {% endif %}
    <!-- 오류표시 End -->
```



## ■ 질문 등록 기능 만들기

### ● 수작업으로 폼 작성하기

- mysite/templates/polls/question\_form.html

```
<div class="form-group">
  <label for="subject">제목</label>
  <input type="text" class="form-control" name="subject"
    value="{{ form.subject.value|default_if_none:'' }}">
</div>
<div class="form-group">
  <label for="content">내용</label>
  <textarea class="form-control" name="content" rows="10">
    {{ form.content.value|default_if_none:'' }}</textarea>
  </div>
  <button type="submit" class="btn btn-primary">저장하기</button>
</form>
</div>
{% endblock %}
```

## ■ 질문 등록 기능 만들기

### ● 결과 확인

질문등록

제목

내용

저장하기



질문등록

제목

• This field is required.

내용

• This field is required.

제목

내용

## ■ 답변 등록 기능에 장고 폼 적용하기

### ● AnswerForm 클래스 작성

- polls/forms.py

```
from .models import Answer

class AnswerForm(forms.ModelForm):
    class Meta:
        model = Answer
        fields = ['content']
        labels = {
            'content': '답변내용',
        }
```

## ■ 답변 등록 기능에 장고 폼 적용하기

### ● answer\_create 함수 수정

- polls/views.py

```
from .forms import AnswerForm

def answer_create(request, question_id):
    question = get_object_or_404(Question, pk=question_id)
    if request.method == "POST":
        form = AnswerForm(request.POST)
        if form.is_valid():
            answer = form.save(commit=False)
            answer.create_date = timezone.now()
            answer.question = question
            answer.save()
            return redirect('polls:detail', question_id=question.id)
    else:
        form = AnswerForm()
    context = {'question': question, 'form': form}
    return render(request, 'polls/question_detail.html', context)
```

## ■ 답변 등록 기능에 장고 폼 적용하기

### ● 질문 상세 템플릿에 오류 표시 영역 추가

- mysite/templates/polls/question\_detail.html

```
<form ... >
    {% csrf_token %}

    <!-- 오류표시 Start -->
    {% if form.errors %}
    <div class="alert alert-danger" role="alert">
        {% for field in form %}
            {% if field.errors %}
                <strong>{{ field.label }}</strong>
                {{ field.errors }}
            {% endif %}
        {% endfor %}
    </div>
    {% endif %}
    <!-- 오류표시 End -->

    ...

</form>
```

## ■ 답변 등록 기능에 장고 폼 적용하기

### ● 결과 확인

### 테스트

궁금합니다

June 15, 2021, 2:12 a.m.

0개의 답변이 있습니다.

답변등록



### 테스트

궁금합니다

June 15, 2021, 2:12 a.m.

0개의 답변이 있습니다.

**답변내용**

- This field is required.

## ■ 네비게이션 기능 추가하기

### ● 로고 / 로그인 링크 추가하기

- mysite/templates/navbar.html

```
<!-- 네비게이션바 -->
<nav
  class="navbar navbar-expand-lg navbar-light bg-light border-bottom">
  <a class="navbar-brand" href="{% url 'polls:index' %}">투표</a>
  <button class="navbar-toggler ml-auto" type="button"
    data-toggle="collapse" data-target="#navbarNav"
    aria-controls="navbarNav" aria-expanded="false"
    aria-label="Toggle navigation">
    <span class="navbar-toggler-icon"></span>
  </button>
  <div class="collapse navbar-collapse flex-grow-0" id="navbarNav">
    <ul class="navbar-nav">
      <li class="nav-item ">
        <a class="nav-link" href="#">로그인</a>
      </li>
    </ul>
  </div>
</nav>
```

## ■ 네비게이션 기능 추가하기

### ● 로고 / 로그인 링크 추가하기

- mysite/templates/base.html

```
...  
<body>  
  <!-- 네비게이션바 -->  
  {% include 'navbar.html' %}  
  
  <!-- 기본 템플릿 안에 삽입될 내용 Start -->  
  {% block content %}  
  {% endblock %}  
  <!-- 기본 템플릿 안에 삽입될 내용 End -->  
</body>  
...
```



## ■ 네비게이션 기능 추가하기

### ● 결과 확인

투표

번호	제목	작성일시
1	<a href="#">테스트</a>	June 15, 2021, 2:12 a.m.
2	<a href="#">하드웨어</a>	June 14, 2021, 9:36 a.m.
3	<a href="#">프로그래밍</a>	June 14, 2021, 9:36 a.m.

질문 등록하기

투표

## 테스트

궁금합니다

June 15, 2021, 2:12 a.m.

0개의 답변이 있습니다.

## ■ 네비게이션 기능 추가하기

### ● 부트스트랩의 기능 동작에 필요한 파일 추가 (CDN)

- mysite/templates/base.html

```
...
<head>
  <meta charset="utf-8">
  <meta name="viewport"
    content="width=device-width, initial-scale=1, shrink-to-fit=no">
  <link rel="stylesheet" type="text/css" href="{% static 'bootstrap.min.css' %}">
  <link rel="stylesheet" type="text/css" href="{% static 'style.css' %}">

  <script
    src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js">
  </script>
  <script
    src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js">
  </script>

  <title>Hello, MySite!</title>
</head>
...
```