Authentication

멋쟁이사자처럼 8기 운영진 이경연

오늘은

- 회원가입/로그인/로그아웃
- 로그인 상태에 따라 다른 navbar
- 작성자만 Post, Comment 수정/삭제
- Social Login (카카오)

과제 복습

- base.html
- base.css
- 템플릿 상속

```
{% load static %}
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="stylesheet" type="text/css" href="{% static 'base.css' %}?v=0.1">
  <title>My Blog</title>
</head>
<body>
  <div class="nav-bar">
    <a class="nav-item" href="{% url 'home' %}">HOME</a>
    <a class="nav-item">로그인</a>
    <a class="nav-item">회원가입</a>
    <a class="nav-item">로그아웃</a>
  </div>
  {% block content %}
  {% endblock content %}
</body>
</html>
```

과제 복습

- base.html
- base.css
- 템플릿 상속

```
{% extends 'base.html' %}
{% block content %}
  <div>
    <div>
      <div>{{ post.title }}</div>
      <div>{{ post.content }}</div>
    </div>
    <a href="{% url 'home' %}">홈으로</a>
    <a href="{% url 'edit' post.pk %}">수정하기</a>
    <a href="{% url 'delete' post.pk %}">삭제하기</a>

    {% for comment in post.comments.all %}
    <li>{{ comment.content }}</li>
    <a href="{% url 'delete_comment' post.pk comment.pk %}">댓글삭제</a>
    {% endfor %}

    <form method="POST">
      {% csrf_token %}
      <input type="text" name="content" placeholder="댓글을 입력하세요">
      <button type="submit">댓글 쓰기</button>
    </form>
  </div>
{% endblock content %}
```

세션 준비

- 터미널을 열고 오늘 세션을 진행할 폴더로 이동
- git clone https://github.com/mangod037/django_auth.git
- cd django_auth
- pipenv shell -> pipenv install
- cd crudproject
- python manage.py migrate
- python manage.py createsuperuser

model 만들기(?)

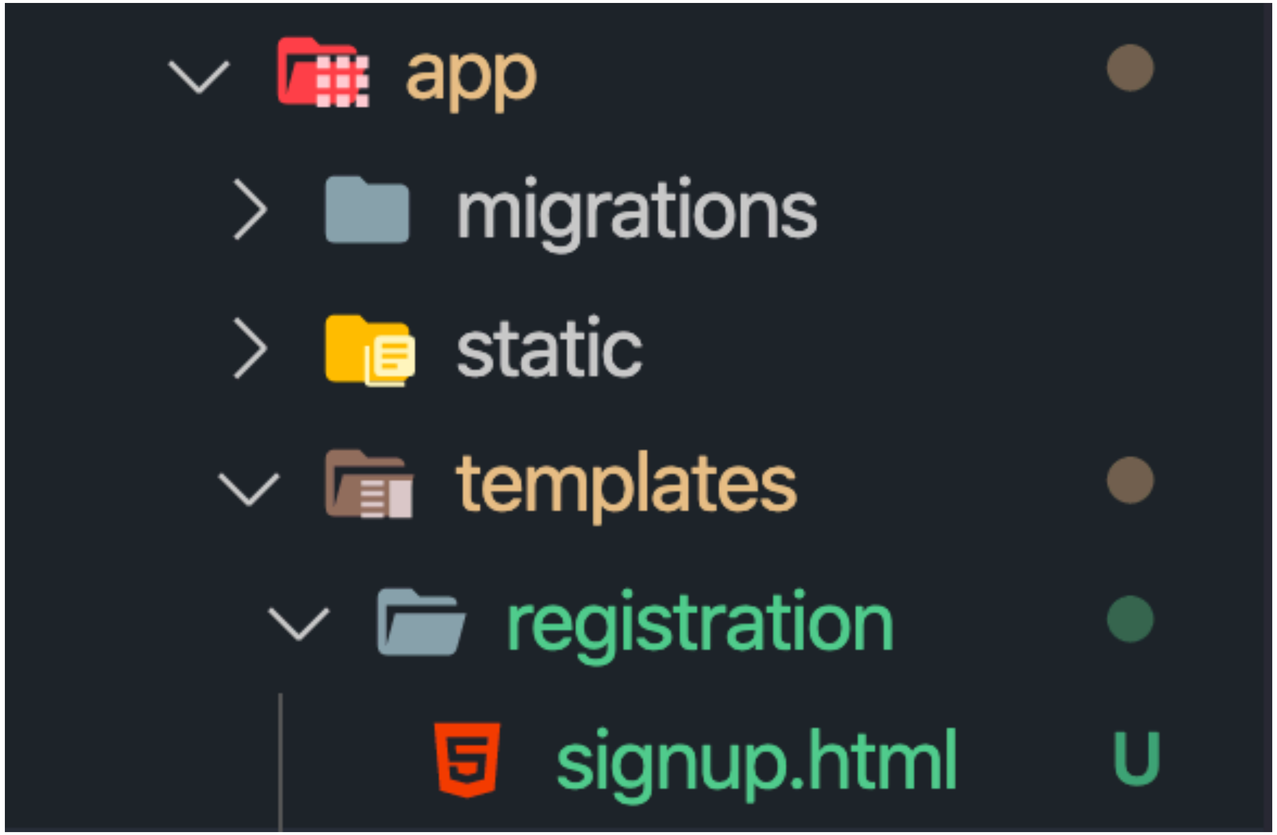
- User 모델은 있으니 모델은 만들 필요 없고 import만 해준다
- views.py

crudproject > app > views.py

```
1    from django.shortcuts import render, redirect
2    from .models import Post, Comment
3    from django.contrib.auth.models import User
4
```

template 만들기

- templates 폴더에 registration 폴더를 만든다(폴더명 오타 주의)
- 그리고 signup.html을 만든다



signup.html

```
{% block content %}
  <h1>회원가입</h1>
  <form method="POST">
    {% csrf_token %}
    <input type="text" name="username" placeholder="이름">
    <input type="password" name="password" placeholder="비밀번호">
    <button type="submit">가입하기</button>
  </form>
{% endblock content %}
```

view를 만든다 (수정: create_user)

```
60    def signup(request):
61        if (request.method == 'POST'):
62            new_user = User.objects.create(
63                username = request.POST['username'],
64                password = request.POST['password']
65            )
66            return redirect('home')
67
68        return render(request, 'registration/signup.html')
```

