

CRUD

# 〈CRUD〉

지난번에

CRUD 란

데이터를 처리를 위해 필요한 기능으로

Create :: 데이터 생성

Read :: 데이터 조회(읽기)

Update :: 데이터 수정

Delete :: 데이터 삭제

를 통틀어 칭하는 말이라고 배웠죠?

지난 번에는 CR을 구현했으니 이번에는 UD를 구현해볼 차례예요 !

UD

not form

## 〈CRUD〉

지난 수업 마지막에 공지드린대로  
crud1 실습 폴더를 사용해줄게요.

git bash (Windows) / terminal (Mac) 에서  
code . 을 통해 해당 폴더에서 VSCODE 를 열어주시고

ctrl + ` (Windows) / control + ` (Mac) 을 통해  
VSCODE 터미널을 열고

Windows :: source venv/Scripts/activate  
Mac :: source venv/bin/activate  
를 통해 가상환경을 실행해줄게요

⟨CRUD not form⟩

CRUD

## crudapp / views.py

```
28  ## UD 실습
29  ✓ def edit(request, id):
30      edit_blog = Blog.objects.get(id = id)
31      return render(request, 'edit.html', {'edit_blog': edit_blog})
32
33  ✓ def update(request, id):
34      update_blog = Blog.objects.get(id = id)
35      update_blog.title = request.POST['title']
36      update_blog.pub_date = timezone.now()
37      update_blog.body = request.POST['body']
38      update_blog.save()
39      return redirect('read')
```

지난 번 작성해준 detail 함수에 이어서  
edit 함수와 update 함수를 작성해줄게요.

# crudapp / views.py

```
9 def new(request):
10     return render(request, 'new.html')
11
12 def create(request):
13     blog = Blog()
14     blog.title = request.POST['title']
15     blog.pub_date = timezone.now()
16     blog.body = request.POST['body']
17     blog.save()
18     return redirect('read')
```

```
28 ## UD 실습
29 ∨ def edit(request, id):
30     edit_blog = Blog.objects.get(id = id)
31     return render(request, 'edit.html', {'edit_blog': edit_blog})
32
33 ∨ def update(request, id):
34     update_blog = Blog.objects.get(id = id)
35     update_blog.title = request.POST['title']
36     update_blog.pub_date = timezone.now()
37     update_blog.body = request.POST['body']
38     update_blog.save()
39     return redirect('read')
```

그런데 코드를 보면 지난 번에 작성한  
글 작성하는데 사용하는 함수가  
new / create 함수 두 개인 것처럼  
글 수정하는데 사용하는 함수도  
edit / update 함수로 작성해줘야 하는 것을  
알 수 있어요

그런데 왜 함수가 두 개 일까요??  
NEW / CREATE 의 관계와 비슷해요.

## EDIT

EDIT은 NEW처럼 글 수정 폼을 보여주며 동시에  
value를 통해 폼에 작성한 내용을 보여줘요.

## UPDATE

UPDATE는 EDIT을 통해 수정한  
글의 데이터베이스를 수정하여 저장하는 역할을 해요.

어떤 로직으로 돌아가는지 감이 좀 잡히죠??



## crudapp / views.py

```
28  ## UD 실습
29  ✓ def edit(request, id):
30      edit_blog = Blog.objects.get(id = id)
31      return render(request, 'edit.html', {'edit_blog': edit_blog})
32
33  ✓ def update(request, id):
34      update_blog = Blog.objects.get(id = id)
35      update_blog.title = request.POST['title']
36      update_blog.pub_date = timezone.now()
37      update_blog.body = request.POST['body']
38      update_blog.save()
39      return redirect('read')
```

