The Web framework for perfectionists with deadlines.


Django form 이용하기
django

새로운 글쓰기


127.0.0.1:8000/post/new/
127.0.0.1:8000/post/new/
Welcome to SunrinPress! :)
New post
Title:
Text:

새로운 글쓰기 완료


127.0.0.1:8000/post/17/
127.0.0.1:8000/post/17/
Welcome to SunrinPress! :)
테스트 중입니다!
2017년 11월 4일 10:00 오후
음하하하하하하!!!!

글 불러와서 수정하기


127.0.0.1:8000/post/5/edi
127.0.0.1:8000/post/5/edit/
Welcome to SunrinPress! :)
New post
Title:
요가파이야
Text:
Django is taking care of validating that all the fields in our form are correct. Isn't it awesome?
As we have recently used the Django admin interface the system currently thinks we are logged in. There are a few situations that could lead to us being logged out (closing the browser, restarting the DB etc.). If you find that you are getting errors creating a post referring to a lack of a logged in user, head to the admin page http://127.0.0.1:8000/admin and log in again. This will fix the issue temporarily. There is a permanent fix awaiting you in the Homework: add security to your website! chapter after the main tutorial.
Save

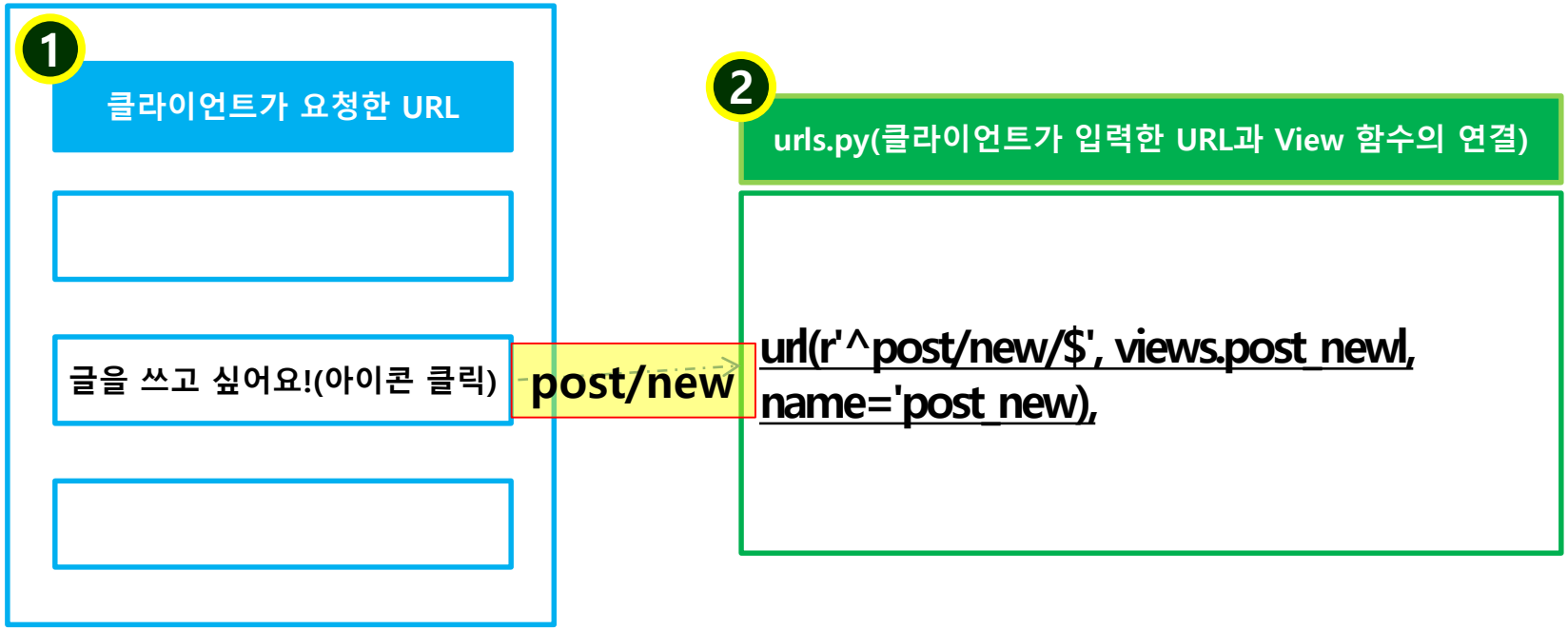
이번 시간에 하고 싶은 것

새로운 글쓰기

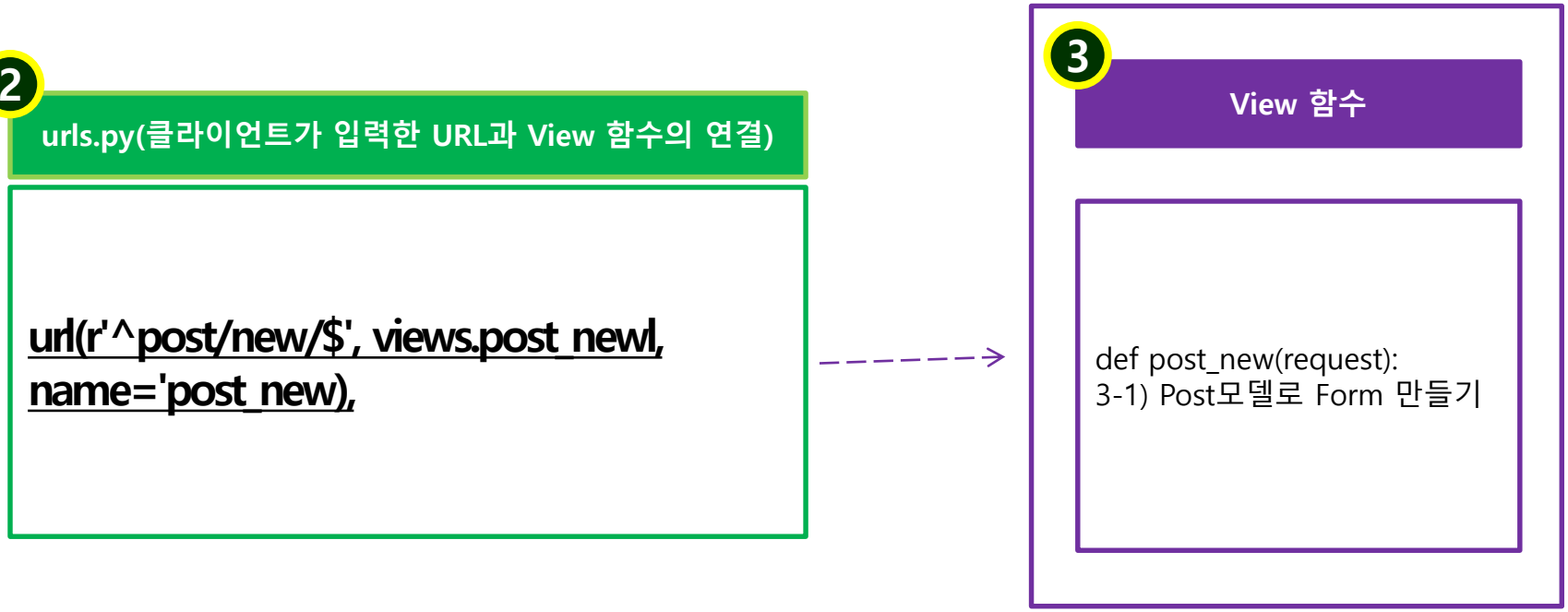

127.0.0.1:8000/post/new/
127.0.0.1:8000/post/new/
Welcome to SunrinPress! :)
New post
Title:
Text:

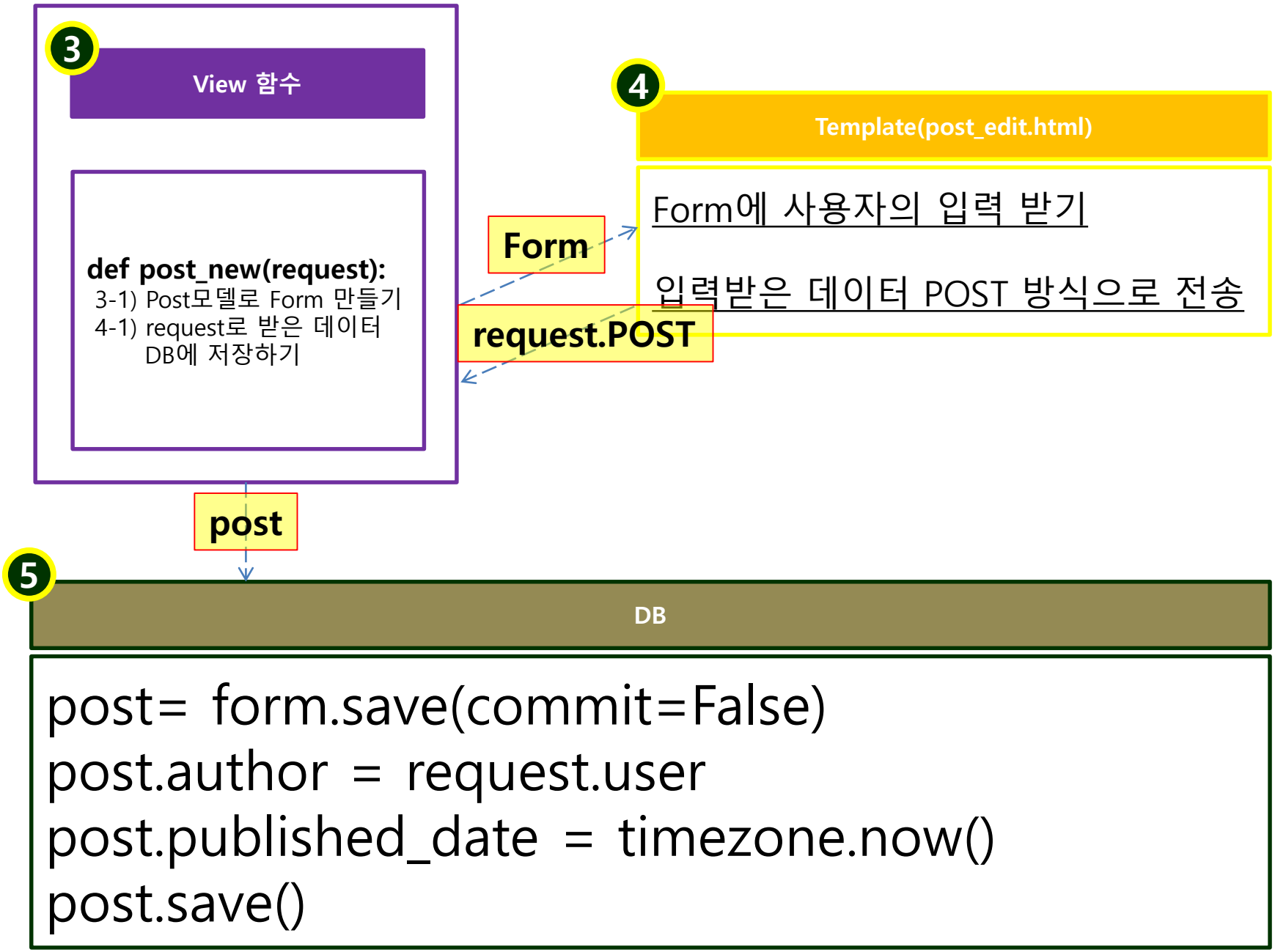
우리가 무엇을 해야 할까?