url을 설정해 준다

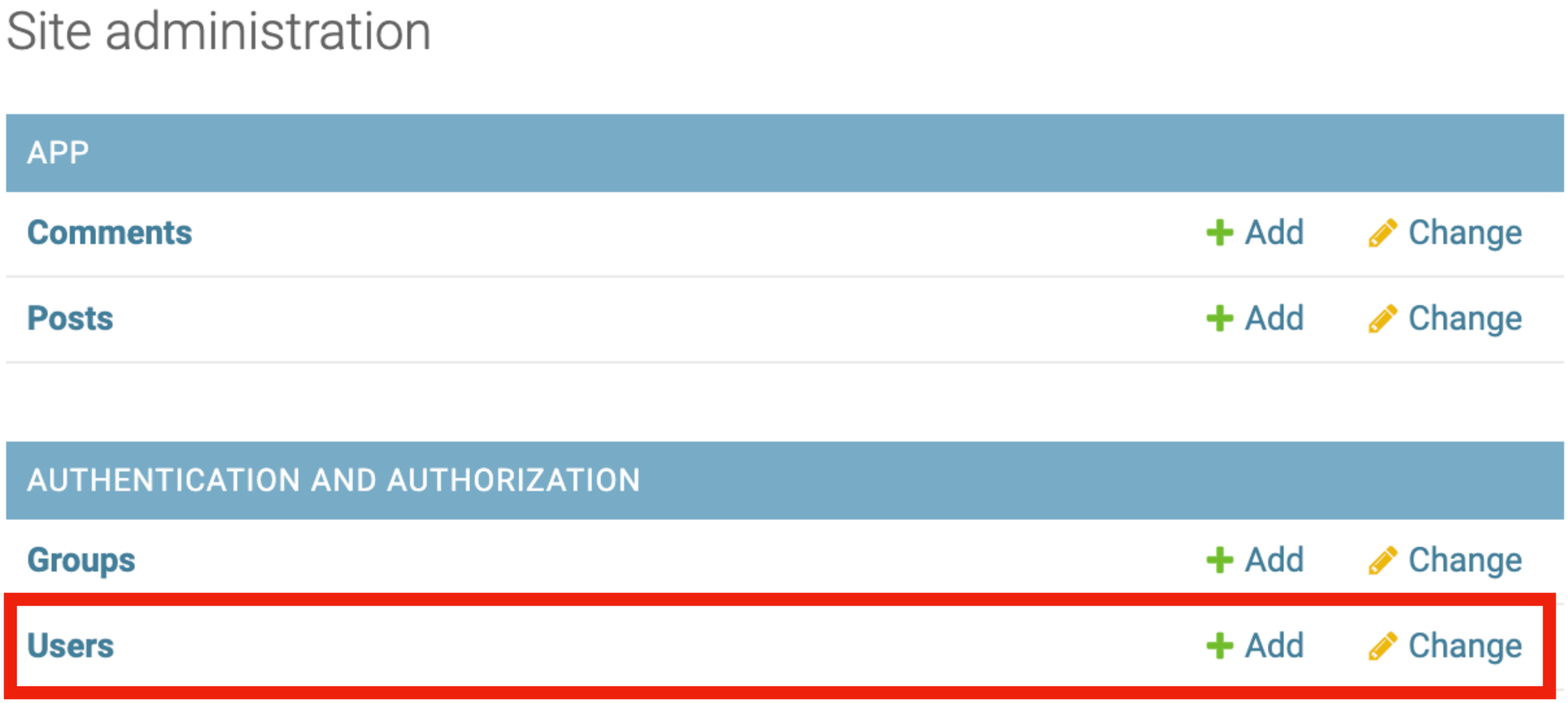
```
20    urlpatterns = [
21        # auth
22        path('registration/signup', views.signup, name="signup"),
23
24        path('admin/', admin.site.urls),
25        path('', views.home, name="home"),
26        path('new/', views.new, name="new"),
27        path('detail/<int:post_pk>', views.detail, name="detail"),
28        path('edit/<int:post_pk>', views.edit, name="edit"),
29        path('delete/<int:post_pk>', views.delete, name="delete"),
30        path('delete_comment/<int:post_pk>/<int:comment_pk>', views.de
31    ]
```

base.html 수정

```
10    <body>
11      <div class="nav-bar">
12        <a class="nav-item" href="{% url 'home' %}">HOME</a>
13        <a class="nav-item">로그인</a>
14        <a class="nav-item" href="{% url 'signup' %}">회원가입</a>
15        <a class="nav-item">로그아웃</a>
16      </div>
17      {% block content %}
18      {% endblock content %}
19    </body>
```

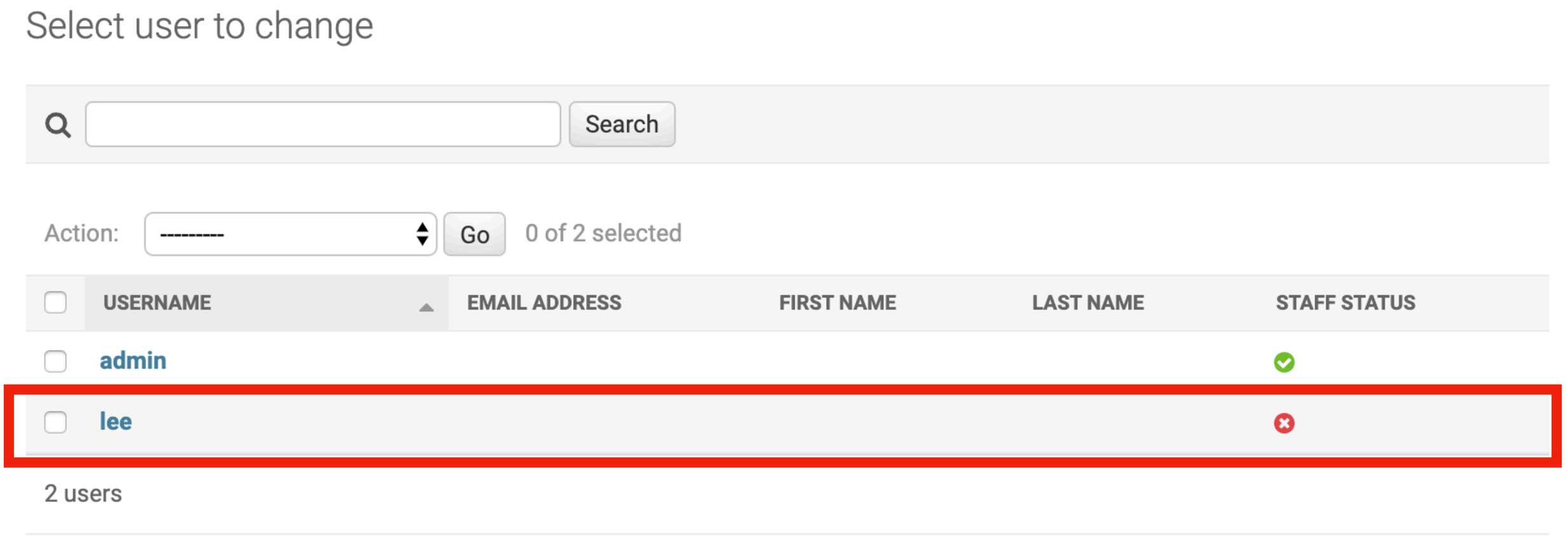
회원가입을 해 봅시다

- localhost:8000/admin



자기가 만든 계정이 잘 만들어 졌는지 확인

- localhost:8000/admin



에러!!

