

멋쟁이사자처럼

인스타그램 4th 강의

세화가 강의한다

POST CUD (Create / Update / Delete)

COMMENT C (Create)

COMMENT UD (Update / Delete) // 과제입니당^__-^

시작해볼까요?

POST CUD (Create / Update / Delete)

COMMENT C (Create)

COMMENT UD (Update / Delete) // 과제입니당^___^

POST Create

POST Update

POST Delete

POST Create
form 생성
url 생성
view 생성
template 생성 / 수정

insta
└ forms.py

```
EXPLORER

INSTAGRAM_SEHWA
├── insta
│   ├── __pycache__
│   ├── migrations
│   ├── __init__.py
│   ├── admin.py
│   ├── apps.py
│   ├── forms.py
│   ├── models.py
│   ├── tests.py
│   ├── urls.py
│   └── views.py
├── instagram
├── media
└── static

insta > forms.py
1  from django import forms
2  from .models import Post, Comment
3
4
5  class PostForm(forms.ModelForm):
6      class Meta:
7          model = Post
8          fields = ('caption', 'image')
9          widgets = {
10             'caption': forms.Textarea()
11         }
12
13  class CommentForm(forms.ModelForm):
14      class Meta:
15          model = Comment
16          fields = ('content',)
```

forms.py 추가

POST Create
form 생성
url 생성
view 생성
template 생성 / 수정

EXPLORER

INSTAGRAM_SEHWA

insta

> __pycache__

> migrations

__init__.py

admin.py

apps.py

forms.py

models.py

tests.py

urls.py

views.py

insta > urls.py

1

2

3

4

5

6

7

8

9

10

11

12

13

from django.urls import path

from . import views

App_name = 'insta'

urlpatterns = [

path('', views.main, name='main'),

path('<int:post_pk>/like/', views.like, name='like'),

path('new/', views.new, name='new'),

]

insta
└─ urls.py



POST Create
form 생성
url 생성
view 생성
template 생성 / 수정

EXPLORER

INSTAGRAM_SEHWA

- insta
 - __pycache__
 - migrations
 - __init__.py
 - admin.py
 - apps.py
 - forms.py
 - models.py
 - tests.py
 - urls.py
 - views.py
- instagram
- media
- static
- templates
- .gitignore
- db.sqlite3

views.py

```
1 import operator
2 from django.shortcuts import render, get_object_or_404, redirect
3 from .models import Post, Comment, User, Like
4 from .forms import PostForm, CommentForm
5
6
7 def new(request):
8     if request.method == "POST":
9         form = PostForm(request.POST, request.FILES)
10        if form.is_valid():
11            post = form.save(commit=False)
12            post.user = request.user
13            post.save()
14            return redirect('main')
15        else:
16            form = PostForm()
17        return render(request, 'insta/post_create.html', {
18            'form': form
19        })
```