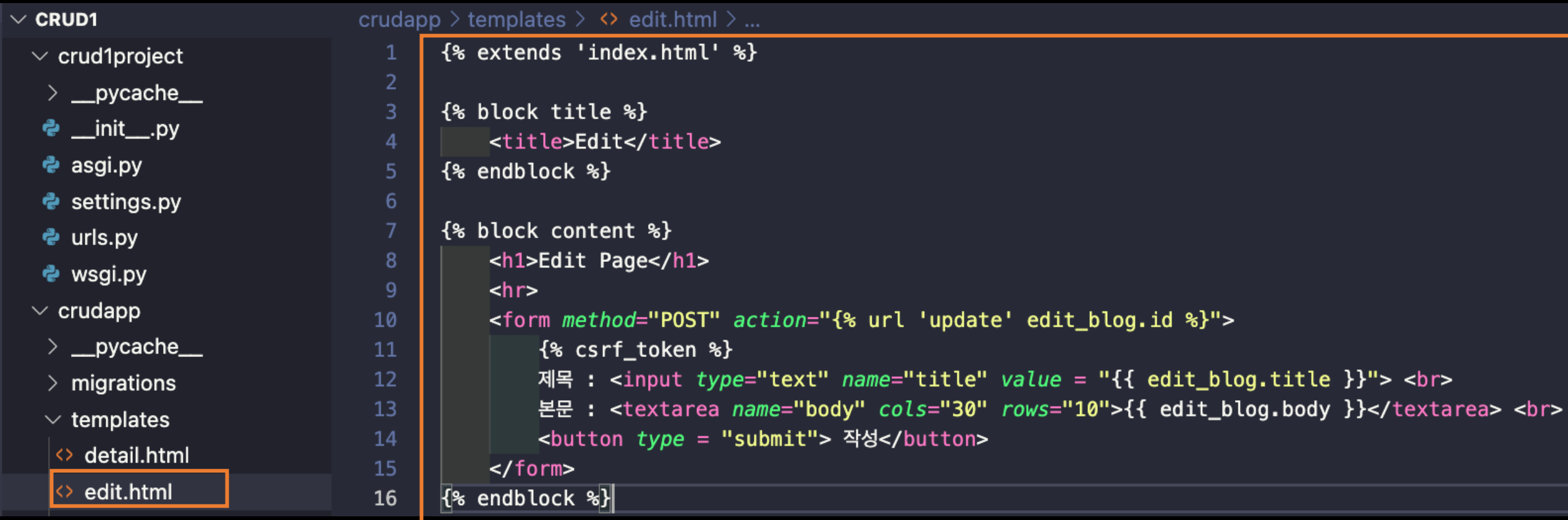
**이 부분은 create 함수와 변수 명만 다르지 동일한 코드로 작성된 객체정보가 모델 테이블에 저장되는 것을 알 수 있어요**

## crudapp / views.py

```
28  ## UD 실습
29  ✓ def edit(request, id):
30      edit_blog = Blog.objects.get(id = id)
31      return render(request, 'edit.html', {'edit_blog': edit_blog})
32
33  ✓ def update(request, id):
34      update_blog = Blog.objects.get(id = id)
35      update_blog.title = request.POST['title']
36      update_blog.pub_date = timezone.now()
37      update_blog.body = request.POST['body']
38      update_blog.save()
39      return redirect('read')
```

저장하면 read 페이지로 redirect 하는 것을 알 수 있습니다

# crudapp / templates / edit.html



The image shows a code editor interface. On the left is a file explorer with a tree view. The root is 'CRUD1', which contains 'crud1project' and 'crudapp'. 'crud1project' has subfolders '\_\_pycache\_\_' and files '\_\_init\_\_.py', 'asgi.py', 'settings.py', 'urls.py', and 'wsgi.py'. 'crudapp' has subfolders '\_\_pycache\_\_' and 'migrations', and a 'templates' folder. The 'templates' folder is expanded, showing 'detail.html' and 'edit.html'. 'edit.html' is selected and highlighted with an orange box. On the right is the code editor for 'edit.html'. The path 'crudapp > templates > <> edit.html > ...' is shown at the top. The code is as follows:

```
1 {% extends 'index.html' %}
2
3 {% block title %}
4     <title>Edit</title>
5 {% endblock %}
6
7 {% block content %}
8     <h1>Edit Page</h1>
9     <hr>
10    <form method="POST" action="{% url 'update' edit_blog.id %}">
11        {% csrf_token %}
12        제목 : <input type="text" name="title" value = "{{ edit_blog.title }}"> <br>
13        본문 : <textarea name="body" cols="30" rows="10">{{ edit_blog.body }}</textarea> <br>
14        <button type = "submit"> 작성</button>
15    </form>
16 {% endblock %}
```

**crudapp / templates 에 edit.html 을 만들어  
이와 같이 작성해주세요.  
코드를 보면 new.html 코드와 비슷한 것 같아요.**

# crudapp / templates / edit.html

crudapp > templates > <> edit.html > ...

```
1  {% extends 'index.html' %}
2
3  {% block title %}
4  <title>Edit</title>
5  {% endblock %}
6
7  {% block content %}
8  <h1>Edit Page</h1>
9  <hr>
10 <form method="POST" action="{% url 'update' edit_blog.id %}">
11     {% csrf_token %}
12     제목 : <input type="text" name="title" value = "{{ edit_blog.title }}"> <br>
13     본문 : <textarea name="body" cols="30" rows="10">{{ edit_blog.body }}</textarea> <br>
14     <button type = "submit"> 작성</button>
15 </form>
16 {% endblock %}
```

이는 read.html에서 작성한 것과 유사하게  
수정할 글의 id값을 가져와 해당 글을 수정하기 위해서 사용하는 것이예요.