- 같은 username으로 회원가입을 시도하면 오류가 난다

IntegrityError at /auth/signup

UNIQUE constraint failed: auth_user.username

Request Method:	POST
Request URL:	http://localhost:8000/auth/signup
Django Version:	3.0.6
Exception Type:	IntegrityError
Exception Value:	`UNIQUE constraint failed: auth_user.username`
Exception Location:	/Users/kyoungyeon/.local/share/virtualenvs/crud-1-n-GAwKw4y0/lib/python3.7/site-packages/django/db/backends/sqlite3/base.py in execute, line 396
Python Executable:	/Users/kyoungyeon/.local/share/virtualenvs/crud-1-n-GAwKw4y0/bin/python
Python Version:	3.7.6

Python Path:	`['/Users/kyoungyeon/Desktop/test/crud-1-n/crudproject',`
	`'/usr/local/var/pyenv/versions/3.7.6/lib/python37.zip',`
	`'/usr/local/var/pyenv/versions/3.7.6/lib/python3.7',`
	`'/usr/local/var/pyenv/versions/3.7.6/lib/python3.7/lib-dynload',`
	`'/Users/kyoungyeon/.local/share/virtualenvs/crud-1-n-GAwKw4y0/lib/python3.7/site-packages']`

Server time:	Sun, 24 May 2020 08:49:07 +0000

예외처리 (views.py)

```python
def signup(request):
    if (request.method == 'POST'):
        found_user = User.objects.filter(username=request.POST['username'])
        if (len(found_user) > 0):
            error = 'username이 이미 존재합니다'
            return render(request, 'registration/signup.html', { 'error' : error })

        new_user = User.objects.create(
            username = request.POST['username'],
            password = request.POST['password']
        )
        auth.login(request, new_user)
        return redirect('home')

    return render(request, 'registration/signup.html')
```

예외처리 (signup.html)

```
{% block content %}
  <h1>회원가입</h1>
  {% if error %}
  <h3 style="color: red;">{{ error }}</h3>
  {% endif %}
  <form method="POST">
    {% csrf_token %}
    <input type="text" name="username" placeholder="이름">
    <input type="password" name="password" placeholder="비밀번호">
    <button type="submit">가입하기</button>
  </form>
{% endblock content %}
```

nav-bar 수정하기 (base.html)

```
11    <div class="nav-bar">
12      <a class="nav-item" href="{% url 'home' %}">HOME</a>
13      {% if user.is_authenticated %}
14      <div class="nav-item">안녕하세요, {{ user.username }}님</div>
15      <a class="nav-item">로그아웃</a>
16      {% else %}
17      <a class="nav-item">로그인</a>
18      <a class="nav-item" href="{% url 'signup' %}">회원가입</a>
19      {% endif %}
20    </div>
21    {% block content %}
22    {% endblock content %}
```

회원가입과 동시에 로그인 시키기

- 모듈 import

```
4    from django.contrib import auth
```

회원가입과 동시에 로그인 시키기

```
60  def signup(request):
61      if (request.method == 'POST'):
62          found_user = User.objects.get(username=request.POST['username'])
63          if (found_user is not None):
64              error = 'username이 이미 존재합니다'
65              return render(request, 'registration/signup.html', { 'error' : error })
66
67          new_user = User.objects.create(
68              username = request.POST['username'],
69              password = request.POST['password']
70          )
71          auth.login(request, new_user)
72          return redirect('home')
73
74      return render(request, 'registration/signup.html')
```

login 만들기 (template)

```
{% block content %}
  <h1>로그인</h1>
  {% if error %}
  <h3 style="color: red;">{{ error }}</h3>
  {% endif %}
  <form method="POST">
    {% csrf_token %}
    <input type="text" name="username" placeholder="이름">
    <input type="password" name="password" placeholder="비밀번호">
    <button type="submit">로그인</button>
  </form>
{% endblock content %}
```

login 만들기 (view)

```
76  def login(request):
77      if (request.method == 'POST'):
78          found_user = auth.authenticate(
79              username = request.POST['username'],
80              password = request.POST['password']
81          )
82          if (found_user is None):
83              error = '아이디 또는 비밀번호가 틀렸습니다'
84              return render(request, 'registration/login.html', { 'error': error })
85
86          auth.login(request, found_user)
87          return redirect('home')
88
89      return render(request, 'registration/login.html')
```

login 만들기 (url)

- url에 login 추가

```
path('registration/login', views.login, name="login"),
```

login 만들기 (base.html)

```
11    <div class="nav-bar">
12      <a class="nav-item" href="{% url 'home' %}">HOME</a>
13      {% if user.is_authenticated %}
14      <div class="nav-item">안녕하세요, {{ user.username }}님</div>
15      <a class="nav-item">로그아웃</a>
16      {% else %}
17      <a class="nav-item" href="{% url 'login' %}">로그인</a>
18      <a class="nav-item" href="{% url 'signup' %}">회원가입</a>
19      {% endif %}
20    </div>
```

logout 만들기 (view)

```
91  def logout(request):
92      auth.logout(request)
93      return redirect('home')
```

logout 만들기 (url)

- url에 logout 추가

```
path('registration/logout', views.logout, name="logout"),
```

logout 만들기 (base.html)

```
11      <div class="nav-bar">
12        <a class="nav-item" href="{% url 'home' %}">HOME</a>
13        {% if user.is_authenticated %}
14        <div class="nav-item">안녕하세요, {{ user.username }}님</div>
15        <a class="nav-item" href="{% url 'logout' %}">로그아웃</a>
16        {% else %}
17        <a class="nav-item" href="{% url 'login' %}">로그인</a>
18        <a class="nav-item" href="{% url 'signup' %}">회원가입</a>
19        {% endif %}
20      </div>
```

오늘은 (두 개 남음)

- 회원가입/로그인/로그아웃
- 로그인 상태에 따라 다른 navbar
- 작성자만 Post, Comment 수정/삭제
- Social Login (카카오)

권한 부여하기

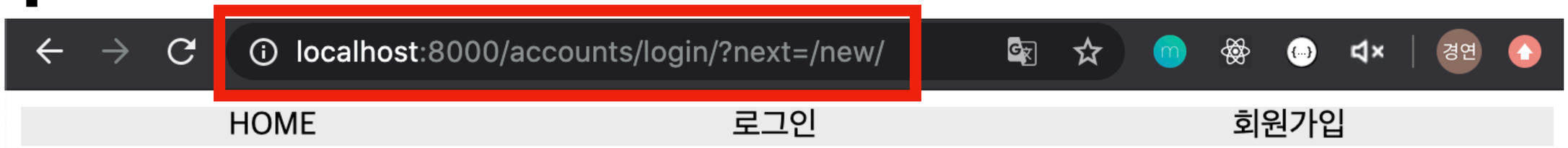
- 모듈 import

```
5    from django.contrib.auth.decorators import login_required
```

권한 부여하기

```
12  @login_required
13  def new(request):
14      if request.method == 'POST':
15          new_post = Post.objects.create(
16              title = request.POST['title'],
17              content = request.POST['content'],
18              author = request.user
19          )
20          return redirect('detail', new_post.pk)
21      return render(request, 'new.html')
```

뭐지??