insta
└ views.py

insta/views.py
post_create 함수 추가

views.py
def new

('main')
main.html로 가랏

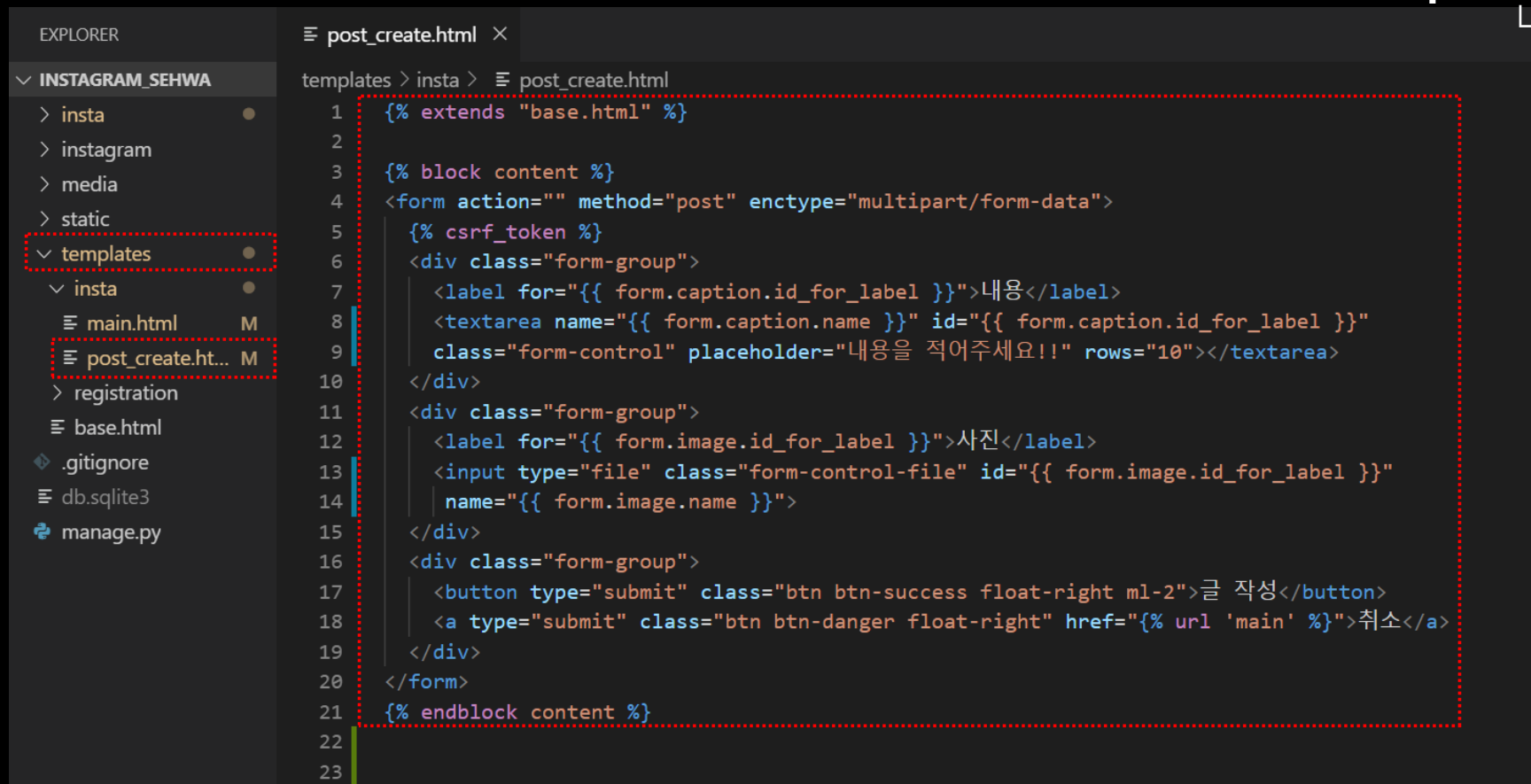
insta
└ templates
└ insta
└ main.html

(request, 'insta/post_create.html')
insta/post_create.html로 가랏

insta
└ templates
└ insta
└ post_create.html

POST Create
form 생성
url 생성
view 생성
template 생성 / 수정

templates/insta post_new.html

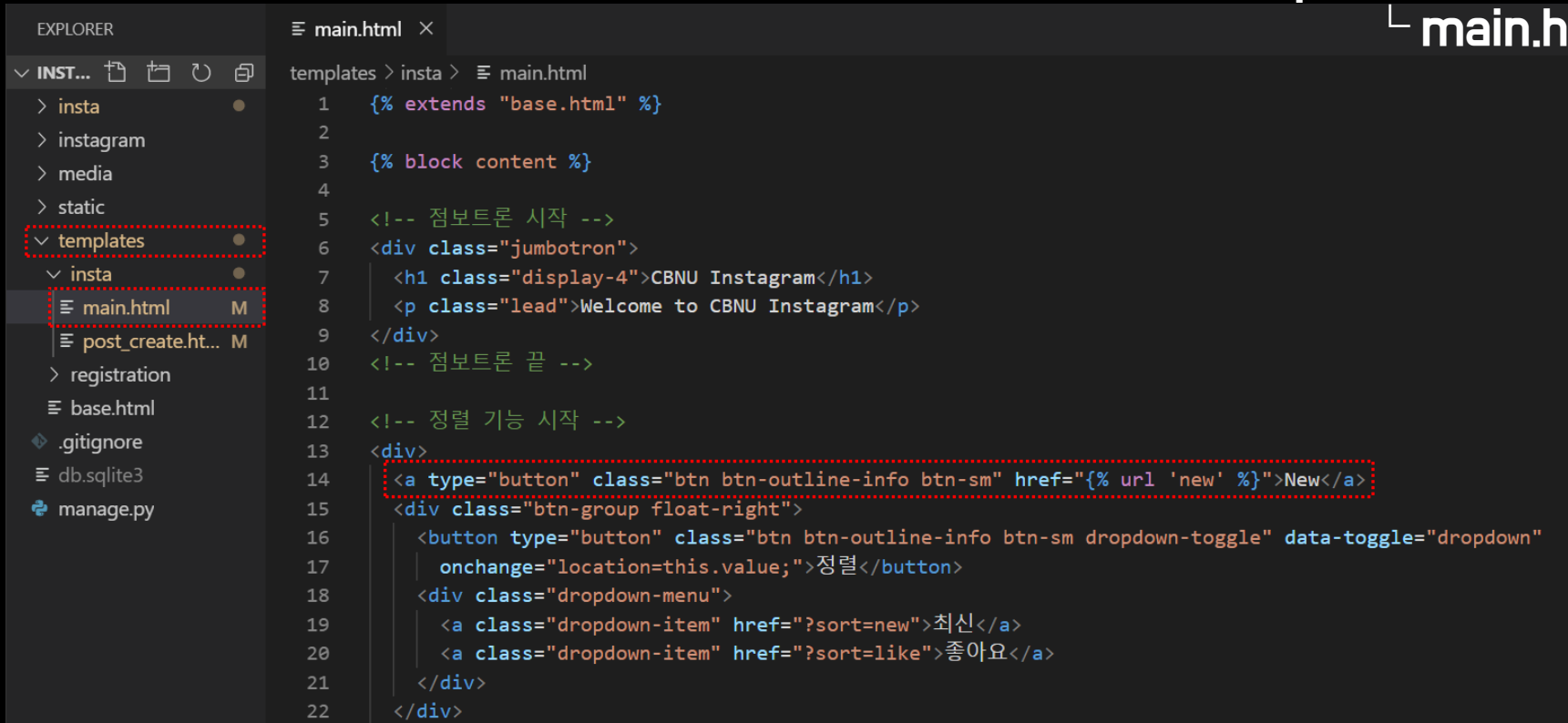


The screenshot shows a code editor with a file explorer on the left and a code editor on the right. The file explorer shows a project structure with a folder named 'INSTAGRAM_SEHWA' containing subfolders 'insta', 'instagram', 'media', 'static', and 'templates'. The 'templates' folder is expanded, showing 'insta' and 'main.html'. The 'post_create.html' file is selected. The code editor shows the content of 'post_create.html', which is a Django template. The code is enclosed in a red dashed box. The code includes a form for creating a new post, with fields for caption and image, and buttons for '글 작성' (Write Post) and '취소' (Cancel).

```
1 {% extends "base.html" %}
2
3 {% block content %}
4 <form action="" method="post" enctype="multipart/form-data">
5     {% csrf_token %}
6     <div class="form-group">
7         <label for="{{ form.caption.id_for_label }}">내용</label>
8         <textarea name="{{ form.caption.name }}" id="{{ form.caption.id_for_label }}"
9             class="form-control" placeholder="내용을 적어주세요!!" rows="10"></textarea>
10     </div>
11     <div class="form-group">
12         <label for="{{ form.image.id_for_label }}">사진</label>
13         <input type="file" class="form-control-file" id="{{ form.image.id_for_label }}"
14             name="{{ form.image.name }}">
15     </div>
16     <div class="form-group">
17         <button type="submit" class="btn btn-success float-right m1-2">글 작성</button>
18         <a type="submit" class="btn btn-danger float-right" href="{% url 'main' %}">취소</a>
19     </div>
20 </form>
21 {% endblock content %}
22
23
```

post_create.html 추가

templates/insta
└─ main.html