# crudapp / templates / edit.html

crudapp > templates > <> edit.html > ...

```
1  {% extends 'index.html' %}
2
3  {% block title %}
4  <title>Edit</title>
5  {% endblock %}
6
7  {% block content %}
8  <h1>Edit Page</h1>
9  <hr>
10 <form method="POST" action="{% url 'update' edit_blog.id %}">
11     {% csrf_token %}
12     제목 : <input type="text" name="title" value = "{{ edit_blog.title }}"> <br>
13     본문 : <textarea name="body" cols="30" rows="10">{{ edit_blog.body }}</textarea> <br>
14     <button type = "submit"> 작성</button>
15 </form>
16 {% endblock %}
```

이는 수정하고 싶은 글의 field를 얻어오기 위해서  
해당하는 정보를 value를 통해 가져왔어요

# crudapp / templates / edit.html

crudapp > templates > <> edit.html > ...

```
1  {% extends 'index.html' %}
2
3  {% block title %}
4  <title>Edit</title>
5  {% endblock %}
6
7  {% block content %}
8  <h1>Edit Page</h1>
9  <hr>
10 <form method="POST" action="{% url 'update' edit_blog.id %}">
11     {% csrf_token %}
12     제목 : <input type="text" name="title" value = "{{ edit_blog.title }}"> <br>
13     본문 : <textarea name="body" cols="30" rows="10">{{ edit_blog.body }}</textarea> <br>
14     <button type = "submit"> 작성</button>
15 </form>
16 {% endblock %}
```

이 부분은 body field 의 내용을 보여줘요



## crudlproject / urls.py

```
20 urlpatterns = [  
21     path('admin/', admin.site.urls),  
22     path('', crudapp.views.index, name = 'index'),  
23     path('new/', crudapp.views.new, name = 'new'),  
24     path('new/create/', crudapp.views.create, name = 'create'),  
25     path('read/', crudapp.views.read, name = 'read'),  
26     path('detail/<str:id>', crudapp.views.detail, name = 'detail'),  
27  
28     ## UD 실습  
29     path('edit/<str:id>', crudapp.views.edit, name = 'edit'),  
30     path('update/<str:id>', crudapp.views.update, name = 'update'),  
31 ]
```

url 이동을 위한 코드를 작성해줄게요.

# crudapp / templates / detail.html

```
crudapp > templates > <> detail.html > ...
```

```
1  {% extends 'index.html' %}
2
3  {% block title %}
4      <title>Detail</title>
5  {% endblock %}
6
7  {% block content %}
8      <h1>Detail Page</h1>
9      <hr>
10     <h3>{{ blog.title }}</h3>
11     <p>{{ blog.pub_date }}</p>
12     <p>{{ blog.body }}</p>
13     <br>
14     <a href="{% url 'edit' blog.id %}">글 수정하기</a>
15 {% endblock %}
```