Sign In

Please sign in with one of your existing third party accounts. Or, sign up for a http://127.0.0.1:8000 account and sign in below:

- Kakao

or

Username: Username

Password: Password

Remember Me:

Forgot Password? Sign In

해결

```
12    @login_required(login_url='/registration/login')
13    def new(request):
14        if request.method == 'POST':
15            new_post = Post.objects.create(
16                title = request.POST['title'],
17                content = request.POST['content'],
18                author = request.user
19            )
20            return redirect('detail', new_post.pk)
21        return render(request, 'new.html')
```

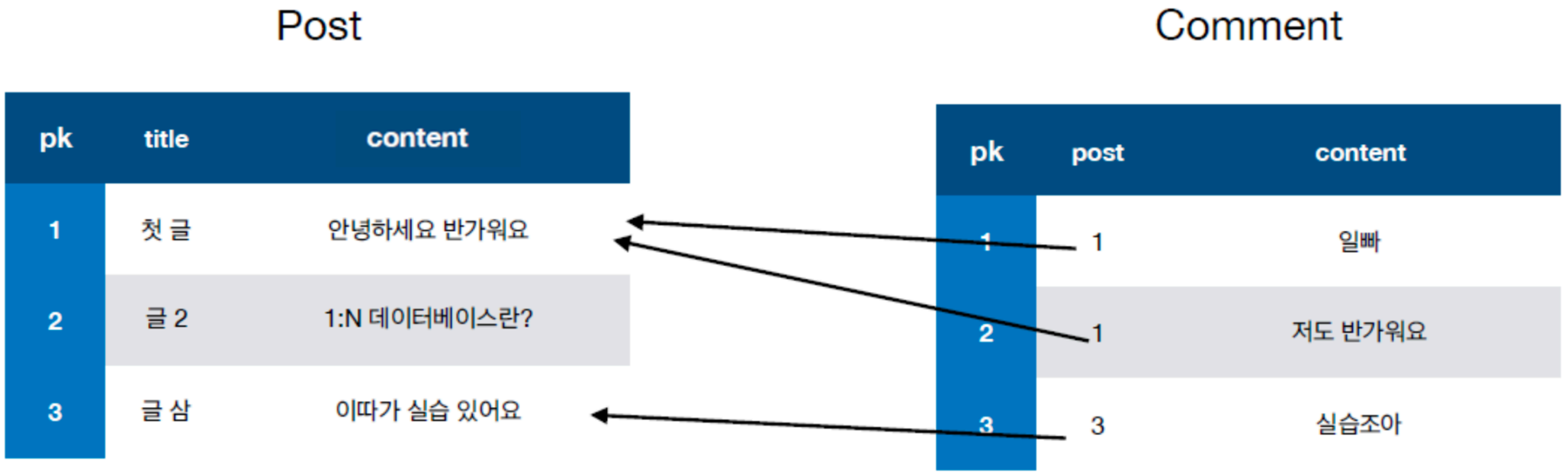
해결

```
def login(request):
    if (request.method == 'POST'):
        found_user = auth.authenticate(
            username = request.POST['username'],
            password = request.POST['password']
        )
        if (found_user is None):
            error = '아이디 또는 비밀번호가 틀렸습니다'
            return render(request, 'registration/login.html', { 'error': error })

        auth.login(
            request,
            found_user,
            backend='django.contrib.auth.backends.ModelBackend'
        )
        return redirect(request.GET.get('next', '/'))
```

작성자를 구분하려면 어떻게 해야할까

장고 모델 1:N



작성자를 저장한다 (model)

- models.py에 User를 import

```
2   from django.contrib.auth.models import User
3
4   # Create your models here.
5   class Post(models.Model):
6       title = models.CharField(max_length=200)
7       content = models.TextField()
8       author = models.ForeignKey(User, on_delete=models.CASCADE, related_name='posts')
9
10      def __str__(self):
11          return self.title
```

작성자를 저장한다 (view)

```
13  @login_required
14  def new(request):
15      if request.method == 'POST':
16          new_post = Post.objects.create(
17              title = request.POST['title'],
18              content = request.POST['content'],
19              author = request.user
20          )
21          return redirect('detail', new_post.pk)
22      return render(request, 'new.html')
```

작성자만 수정/삭제 버튼을 보여준다

```
8   <a href="{% url 'home' %}">홈으로</a>
9
10  {% if user.is_authenticated and post.author.pk == user.pk %}
11  <a href="{% url 'edit' post.pk %}">수정하기</a>
12  <a href="{% url 'delete' post.pk %}">삭제하기</a>
13  {% endif %}
14
15  {% for comment in post.comments.all %}
16  <li>{{ comment.content }}</li>
17  <a href="{% url 'delete_comment' post.pk comment.pk %}">댓글삭제</a>
18  {% endfor %}
```

실습

- Comment model에 author를 추가한다 (ForeignKey)
- 댓글을 저장할 때에 author를 같이 저장한다
- 댓글삭제 버튼을 작성자만 볼 수 있게 한다

- (migrate할 때에 default 1로)

작성자를 저장한다 (model)

```
13    class Comment(models.Model):
14        post = models.ForeignKey(Post, on_delete=models.CASCADE, related_name='comments')
15        content = models.TextField()
16        author = models.ForeignKey(User, on_delete=models.CASCADE, related_name='comments')
```

작성자를 저장한다 (view)

```
24    def detail(request, post_pk):
25        post = Post.objects.get(pk=post_pk)
26
27        if (request.method == 'POST'):
28            Comment.objects.create(
29                post = post,
30                content = request.POST['content'],
31                author = request.user
32            )
33            return redirect('detail', post_pk)
34
35        return render(request, 'detail.html', {'post': post})
```

작성자만 삭제 버튼을 보여준다

```
15    {% for comment in post.comments.all %}
16      <li>{{ comment.content }}</li>
17      {% if user.is_authenticated and comment.author.pk == user.pk %}
18      <a href="{% url 'delete_comment' post.pk comment.pk %}">댓글삭제</a>
19      {% endif %}
20    {% endfor %}
```

추가로 로그인 했을 때에만 댓글 쓸 수 있게

```
22      {% if user.is_authenticated %}
23      <form method="POST">
24        {% csrf_token %}
25        <input type="text" name="content" placeholder="댓글을 입력하세요">
26        <button type="submit">댓글 쓰기</button>
27      </form>
28      {% endif %}
```

오늘은 (한 개 남음)

- 회원가입/로그인/로그아웃
- 로그인 상태에 따라 다른 navbar
- 작성자만 Post, Comment 수정/삭제
- Social Login (카카오)

모듈 설치

- 서버를 꺼주고 (control + c)
- pipenv install django-allauth

setting.py

- INSTALLED_APPS 에
- django.contrib.sites
- allauth
- allauth.account
- allauth.socialaccount
- allauth.socialaccount.providers.kakao

setting.py

- 오타주의!!!

```
33  INSTALLED_APPS = [
34      'django.contrib.admin',
35      'django.contrib.auth',
36      'django.contrib.contenttypes',
37      'django.contrib.sessions',
38      'django.contrib.messages',
39      'django.contrib.staticfiles',
40      'app',
41
42      # social login 설정
43      'django.contrib.sites',
44      'allauth',
45      'allauth.account',
46      'allauth.socialaccount',
47      'allauth.socialaccount.providers.kakao',
48  ]
```

setting.py

```
# social login 설정
AUTHENTICATION_BACKENDS = (
    'django.contrib.auth.backends.ModelBackend',
    'allauth.account.auth_backends.AuthenticationBackend',
)
SITE_ID = 1
LOGIN_REDIRECT_URL = '/'
```

setting.py

- 오타주의!!!

```
130    # social login 설정
131    AUTHENTICATION_BACKENDS = (
132        'django.contrib.auth.backends.ModelBackend',
133        'allauth.account.auth_backends.AuthenticationBackend',
134    )
135    SITE_ID = 1
136    LOGIN_REDIRECT_URL = '/'
```