The screenshot shows a code editor with a file explorer on the left and a code editor on the right. The file explorer shows a project structure with folders like 'insta', 'instagram', 'media', 'static', 'templates', and 'registration'. The 'templates' folder is expanded, showing 'insta' and 'main.html'. The 'main.html' file is selected and highlighted with a red dashed box. The code editor shows the content of 'main.html', which is a Django template. The code includes a Jinja2-style 'extends' statement, a 'block content' block, and a 'jumbotron' section. The 'jumbotron' section contains a heading, a paragraph, and a 'div' with a 'button' and a 'dropdown-menu'. The 'button' is highlighted with a red dashed box. The code is as follows:

```
1 {% extends "base.html" %}
2
3 {% block content %}
4
5 <!-- 점보트론 시작 -->
6 <div class="jumbotron">
7   <h1 class="display-4">CBNU Instagram</h1>
8   <p class="lead">Welcome to CBNU Instagram</p>
9 </div>
10 <!-- 점보트론 끝 -->
11
12 <!-- 정렬 기능 시작 -->
13 <div>
14   <a type="button" class="btn btn-outline-info btn-sm" href="{% url 'new' %}">New</a>
15   <div class="btn-group float-right">
16     <button type="button" class="btn btn-outline-info btn-sm dropdown-toggle" data-toggle="dropdown"
17       onchange="location=this.value;">정렬</button>
18     <div class="dropdown-menu">
19       <a class="dropdown-item" href="?sort=new">최신</a>
20       <a class="dropdown-item" href="?sort=like">좋아요</a>
21     </div>
22   </div>
23 </div>
```

main.html 수정

POST Create

POST Update

POST Delete

POST Update
url 생성
view 생성
template 생성 / 수정


```
EXPLORER
INST...
└ insta
  ├── __pycache__
  ├── migrations
  ├── __init__.py
  ├── admin.py
  ├── apps.py
  ├── forms.py
  ├── models.py
  ├── tests.py
  ├── urls.py
  └── views.py

insta > urls.py
1 from django.urls import path
2 from . import views
3
4 App_name = 'insta'
5
6 urlpatterns = [
7     path('', views.main, name='main'),
8     path('<int:post_pk>/like/', views.like, name='like'),
9     path('new/', views.new, name='new'),
10    path('<int:post_pk>/edit/', views.edit, name='edit'),
11 ]
12
13
```

insta
└ urls.py



POST Update
url 생성
view 생성
template 생성 / 수정

insta
└─ views.py

```
57  
58 def edit(request, post_pk):  
59     post = get_object_or_404(Post, pk = post_pk)  
60     if request.method == "POST":  
61         form = PostForm(request.POST, request.FILES, instance = post)  
62         if form.is_valid():  
63             post = form.save(commit=False)  
64             post.user = request.user  
65             post.save()  
66             return redirect('main')  
67     else:  
68         form = PostForm(instance = post)  
69     return render(request, 'insta/post_edit.html', {  
70         'form': form  
71     })  
72
```