글 수정 페이지로 이동하기 위해  
detail.html 에 a 태그를 작성해줄게요



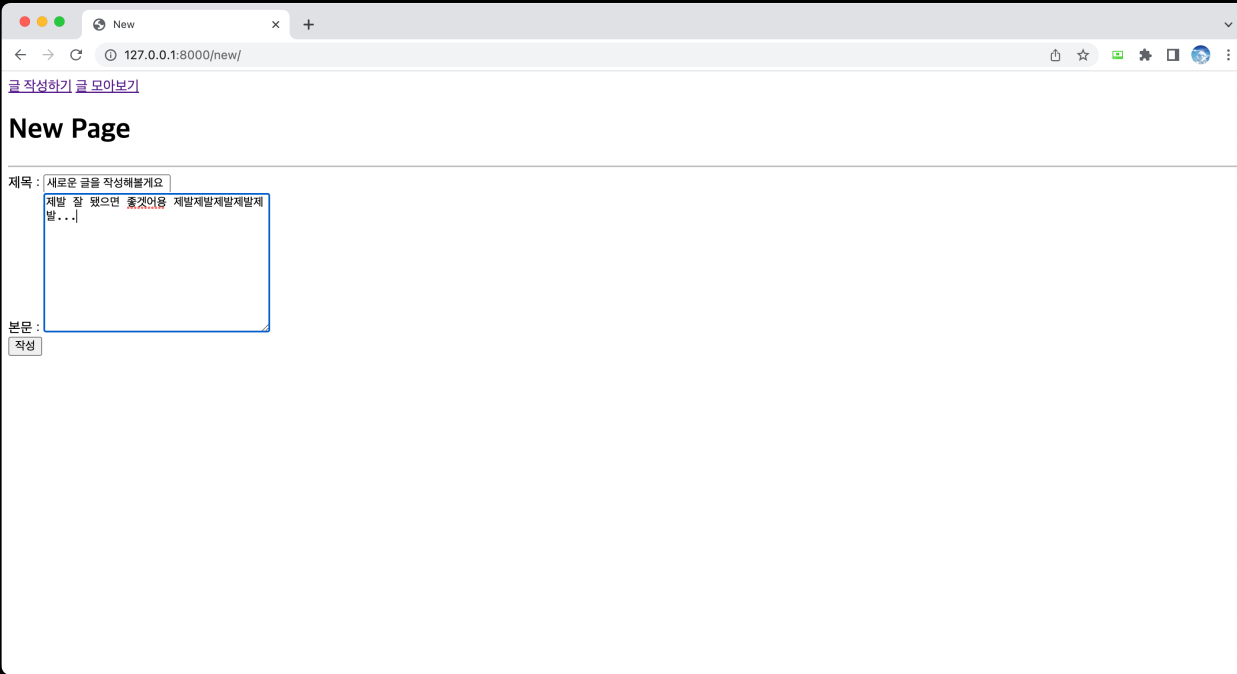
# 런 서버 해주기

**Windows :: python manage.py runserver**

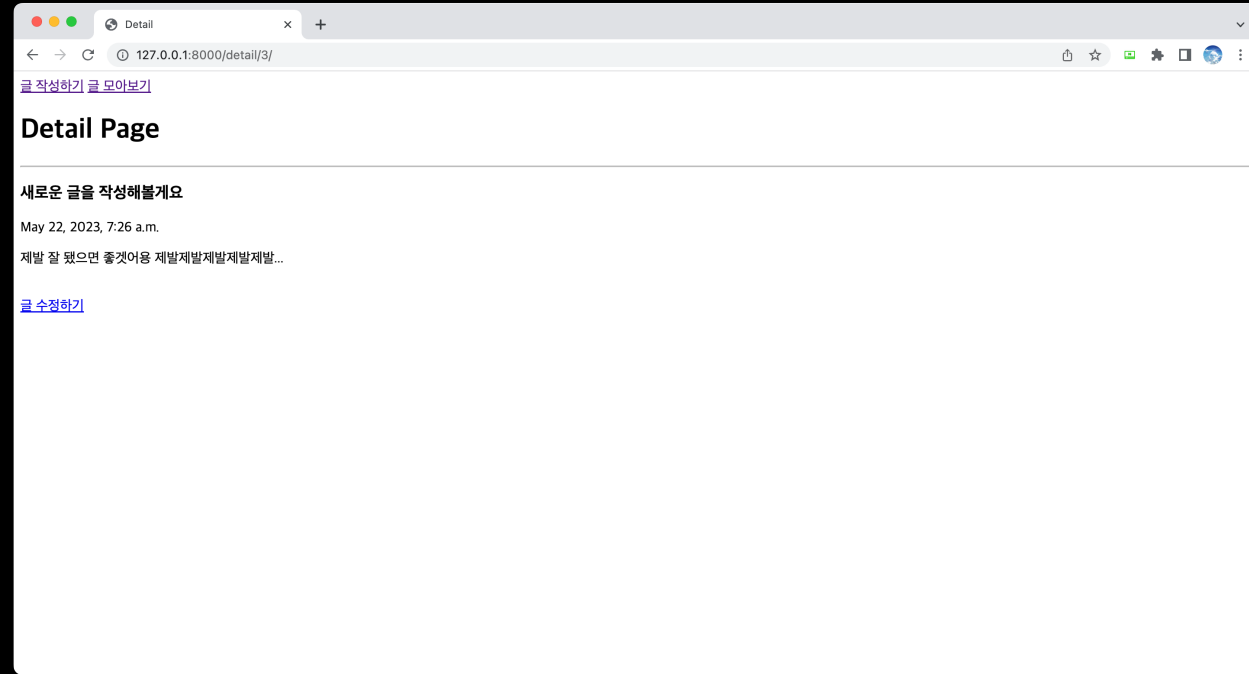
**Mac :: python3 manage.py runserver**

**런서버 해줄게요**

# 실습 확인



지난 실습이 오류없이 되는지 확인하기 위해  
이렇게 새 글을 작성해주고  
read 페이지에서 해당 글을 클릭해주세요



이렇게 글 내용이 detail 페이지에 잘 뜬다면

아래 글 수정하기 라고 되어있는  
a 태그를 클릭해주세요

# 실습 확인

127.0.0.1:8000/edit/3/

[글 작성하기 글 모아보기](#)

### Edit Page

제목 : 새로운 글을 작성해볼게요

제발 잘 됐으면 좋겠어용 제발제발제발제발제발...