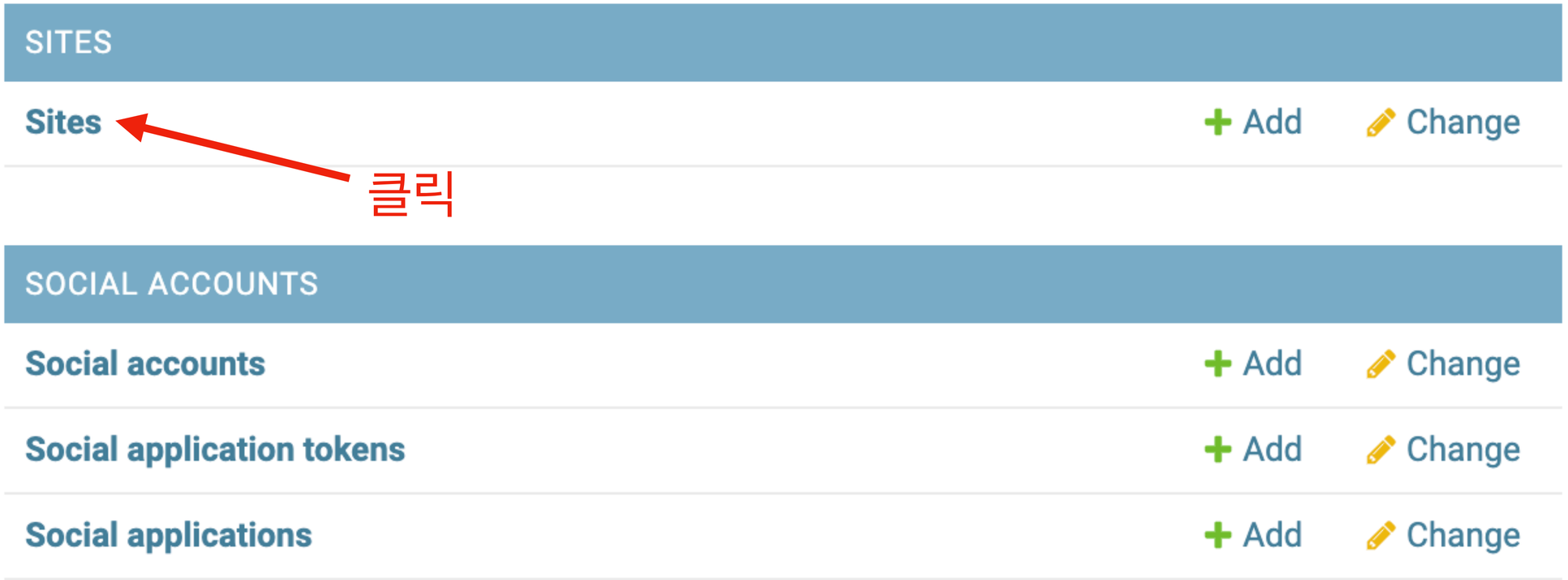
views.py

- auth.login에 backend='django.contrib.auth.backends.ModelBackend' 추가
- signup과 login view에 있음

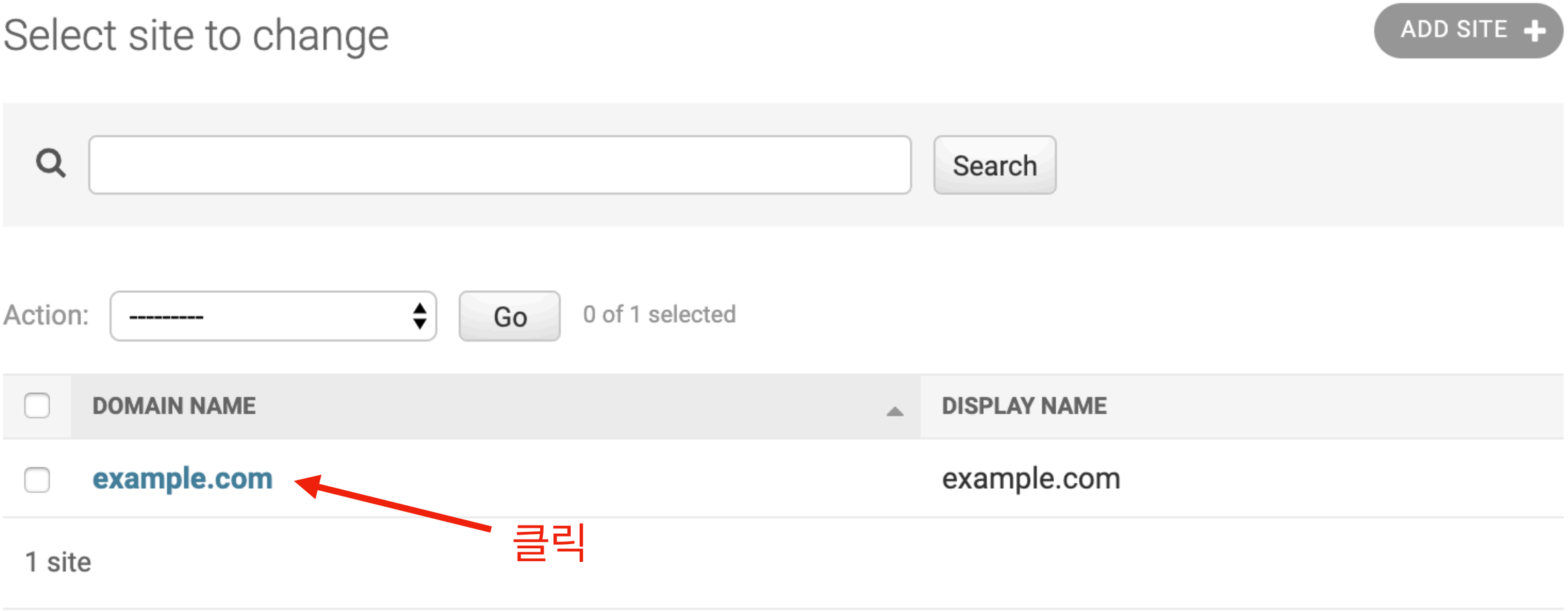
```
auth.login(
    request,
    found_user,
    backend='django.contrib.auth.backends.ModelBackend'
)
```