insta/views.py
edit 함수 추가

views.py
def edit

('main')
main.html로 가랏

templates
└─ insta
└─ main.html

(request, 'insta/post_edit.html')
insta/post_edit.html로 가랏

templates
└─ insta
└─ post_edit.html

POST Update
url 생성
view 생성
template 생성 / 수정

templates/insta └ post_edit.html

EXPLORER

INSTAGRAM_SEHWA

- > insta
- > instagram
- > media
- > static
- templates
 - insta
 - main.html M
 - post_create.ht... M
 - post_edit.html U
 - registration
- base.html
- .gitignore
- db.sqlite3
- manage.py

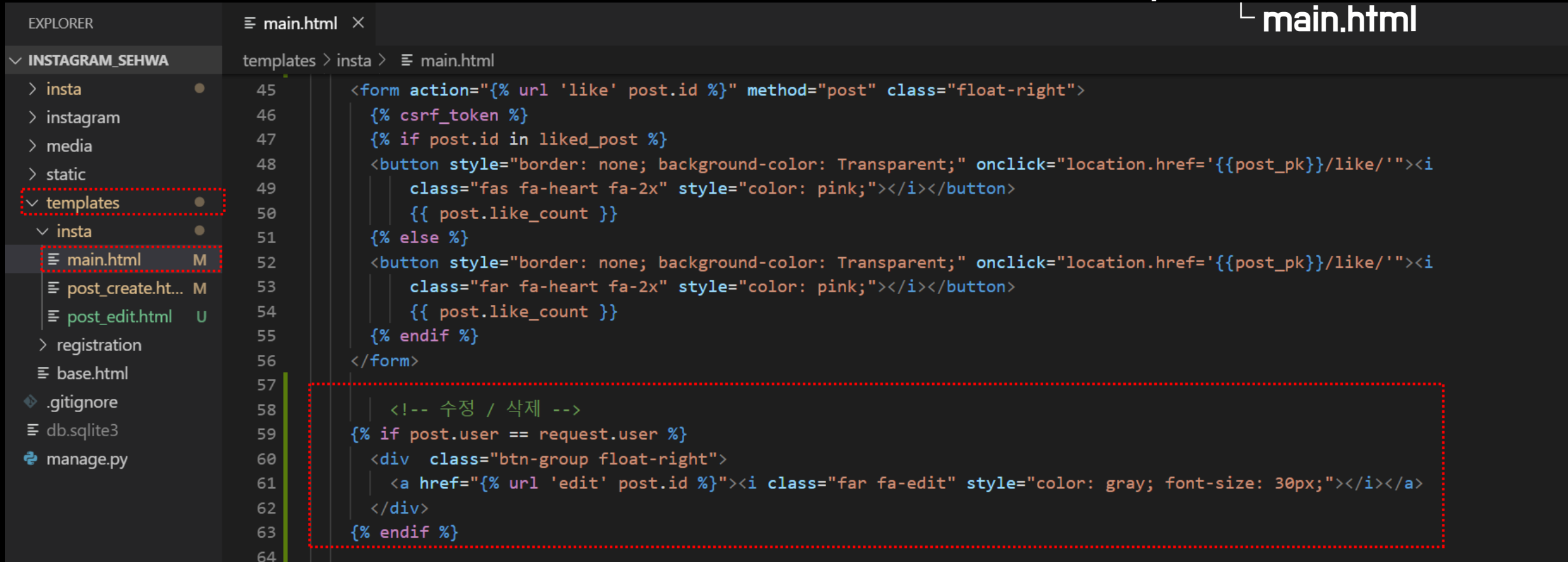
post_edit.html

templates > insta > post_edit.html

```
1 {% extends "base.html" %}
2
3 {% block content %}
4 <form action="" method="post" enctype="multipart/form-data">
5     {% csrf_token %}
6     <div class="form-group">
7         {{ form.as_p }}
8     </div>
9     <div class="form-group">
10         <button type="submit" class="btn btn-success float-right m1-2">글 수정</button>
11         <a type="submit" class="btn btn-danger float-right" href="{% url 'main' %}">취소</a>
12     </div>
13 </form>
14 {% endblock content %}
15
16
17
```

post_edit.html 추가

templates/insta main.html



```
45 <form action="{% url 'like' post.id %}" method="post" class="float-right">
46     {% csrf_token %}
47     {% if post.id in liked_post %}
48     <button style="border: none; background-color: Transparent;" onclick="location.href='{{post_pk}}/like/'"><i
49         class="fas fa-heart fa-2x" style="color: pink;"></i></button>
50         {{ post.like_count }}
51     {% else %}
52     <button style="border: none; background-color: Transparent;" onclick="location.href='{{post_pk}}/like/'"><i
53         class="far fa-heart fa-2x" style="color: pink;"></i></button>
54         {{ post.like_count }}
55     {% endif %}
56 </form>
57 <!-- 수정 / 삭제 -->
58 {% if post.user == request.user %}
59 <div class="btn-group float-right">
60     <a href="{% url 'edit' post.id %}"><i class="far fa-edit" style="color: gray; font-size: 30px;"></i></a>
61 </div>
62 {% endif %}
63
```