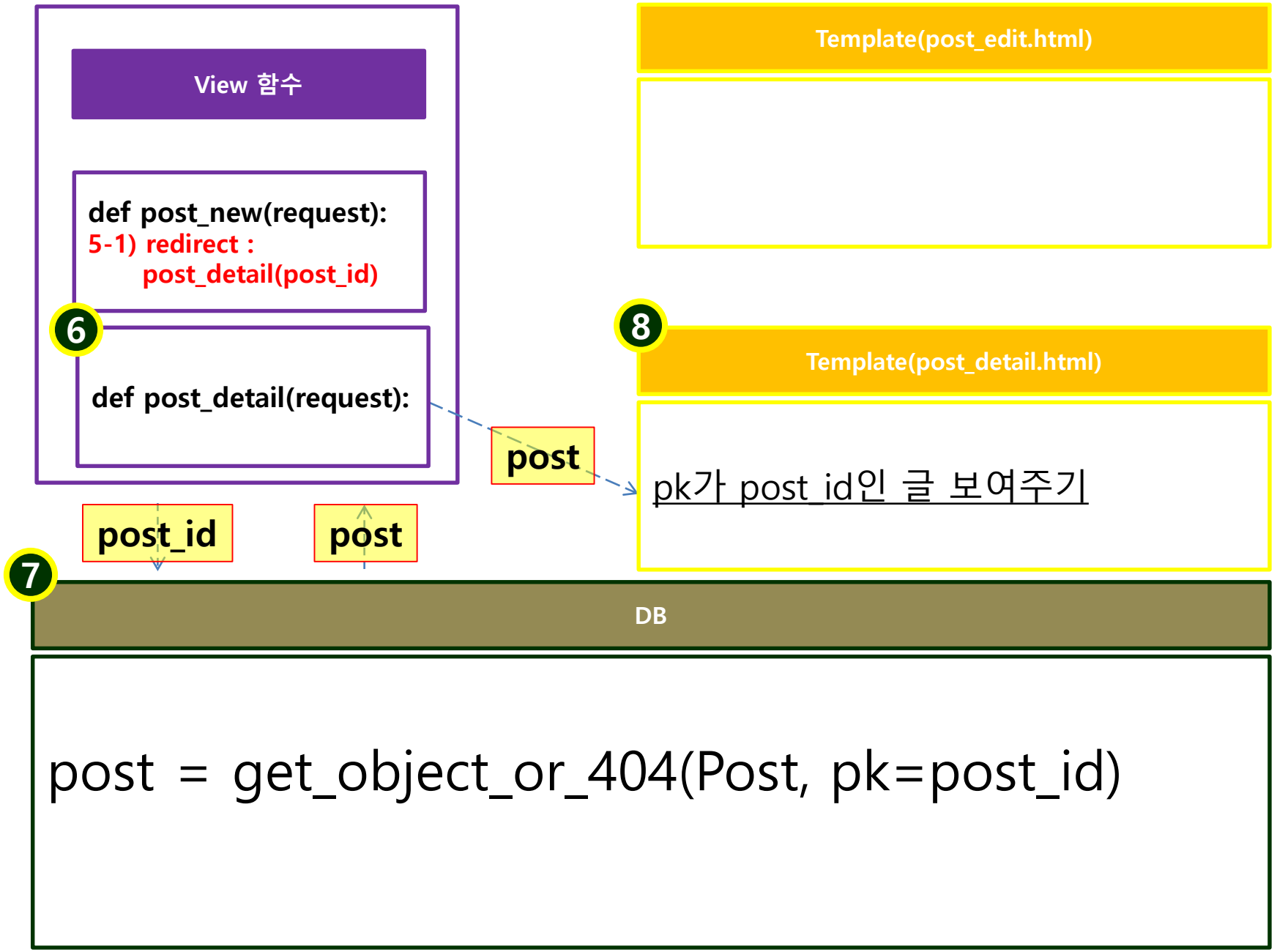
클라이언트가 글쓰기 아이콘을 클릭했을 때



URL Dispatcher에서 View 함수 호출




3
View 함수
def post_new(request):
3-1) Post모델로 Form 만들기
4-1) request로 받은 데이터
DB에 저장하기
Form
request.POST
4
Template(post_edit.html)
Form에 사용자의 입력 받기
입력받은 데이터 POST 방식으로 전송
post
5
DB
post= form.save(commit=False)
post.author = request.user
post.published_date = timezone.now()
post.save()


View 함수
def post_new(request):
5-1) redirect :
post_detail(post_id)
6
def post_detail(request):
post
Template(post_edit.html)
8
Template(post_detail.html)
pk가 post_id인 글 보여주기
post_id
post
7
DB
post = get_object_or_404(Post, pk=post_id)

그래서 마지막 장면이 이 화면!


그래서 지금부터 만들어봅시다!
django

post_list.html 수정 : 글쓰기 버튼 만들기

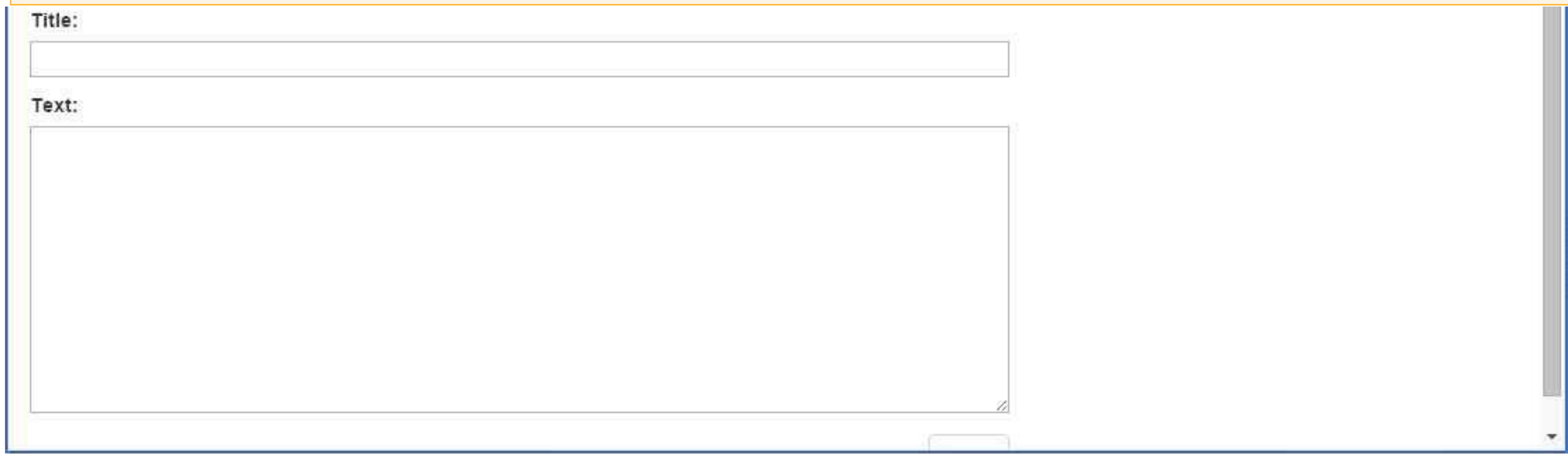
```
<div id="flip"class="page-header">
        <a href="{% url "post_new" %}"class="top-menu">
        <span class="glyphicon glyphicon-plus"></span></a>
        <h1><a href="/">Welcome to SunrinPress :)</a></h1>
</div>
```