확인하기

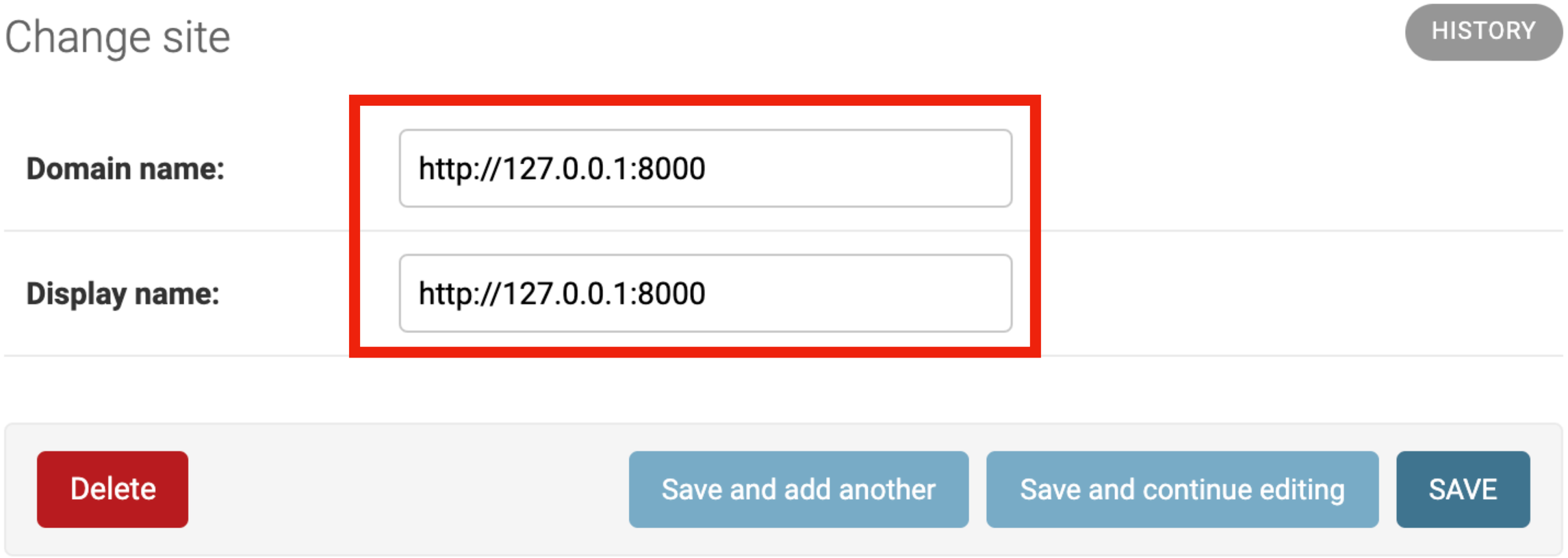
- python manage.py migrate 해주고 admin 페이지로



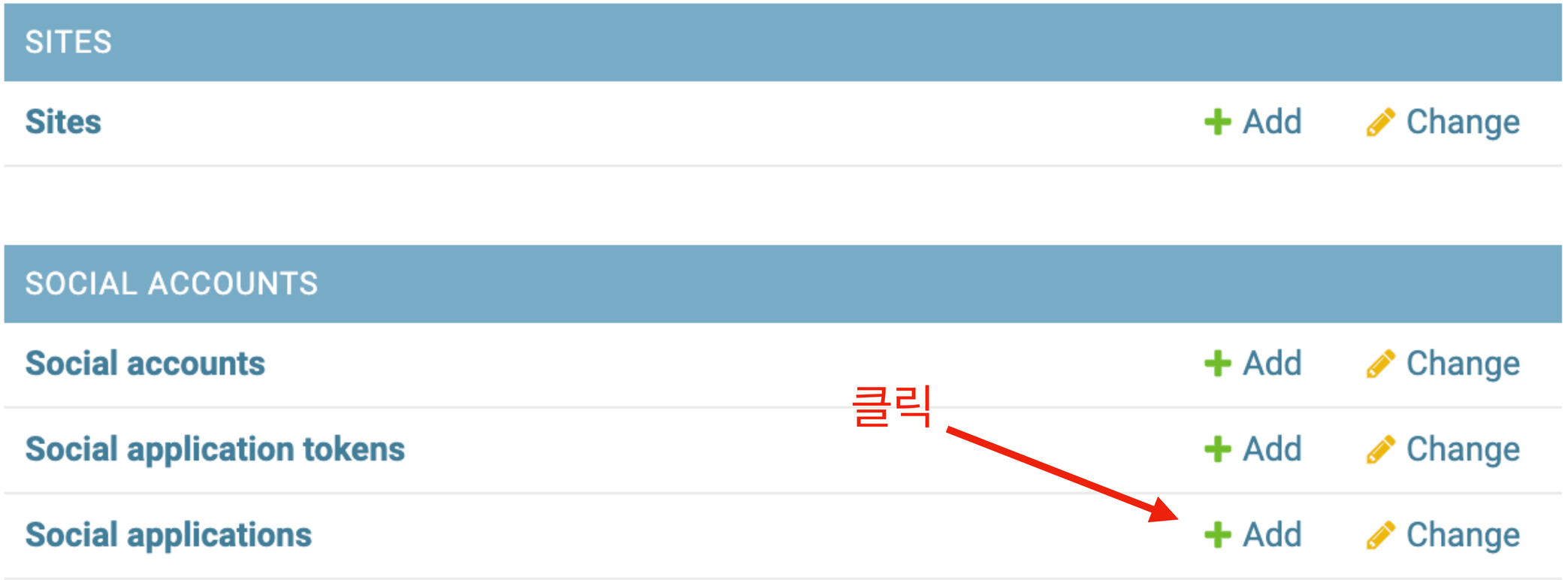
sites 수정하기 (admin)



sites 수정하기 (admin)



Social applications 추가



kakao에 어플 생성

- https://developers.kakao.com/
- 회원가입/로그인
- 내 어플리케이션
- 어플리케이션 추가하기

kakao에 어플 생성

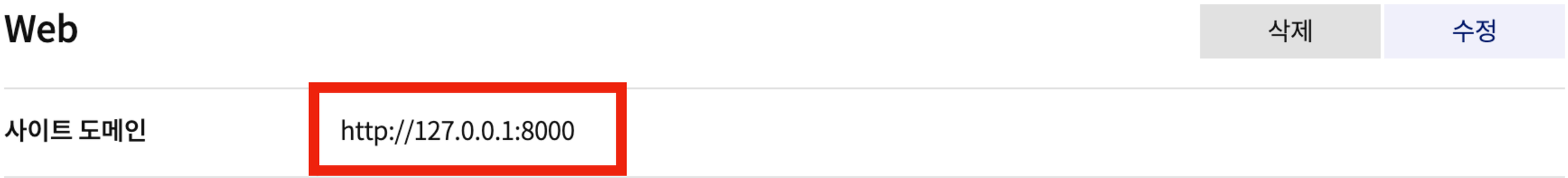
- 요약정보

앱 키

네이티브 앱 키	[redacted]
REST API 키	[redacted]
JavaScript 키	[redacted]
Admin 키	[redacted]