main.html 수정

POST Create

POST Update

POST Delete

POST Delete

url 생성

view 생성

template 수정

EXPLORER

INSTAGRAM_SEHWA

insta

__pycache__

migrations

__init__.py

admin.py

apps.py

forms.py

models.py

tests.py

urls.py

views.py

instagram

urls.py

insta > urls.py

1 from django.urls import path

2 from . import views

3

4 App_name = 'insta'

5

6 urlpatterns = [

7 path('', views.main, name='main'),

8 path('<int:post_pk>/like/', views.like, name='like'),

9 path('new/', views.new, name='new'),

10 path('<int:post_pk>/edit/', views.edit, name='edit'),

11 path('<int:post_pk>/delete/', views.delete, name='delete'),

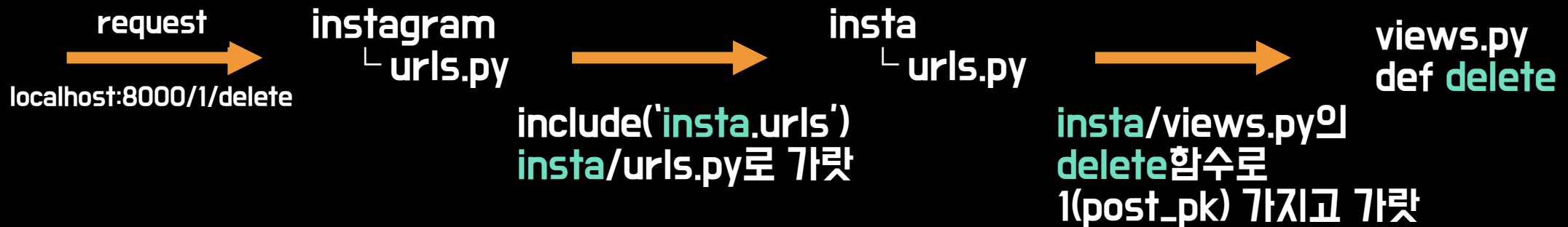
12]

13

14

insta

└ urls.py



POST Delete

url 생성

view 생성

template 수정

insta
└─ views.py

```
EXPLORER
└─ INSTAGRAM_SEHWA
  └─ insta
    ├── __pycache__
    ├── migrations
    ├── __init__.py
    ├── admin.py
    ├── apps.py
    ├── forms.py
    ├── models.py
    ├── tests.py
    ├── urls.py
    └── views.py
  ├── instagram
  ├── media
  ├── static
  └── templates
    └─ insta
      ├── main.html
      └─ post_create.ht...

views.py
58 def edit(request, post_pk):
59     post = get_object_or_404(Post, pk = post_pk)
60     if request.method == "POST":
61         form = PostForm(request.POST, request.FILES, instance = post)
62         if form.is_valid():
63             post = form.save(commit=False)
64             post.user = request.user
65             post.save()
66             return redirect('main')
67     else:
68         form = PostForm(instance = post)
69     return render(request, 'insta/post_edit.html', {
70         'form': form
71     })
72
73
74 def delete(request, post_pk):
75     post = get_object_or_404(Post, pk = post_pk)
76     post.delete()
77     return redirect('main')
78
```