Form 만들기

firstproject ▶ blog ▶ forms.py

```
1  from django import forms
2  from .models import Post
3
4  class PostForm(forms.ModelForm):
5      class Meta:
6          model = Post
7          fields = ('title', 'text',)
```

Colored by Color Scripter



URL 패턴, 이름 만들기

```
# firstproject ▶ blog ▶ urls.py
```

```
url(r'^post/new/$', views.post_new, name='post_new'),
```

View 함수 만들기 : post_new

```
# firstproject ▶ blog ▶ views.py
from .forms import PostForm
(중략)
def post_new(request):
    form = PostForm()
    return render(request, 'blog/post_edit.html', {'form': form})
```

post_edit.html(post_list.html 복붙해서 수정)

```
<h1>New post</h1>
<form method="POST" class="post-form">
    {% csrf_token %}
    {{ form.as_p }}
    <button type="submit" class="save btn btn-default">Save</button>
</form>
```

POST method로 받은 request에 대한 처리

```
# firstproject ▶ blog ▶ views.py
```

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

```
def post_new(request):
    if request.method =="POST":
        form = PostForm(request.POST)
        if form.is_valid():
            post = form.save(commit=False)
            post.author = request.user
            post.published_date = timezone.now()
            post.save()
            return redirect('blog.views.post_detail', post_id=post.pk)
    else:
        form = PostForm()
    return render(request, 'blog/post_edit.html', {'form': form})
```

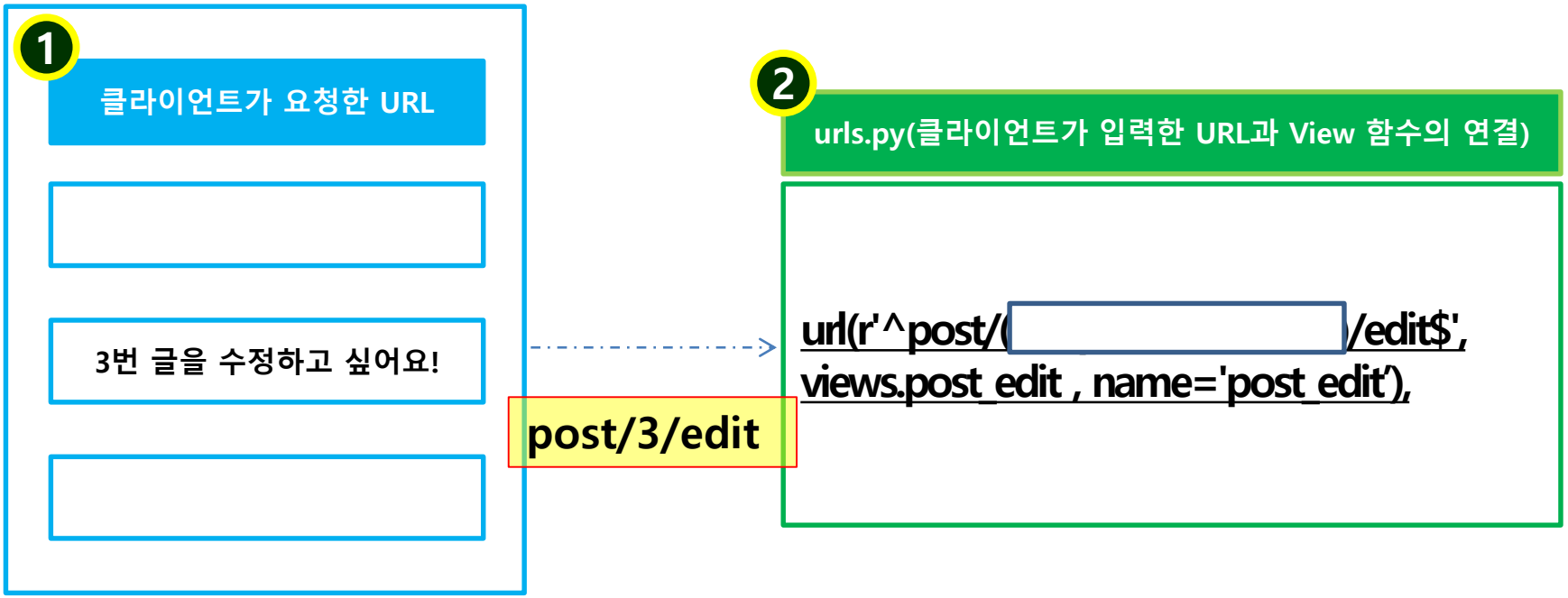

다음은 글 불러와서 수정하기!
django

post_detail.html에서 연필 모양 아이콘 클릭!