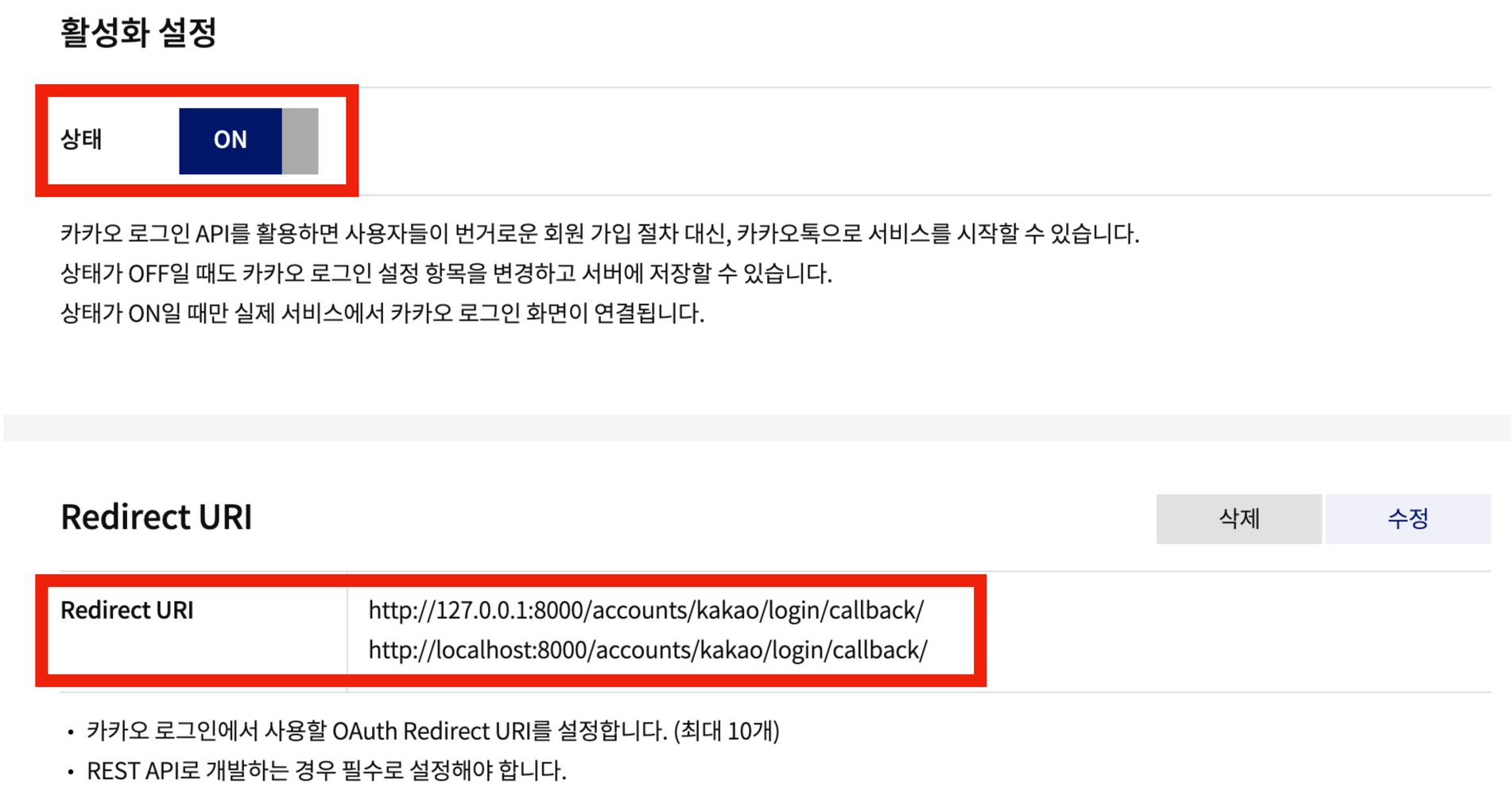
kakao에 어플 생성

- 요약정보 -> 플랫폼 -> Web



kakao에 어플 생성

- 카카오 로그인



kakao에 어플 생성

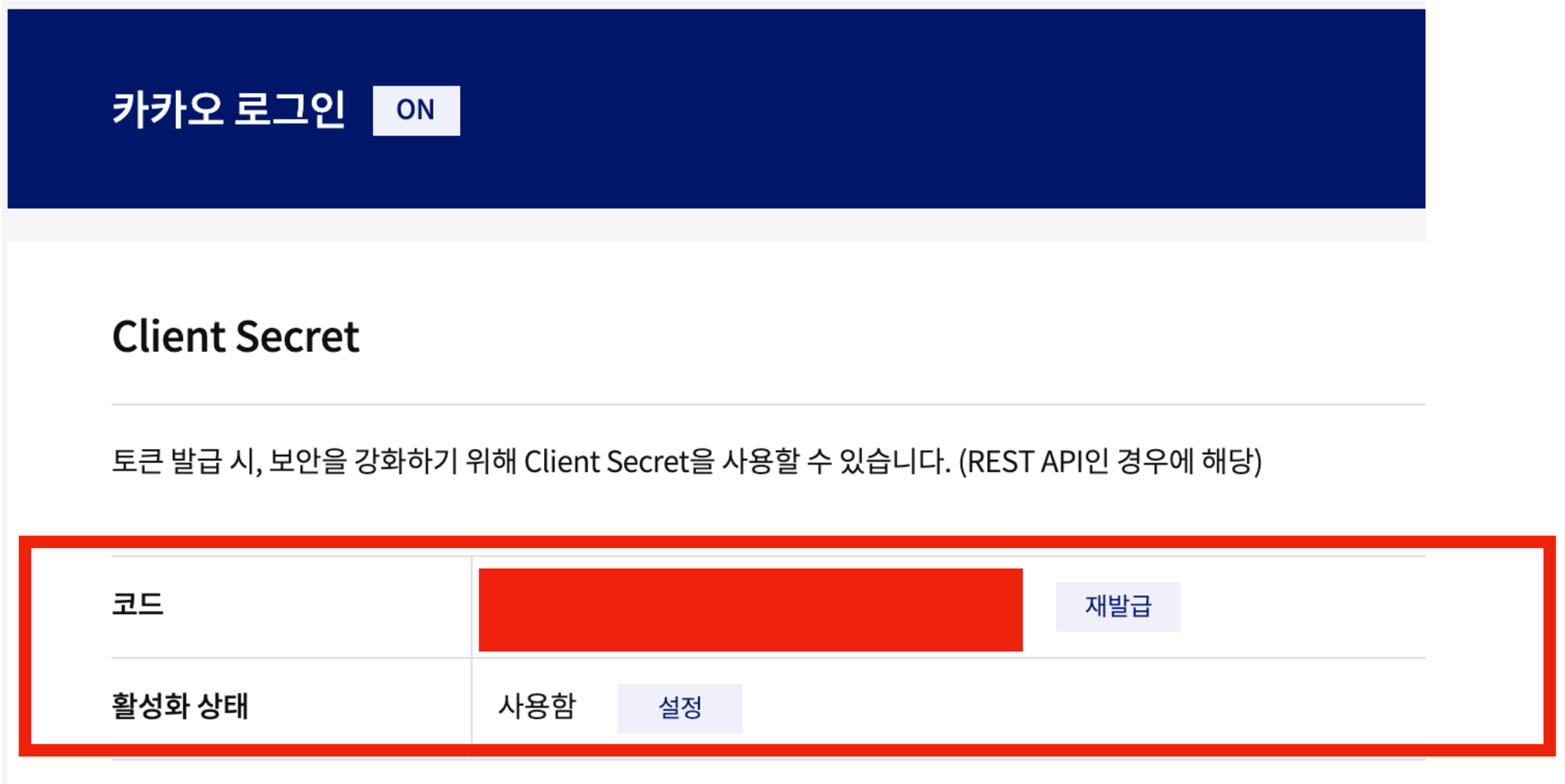
- 동의항목

개인정보 보호항목

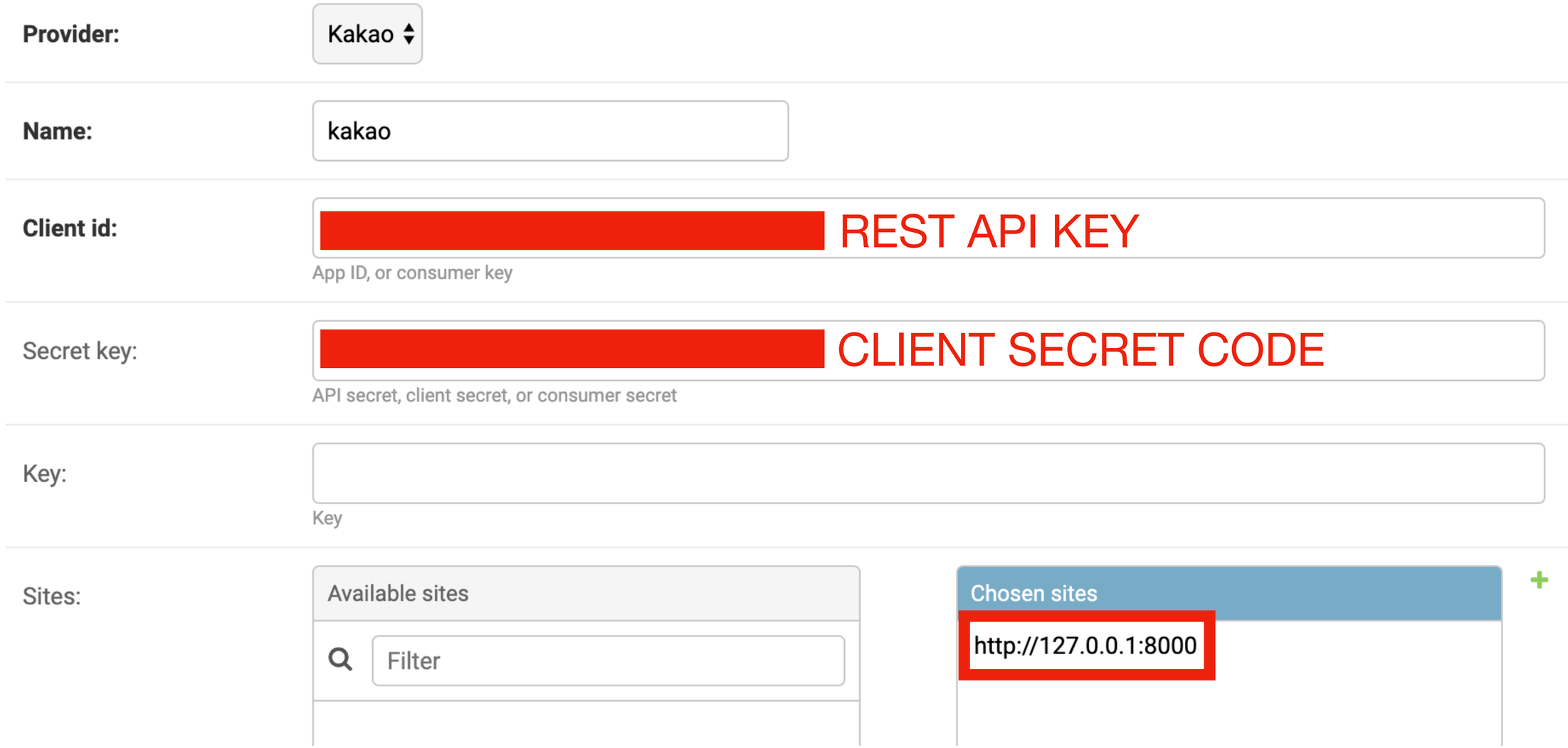
항목 이름	Id	상태	
프로필 정보(닉네임/프로필 사진)	profile	● 필수 동의	설정
카카오톡 채널 추가 상태 및 내역	plusfriends	○ 권한 없음	
카카오계정(이메일)	account_email	● 사용 안함	설정
성별	gender	● 사용 안함	설정
연령대	age_range	● 사용 안함	설정
카카오 서비스 내 친구목록	friends	● 사용 안함	설정
생일	birthday	● 사용 안함	설정

kakao에 어플 생성

- 보안



admin에 kakao에 등록



base.html

```
{% load static %}
{% load socialaccount %}
{% providers_media_js %}
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="stylesheet" type="text/css" href="{% static 'base.css' %}?v=0.1">
  <title>My Blog</title>
</head>
<body>
  <div class="nav-bar">
    <a class="nav-item" href="{% url 'home' %}">HOME</a>
    {% if user.is_authenticated %}
    <div class="nav-item">안녕하세요, {{ user.username }}님</div>
    <a class="nav-item" href="{% url 'logout' %}">로그아웃</a>
    {% else %}
    <a class="nav-item" href="{% url 'login' %}">로그인</a>
    <a class="nav-item" href="{% provider_login_url 'kakao'%}">카카오 로그인</a>
    <a class="nav-item" href="{% url 'signup' %}">회원가입</a>
    {% endif %}
  </div>
  {% block content %}
  {% endblock content %}
</body>
</html>
```

끝

- 회원가입/로그인/로그아웃
- 로그인 상태에 따라 다른 navbar
- 작성자만 Post, Comment 수정/삭제
- Social Login (카카오)

세션자료

- 오늘 세션에 사용한 프로젝트
- https://github.com/mangod037/django_auth_master
- 여기에 다 올려놨습니다

과제 (다음주 월요일까지 - 6월 1일)

- 자신이 만든 TODO LIST 과제에 로그인기능 추가하기
  (작성자만 글/댓글 수정삭제 가능하게 - 댓글 수정도 할 수 있게)