insta/views.py
delete 함수 추가

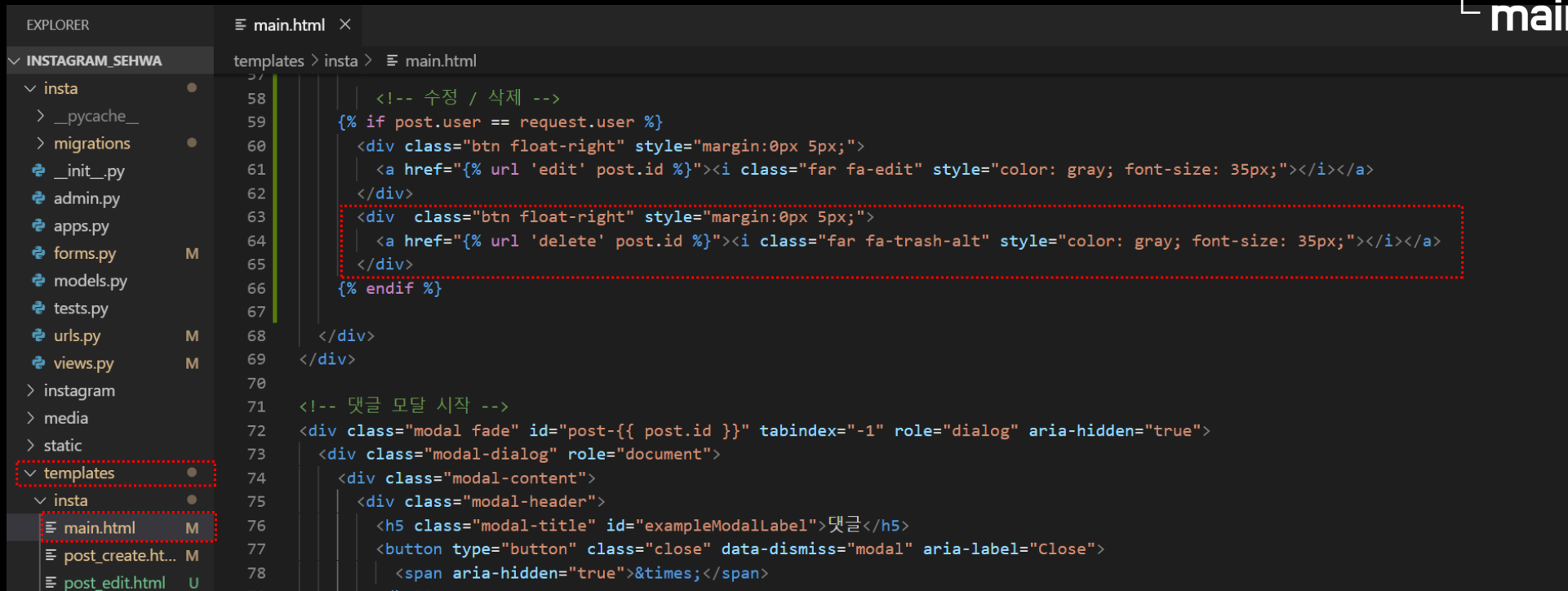
views.py
def delete

→
redirect('main')
insta/main.html로 가랏

templates
└─ insta
└─ main.html

POST Delete
url 생성
view 생성
template 수정

templates/insta main.html



```
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79
```

```
<!-- 수정 / 삭제 -->  
{% if post.user == request.user %}  
<div class="btn float-right" style="margin:0px 5px;">  
  <a href="{% url 'edit' post.id %}"><i class="far fa-edit" style="color: gray; font-size: 35px;"></i></a>  
</div>  
<div class="btn float-right" style="margin:0px 5px;">  
  <a href="{% url 'delete' post.id %}"><i class="far fa-trash-alt" style="color: gray; font-size: 35px;"></i></a>  
</div>  
{% endif %}  
  
</div>  
</div>  
  
<!-- 댓글 모달 시작 -->  
<div class="modal fade" id="post-{{ post.id }}" tabindex="-1" role="dialog" aria-hidden="true">  
  <div class="modal-dialog" role="document">  
    <div class="modal-content">  
      <div class="modal-header">  
        <h5 class="modal-title" id="exampleModallabel">댓글</h5>  
        <button type="button" class="close" data-dismiss="modal" aria-label="Close">  
          <span aria-hidden="true">&times;</span>  
      </div>
```

main.html 수정

POST CUD (Create / Update / Delete)

COMMENT C (Create)

COMMENT UD (Update / Delete) // 과제입니당^___^

COMMENT Create

COMMENT Create

url 생성

view 생성 / 수정

template 수정


```
EXPLORER
INSTAGRAM_SEHWA
└ insta
  ├── __pycache__
  ├── migrations
  ├── __init__.py
  ├── admin.py
  ├── apps.py
  ├── forms.py
  ├── models.py
  ├── tests.py
  ├── urls.py
  └ views.py
instagram

urls.py
1 from django.urls import path
2 from . import views
3
4 App_name = 'insta'
5
6 urlpatterns = [
7     path('', views.main, name='main'),
8     path('<int:post_pk>/like/', views.like, name='like'),
9     path('new/', views.new, name='new'),
10    path('<int:post_pk>/edit/', views.edit, name='edit'),
11    path('<int:post_pk>/delete/', views.delete, name='delete'),
12    path('<int:post_pk>/comment/', views.create_comment, name='create_comment'),
13 ]
14
```