본문 :

작성

이렇게 글 수정 form 에  
아까 detail 페이지에서  
보았던 내용이 잘 뜨죠?

127.0.0.1:8000/edit/3/

[글 작성하기 글 모아보기](#)

### Edit Page

제목 : 글을 수정해봅니다

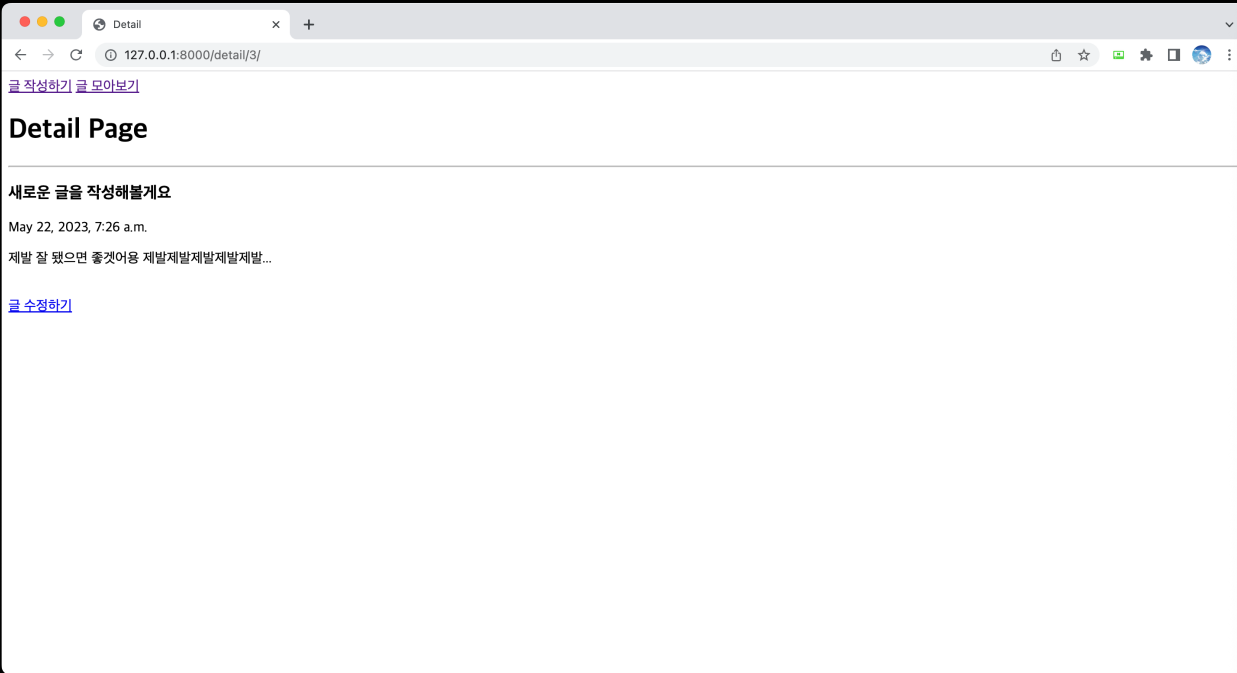
제발제발제발제발 😊

본문 :