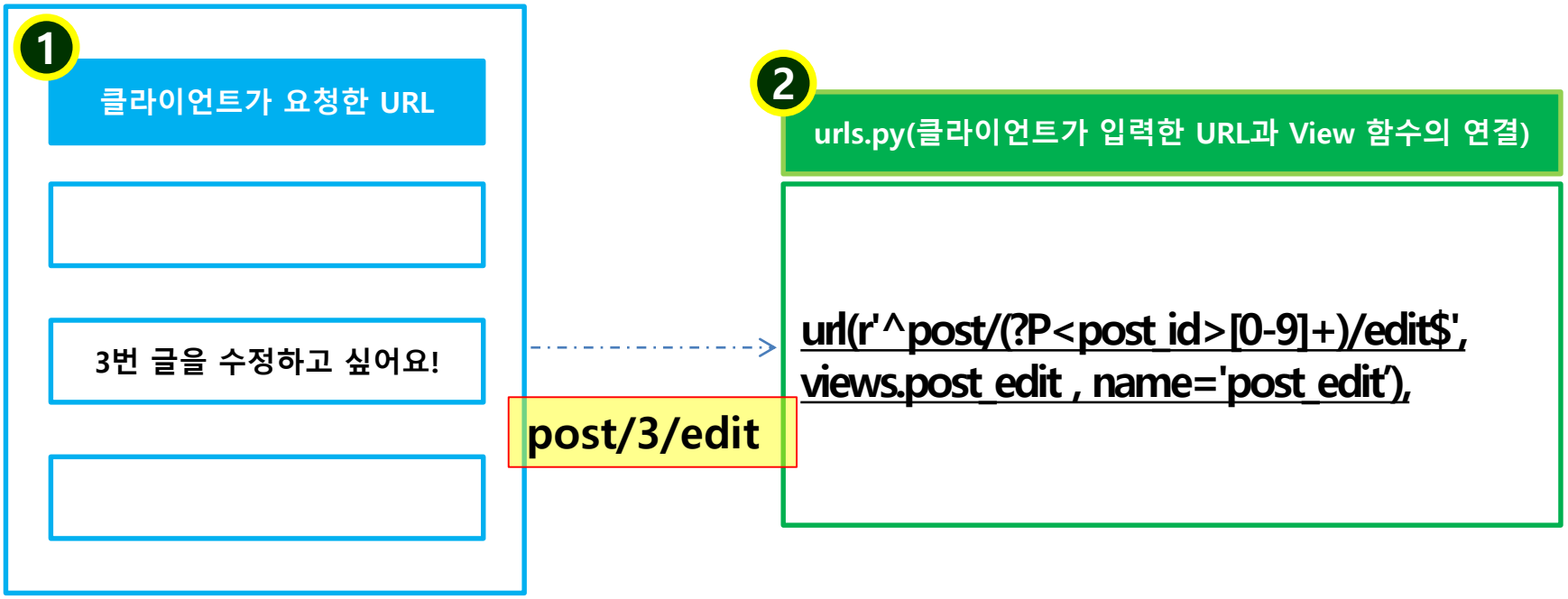
이렇게 클릭한 글의 내용이 자동으로 뙇!!!

우리가 무엇을 해야 할까?