- 자기가 쓴 글만 모아볼 수 있는 페이지
- Google 소셜로그인 추가
- 참고: https://docs.djangoproject.com/en/2.2/ref/contrib/auth/

수정 1

```
58    def signup(request):
59        if (request.method == 'POST'):
60            found_user = User.objects.filter(username=request.POST['username'])
61            if (len(found_user) > 0):
62                error = 'username이 이미 존재합니다'
63                return render(request, 'registration/signup.html', { 'error' : error })
64
65            new_user = User.objects.create_user(
66                username = request.POST['username'],
67                password = request.POST['password']
68            )
69            auth.login(
70                request,
71                found_user,
72                backend='django.contrib.auth.backends.ModelBackend'
73            )
74            return redirect('home')
```

수정 2

```
20  urlpatterns = [
21      # auth
22      path('registration/signup', views.signup, name="signup"),
23      path('registration/login', views.login, name="login"),
24      path('registration/logout', views.logout, name="logout"),
25      # social login
26      path('accounts/', include('allauth.urls')),
27
28      path('admin/', admin.site.urls),
29      path('', views.home, name="home"),
30      path('new/', views.new, name="new"),
31      path('detail/<int:post_pk>', views.detail, name="detail"),
32      path('edit/<int:post_pk>', views.edit, name="edit"),
33      path('delete/<int:post_pk>', views.delete, name="delete"),
34      path('delete_comment/<int:post_pk>/<int:comment_pk>', views.
35  ]
```