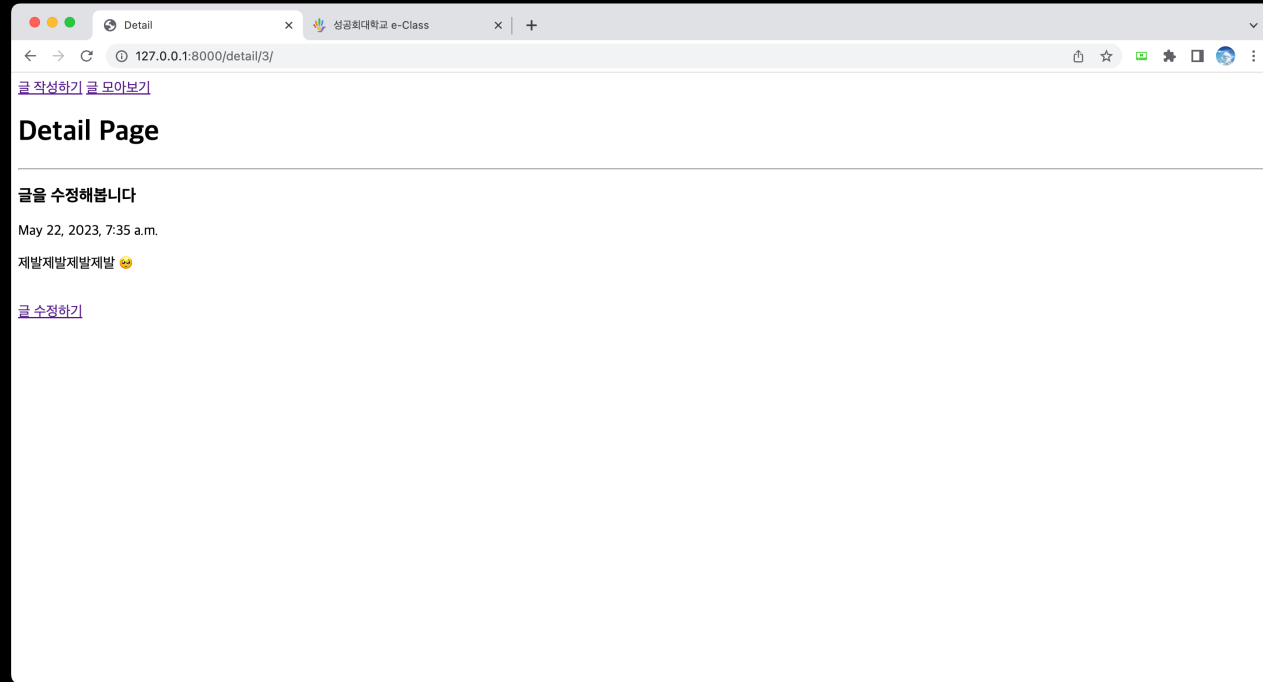
작성

내용을 수정해주고  
작성 버튼 클릭

# 실습 확인



그럼 이랬던 글이



이렇게 잘 수정이 되었어요 !

**서버 종료**

**잘 되신다면**

**Windows :: ctrl + c**

**Mac :: control + c**

**서버 종료!**

⟨CRUD not form⟩

CRUD

## crudapp / views.py

```
41 def delete(request, id):  
42     delete_blog = get_object_or_404(Blog, id = id)  
43     delete_blog.delete()  
44     return redirect('read')
```

익숙한 코드죠?

**get\_object\_or\_404** 는

Blog모델에서 id값을 가져오지 못하면 404에러를 일으키는 기능을 한다고  
지난 시간에 배웠어요!

## crudlproject / urls.py

```
20 urlpatterns = [  
21     path('admin/', admin.site.urls),  
22     path('', crudapp.views.index, name = 'index'),  
23     path('new/', crudapp.views.new, name = 'new'),  
24     path('new/create/', crudapp.views.create, name = 'create'),  
25     path('read/', crudapp.views.read, name = 'read'),  
26     path('detail/<str:id>', crudapp.views.detail, name = 'detail'),  
27  
28     ## UD 실습  
29     path('edit/<str:id>', crudapp.views.edit, name = 'edit'),  
30     path('update/<str:id>', crudapp.views.update, name = 'update'),  
31     path('delete/<str:id>', crudapp.views.delete, name = 'delete'),  
32 ]
```

**delete 기능을 위한 url 연결도 해줄게요**



# crudapp / templates / detail.html

crudapp > templates > <> detail.html > ...

```
1  {% extends 'index.html' %}
2
3  {% block title %}
4  <title>Detail</title>
5  {% endblock %}
6
7  {% block content %}
8  <h1>Detail Page</h1>
9  <hr>
10 <h3>{{ blog.title }}</h3>
11 <p>{{ blog.pub_date }}</p>
12 <p>{{ blog.body }}</p>
13 <br>
14 <a href="{% url 'edit' blog.id %}">글 수정하기</a>
15 <br>
16 <a href="{% url 'delete' blog.id %}">글 삭제하기</a>
17 {% endblock %}
```