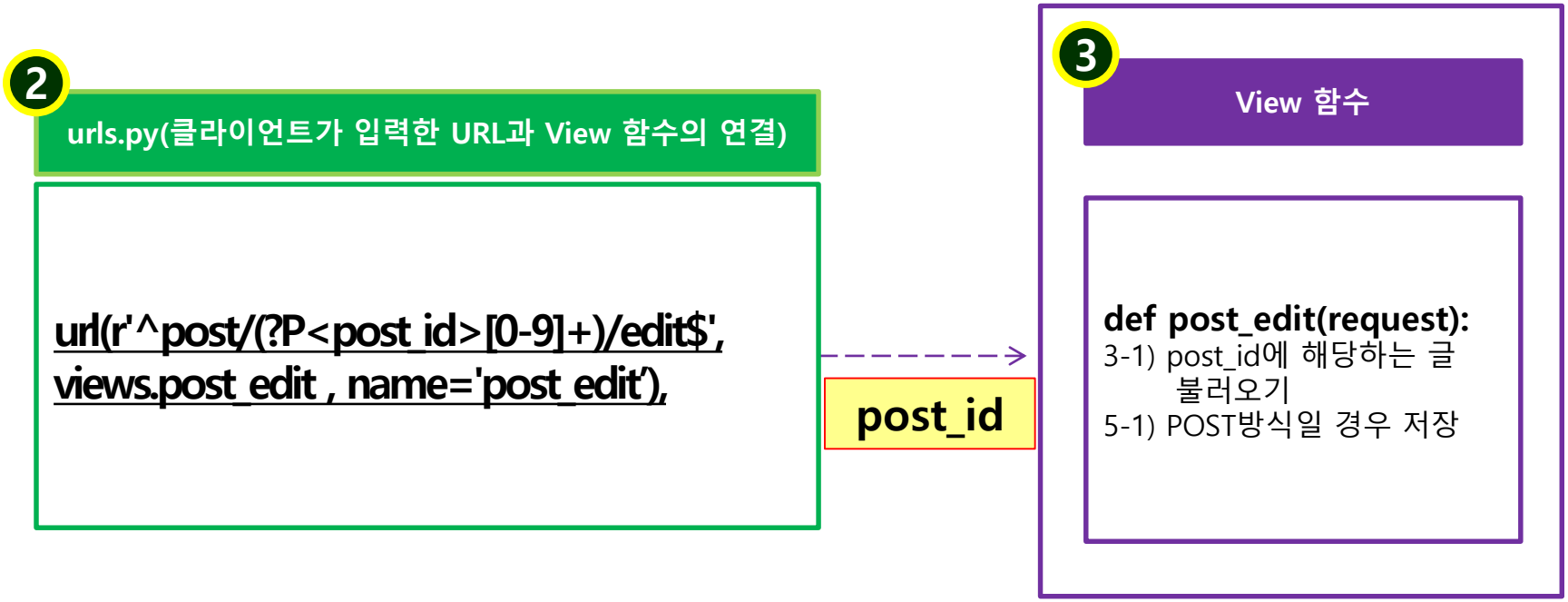
클라이언트가 글 수정 아이콘을 클릭했을 때

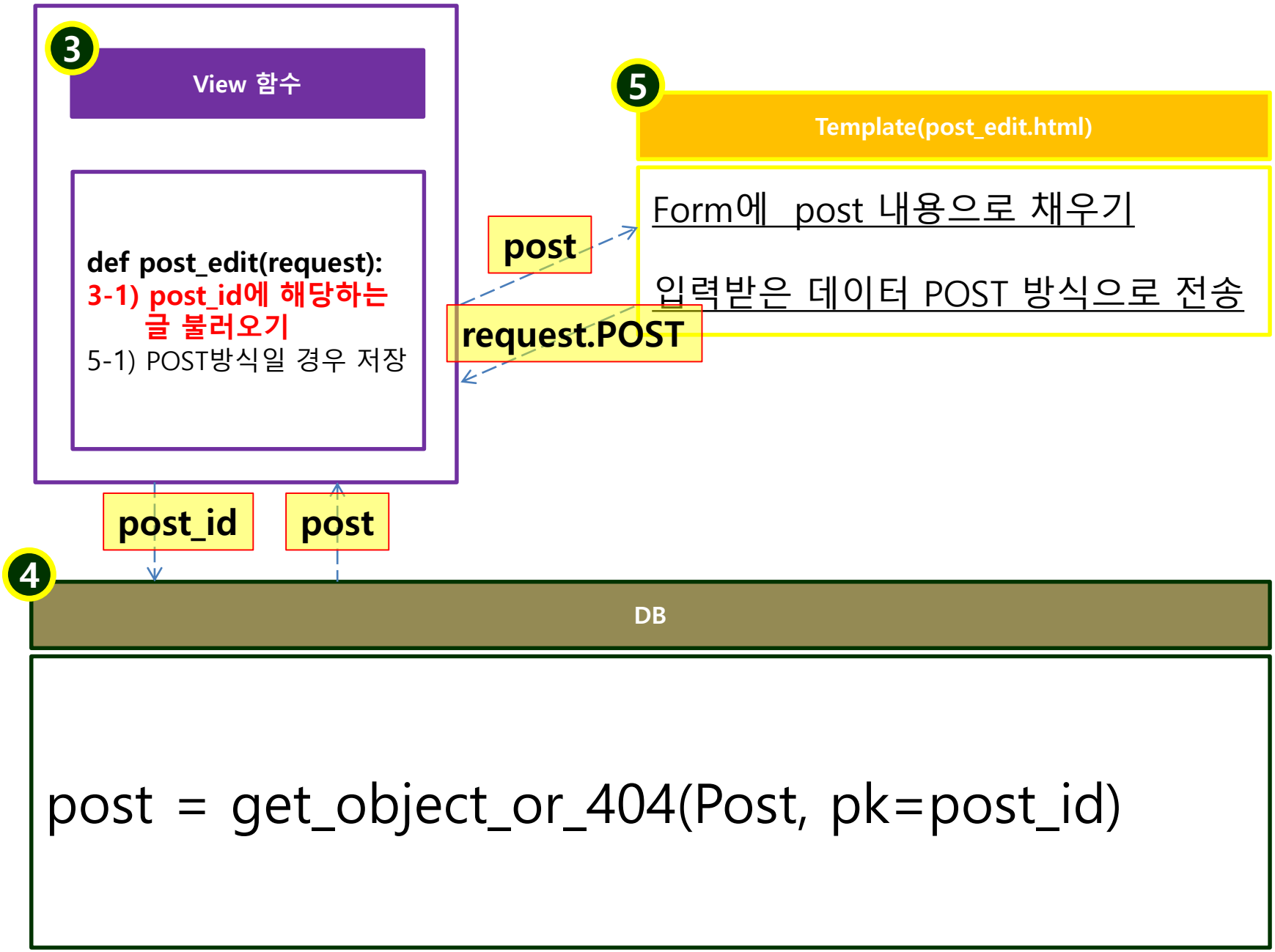


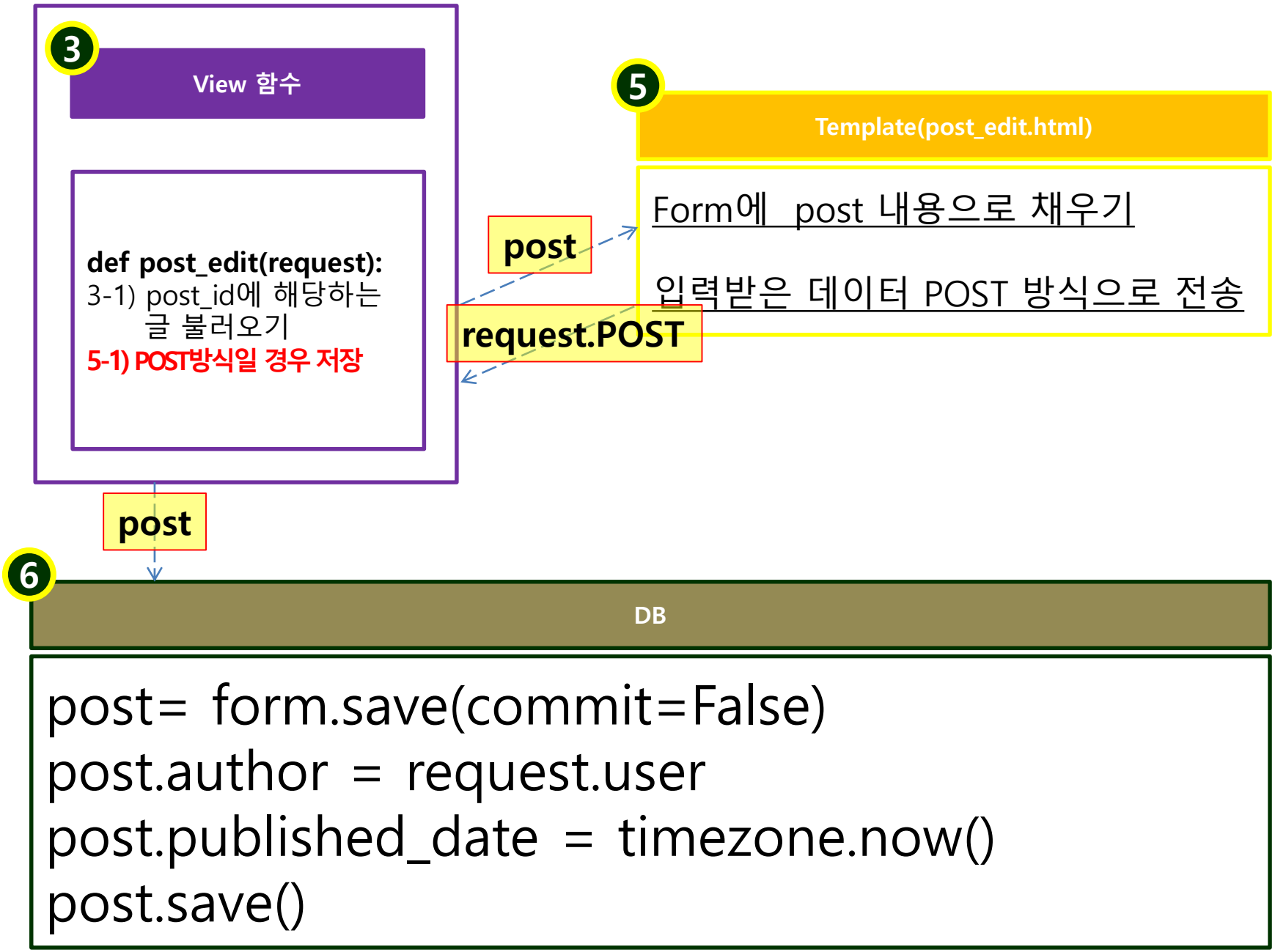
클라이언트가 글 수정 아이콘을 클릭했을 때



URL Dispatcher에서 View 함수 호출




3
View 함수
def post_edit(request):
3-1) post_id에 해당하는 글 불러오기
5-1) POST방식일 경우 저장
5
Template(post_edit.html)
Form에 post 내용으로 채우기
입력받은 데이터 POST 방식으로 전송
post
request.POST
post_id
post
4
DB
post = get_object_or_404(Post, pk=post_id)


3
View 함수
def post_edit(request):
3-1) post_id에 해당하는 글 불러오기
5-1) POST방식일 경우 저장
5
Template(post_edit.html)
Form에 post 내용으로 채우기
입력받은 데이터 POST 방식으로 전송
post
request.POST
post
6
DB
post= form.save(commit=False)
post.author = request.user
post.published_date = timezone.now()
post.save()


그래서 지금부터 만들어봅시다!
django

post_detail.html 수정 : 글 수정 버튼 만들기

firstproject ▶ blog ▶ templates ▶ blog ▶ post_detail.html

```
1  {% extends "blog/base.html" %}
2  {% block content %}
3      <div class="post">
4          {% if post.published_date %}
5              <div class="date">
6                  {{ post.published_date }}
7              </div>
8          {% endif %}
9          <a class="btn btn-default" href="{% url "post_edit" post_id=post.pk %}">
10         <span class="glyphicon glyphicon-pencil"></span></a>
11         <h1>{{ post.title }}</h1>
12         <p>{{ post.text }}</p>
13     </div>
14 {% endblock content %}
```