insta
└ urls.py



COMMENT Create

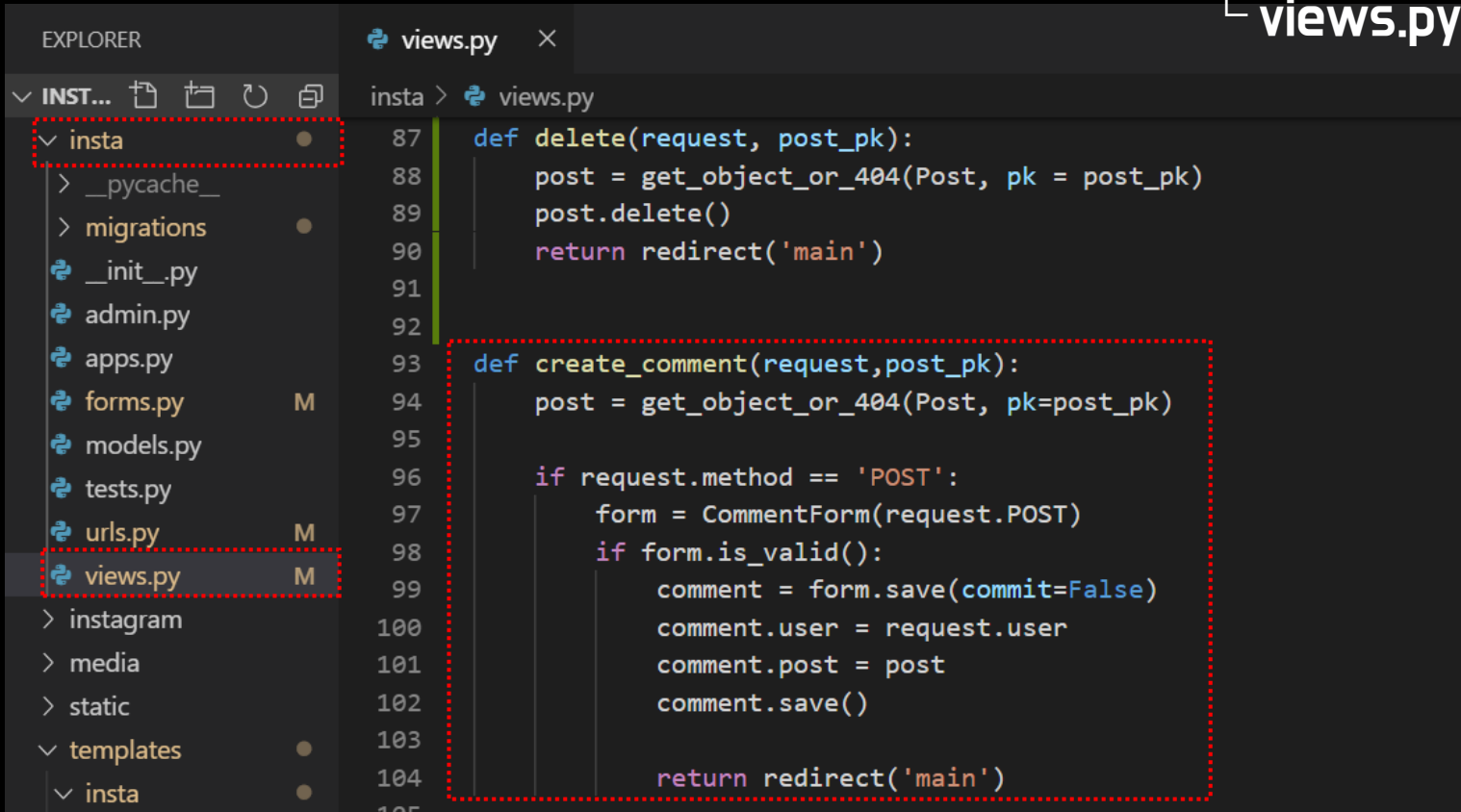
url 생성

view 생성 / 수정

template 수정

insta

└ **views.py**



insta/views.py create_comment 함수 추가

```
views.py
def create_comment
```

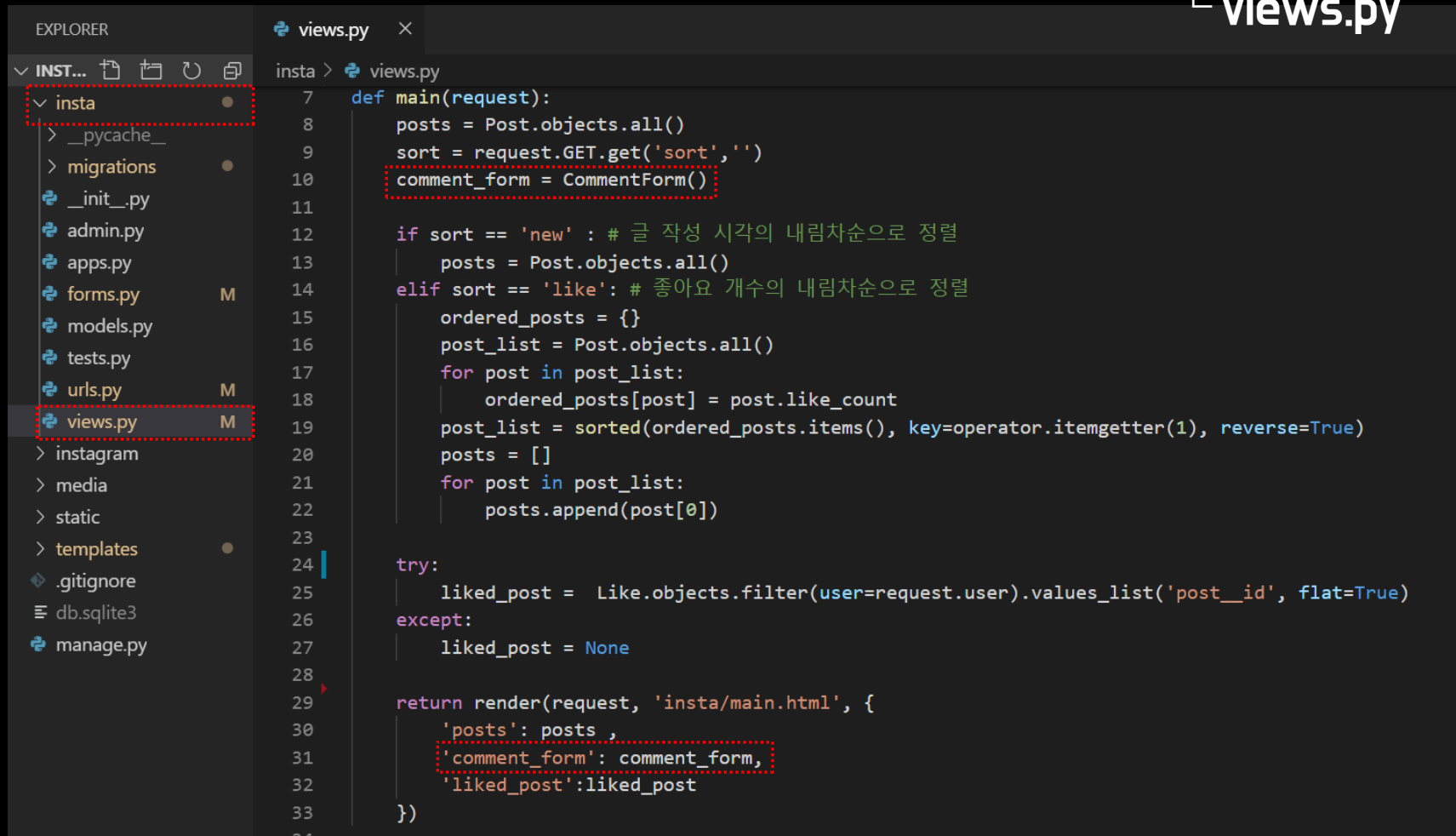


redirect('main')
insta/main.html로 가랏

```
templates
├── insta
│   └── main.html
```

insta

views.py



```
7 def main(request):
8     posts = Post.objects.all()
9     sort = request.GET.get('sort', '')
10    comment_form = CommentForm()
11
12    if sort == 'new' : # 글 작성 시각의 내림차순으로 정렬
13        posts = Post.objects.all()
14    elif sort == 'like': # 좋아요 개수의 내림차순으로 정렬
15        ordered_posts = {}
16        post_list = Post.objects.all()
17        for post in post_list:
18            ordered_posts[post] = post.like_count
19        post_list = sorted(ordered_posts.items(), key=operator.itemgetter(1), reverse=True)
20        posts = []
21        for post in post_list:
22            posts.append(post[0])
23
24    try:
25        liked_post = Like.objects.filter(user=request.user).values_list('post__id', flat=True)
26    except:
27        liked_post = None
28
29    return render(request, 'insta/main.html', {
30        'posts': posts ,
31        'comment_form': comment_form,
32        'liked_post': liked_post
33    })
```

insta/views.py
main 함수 수정

COMMENT Create

url 생성

view 생성 / 수정

template 수정

EXPLORER

INSTAGRAM_SEHWA

insta

instagram

media

static

templates

insta

main.html

post_create.ht...

post_edit.html

registration

base.html

.gitignore

db.sqlite3

manage.py

main.html

templates > insta > main.html

```

71 <!-- 댓글 모달 시작 -->
72 <div class="modal fade" id="post-{{ post.id }}" tabindex="-1" role="dialog" aria-hidden="true">