이렇게 detail 페이지에 삭제를 위한 a 태그를 작성해주세요

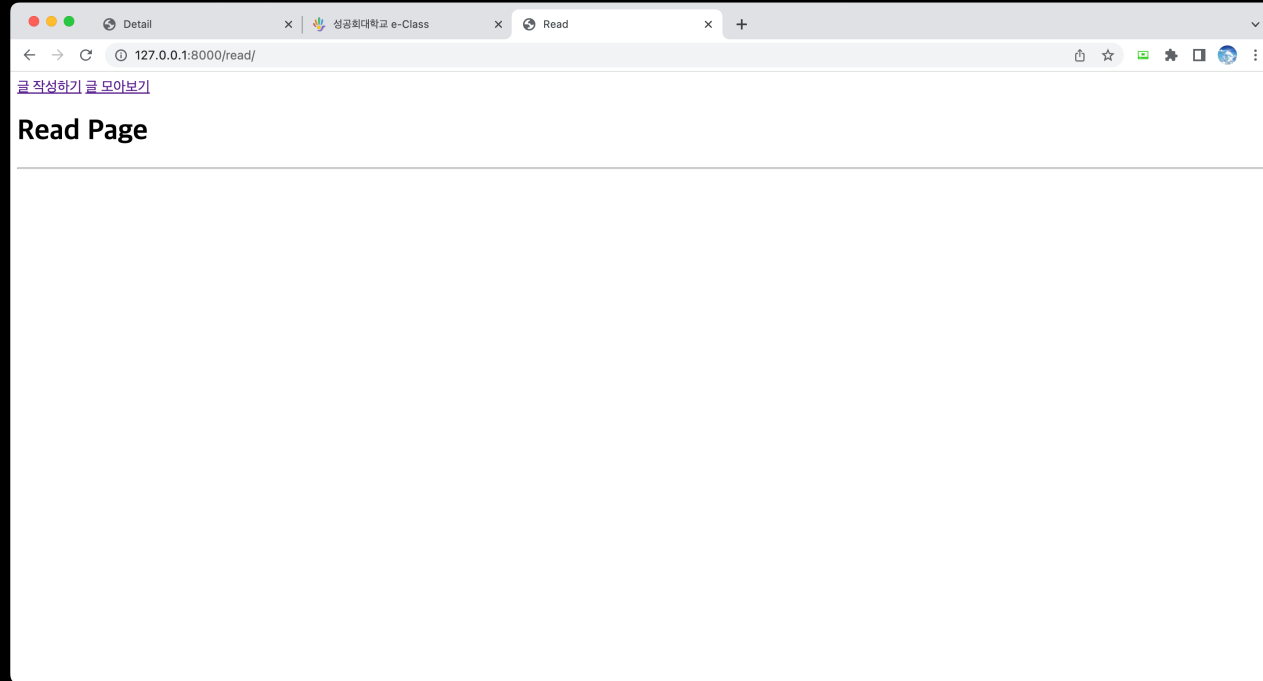
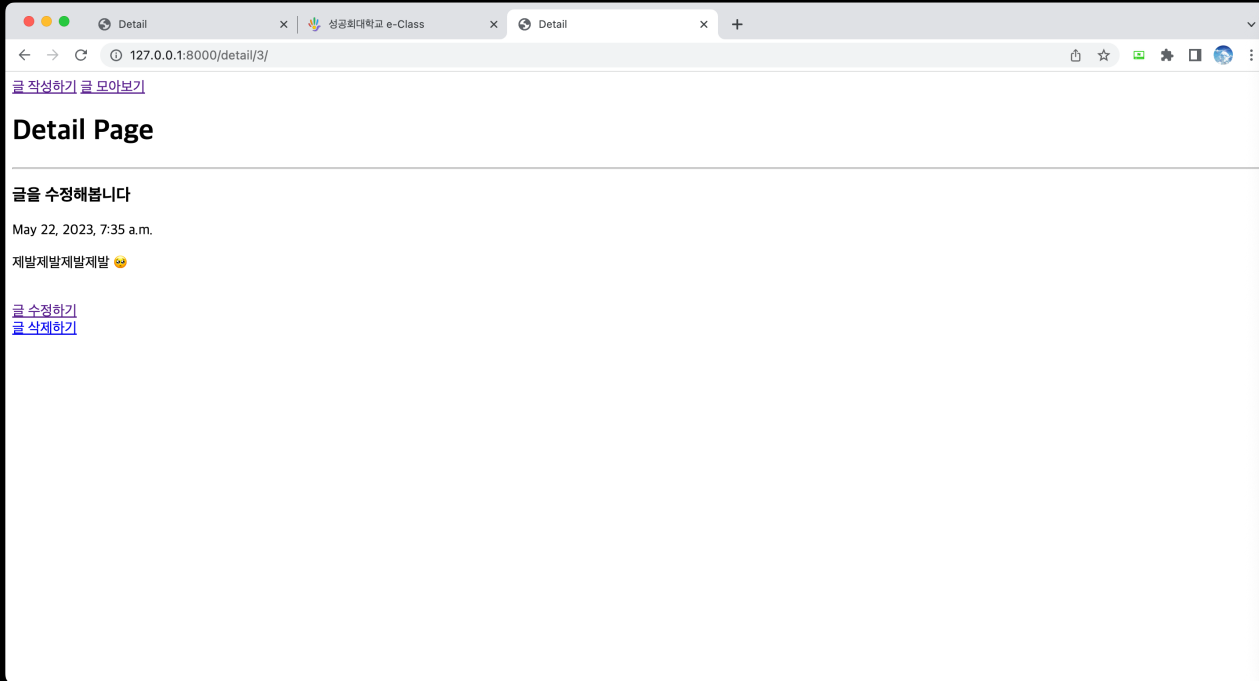
# 런 서버 해주기