url 패턴과 view 연결 만들기

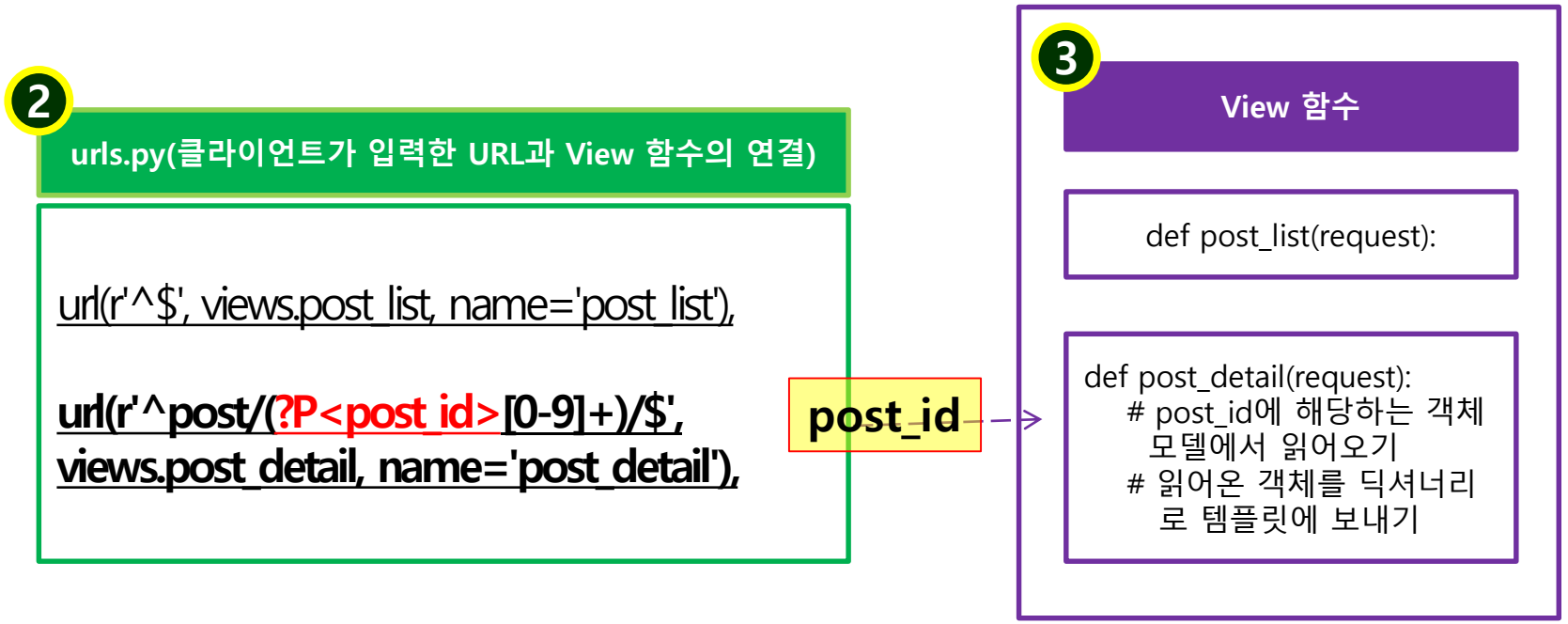
```
# firstproject ▶ blog ▶ urls.py
```

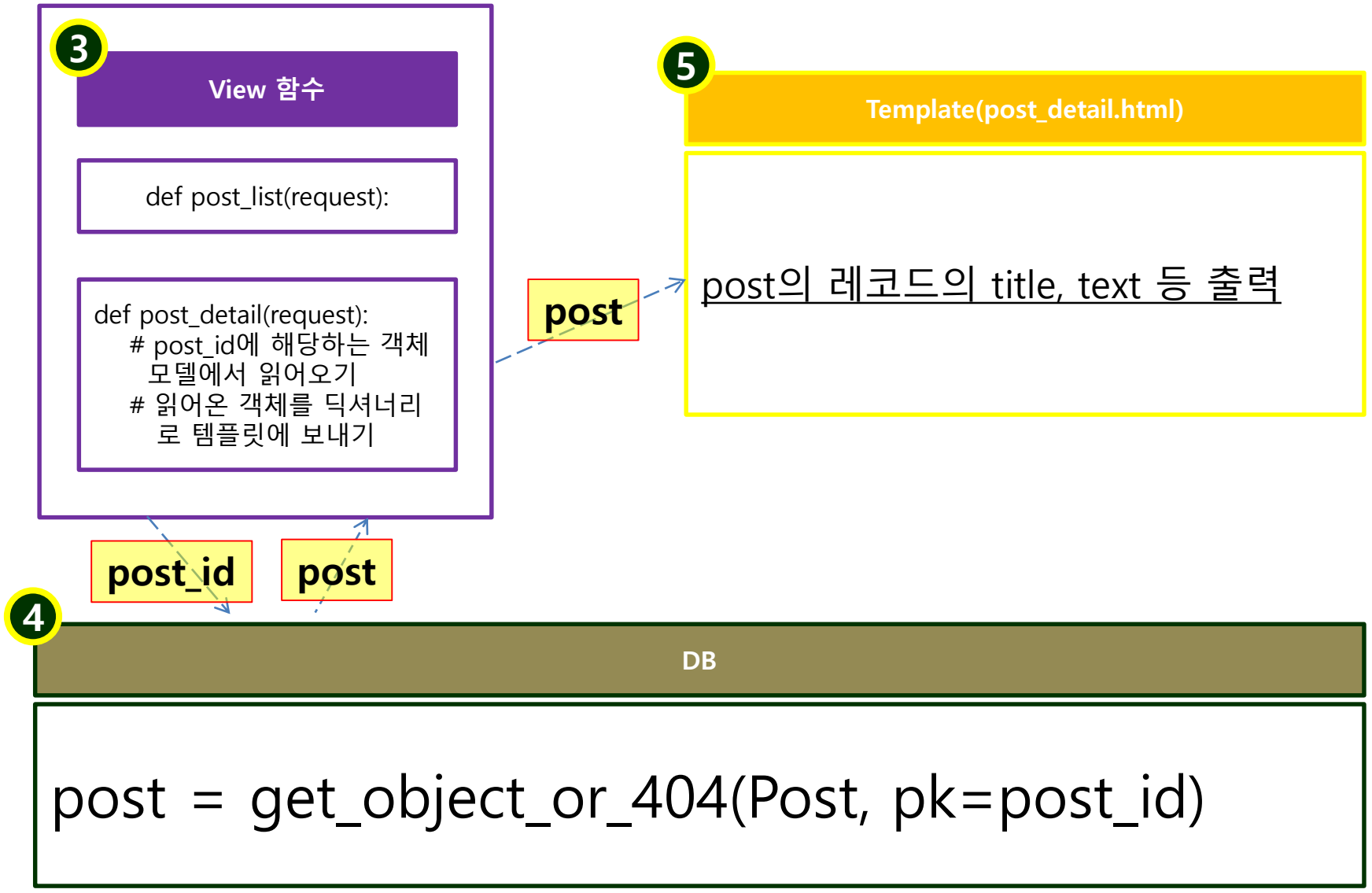
```
url(r'^post/(?P<post_id>[0-9]+)/edit/$', views.post_edit, name='post_edit'),
```

firstproject ▶ blog ▶ views.py

```
def post_edit(request, post_id):
    post = get_object_or_404(Post, pk=post_id)
    if request.method =="POST":
        form = PostForm(request.POST)
        if form.is_valid():
            post = form.save(commit=False)
            post.author = request.user
            post.published_date = timezone.now()
            post.save()
            return redirect('blog.views.post_detail', post_id=post.pk)
    else:
        form = PostForm(instance=post)
    return render(request, 'blog/post_edit.html', {'form': form})
```

URL Dispatcher에서 패턴 분석 : post_id 추출




3
View 함수
def post_list(request):
def post_detail(request):
# post_id에 해당하는 객체 모델에서 읽어오기
# 읽어온 객체를 딕셔너리로 템플릿에 보내기
post
5
Template(post_detail.html)
post의 레코드의 title, text 등 출력
post_id
post
4
DB
post = get_object_or_404(Post, pk=post_id)