73   <div class="modal-dialog" role="document">
74     <div class="modal-content">
75       <div class="modal-header">
76         <h5 class="modal-title" id="exampleModalLabel">댓글</h5>
77         <button type="button" class="close" data-dismiss="modal" aria-label="Close">
78           <span aria-hidden="true">&times;</span>
79         </button>
80       </div>
81       <div class="modal-body">
82         {% for comment in post.comment_set.all %}
83         <blockquote class="blockquote">
84           
85           <div style="padding-inline-start:15%">
86             <p class="mb-0">{{ comment.content }}</p>
87             <footer class="blockquote-footer">{{ comment.user }}</footer>
88           </div>
89         </blockquote>
90       {% endfor %}
91     </div>
92     <div class="modal-footer" style="display: flex; justify-content: center;">
93       <form method='post' action="{% url 'create_comment' post.id %}">
94         {% csrf_token %}
95         <div style="display: flex; flex-direction: column;">
96           <textarea name="{{ comment_form.content.name }}" class="form-control" cols="50" placeholder="댓글을 남겨주세요~!"></textarea>
97           <div class="modal-footer">
98             <button type="button" class="btn btn-secondary" data-dismiss="modal">닫기</button>
99             <button type="submit" class="btn btn-primary">댓글 작성</button>
100           </div>
101         </div>
102       </form>
103     </div>
104   </div>
105 </div>
106 </div>
107 <!-- 댓글 모달 끝 -->

```

POST CUD (Create / Update / Delete)

COMMENT C (Create)

COMMENT UD (Update / Delete) // 과제입니당^__^

COMMENT Update

url 생성

view 생성

template 수정

COMMENT Delete

url 생성

view 생성

template 수정