**Windows :: python manage.py runserver**

**Mac :: python3 manage.py runserver**

**런서버 해줄게요**

# 실습 확인



**Read 페이지에 가서 아까 작성한 글을 누르면  
이렇게 추가한  
글 삭제를 위한 a 태그가 잘 뜨시나요?**

**잘 뜨면 글 삭제하기를 눌러주세요 !**

**그럼 이렇게  
글 삭제와 동시에  
read 페이지로 이동합니다 ~.~**

**서버 종료**

**잘 되신다면**

**Windows :: ctrl + c**

**Mac :: control + c**

**서버 종료!**

**UD**

**use form**

## 〈CRUD〉

form 을 사용하지 않는 코드로 실습했으니  
이번에는 form 을 사용하는 코드로 실습을 진행해줘야겠죠?  
지난 시간에 사용한 crud2 폴더를 사용해줄게요.

git bash (Windows) / terminal (Mac) 에서  
code . 을 통해 해당 폴더에서 VSCODE 를 열어주시고

ctrl + ` (Windows) / control + ` (Mac) 을 통해  
VSCODE 터미널을 열고

Windows :: source venv/Scripts/activate  
Mac :: source venv/bin/activate  
를 통해 가상환경을 실행해줄게요

〈CRUD use form〉

CRUD

## crudapp / views.py

```
31  ## UD 실습
32  def update(request, id):
33      blog = get_object_or_404(Blog, id = id)
34      if request.method == 'POST':
35          form = BlogForm(request.POST, instance = blog)
36          if form.is_valid():
37              form = form.save(commit = False)
38              form.pub_date = timezone.now()
39              form.save()
40              return redirect('read')
41      else:
42          form = BlogForm(instance = blog)
43      return render(request, 'update.html', {'form': form})
```

지난 번에 작성해준 detail 함수에 이어 작성해주세요



# crudapp / views.py

```
def create(request):  
    if request.method == 'POST':  
        form = BlogForm(request.POST)  
        if form.is_valid():  
            form = form.save(commit = False)  
            form.pub_date = timezone.now()  
            form.save()  
            return redirect('read')  
    else:  
        form = BlogForm  
        return render(request, 'create.html', {'form': form})
```

```
31  ## UD 실습  
32  def update(request, id):  
33      blog = get_object_or_404(Blog, id = id)  
34      if request.method == 'POST':  
35          form = BlogForm(request.POST, instance = blog)  
36          if form.is_valid():  
37              form = form.save(commit = False)  
38              form.pub_date = timezone.now()  
39              form.save()  
40              return redirect('read')  
41      else:  
42          form = BlogForm(instance = blog)  
43          return render(request, 'update.html', {'form': form})
```

**이것도 지난 번에 작성한 create 함수와  
꽤 유사한 것 같죠?**

**그럼 코드리뷰 해볼게요**

## crudapp / views.py

```
## UD 실습
```

```
def update(request, id):
```

```
    blog = get_object_or_404(Blog, id = id)
```

```
    if request.method == 'POST':
```

```
        form = BlogForm(request.POST, instance = blog)
```

```
        if form.is_valid():
```

```
            form = form.save(commit = False)
```

```
            form.pub_date = timezone.now()
```

```
            form.save()
```

```
            return redirect('read')
```

```
    else:
```

```
        form = BlogForm(instance = blog)
```

```
        return render(request, 'update.html', {'form': form})
```

정보를 받는 방식인 Method가 Post 방식이라면 :

## crudapp / views.py

```
## UD 실습
```

```
def update(request, id):
```

```
    blog = get_object_or_404(Blog, id = id)
```

```
    if request.method == 'POST':
```

```
        form = BlogForm(request.POST, instance = blog)
```

```
        if form.is_valid():
```

```
            form = form.save(commit = False)
```

```
            form.pub_date = timezone.now()
```

```
            form.save()
```

```
            return redirect('read')
```

```
    else:
```

```
        form = BlogForm(instance = blog)
```

```
        return render(request, 'update.html', {'form': form})
```

**POST로 요청받은 BlogForm이 정보를 잘 받았는지 확인하고**

## crudapp / views.py

```
## UD 실습
def update(request, id):
    blog = get_object_or_404(Blog, id = id)
    if request.method == 'POST':
        form = BlogForm(request.POST, instance = blog)
        if form.is_valid():
            form = form.save(commit = False)
            form.pub_date = timezone.now()
            form.save()
            return redirect('read')
        else:
            form = BlogForm(instance = blog)
    return render(request, 'update.html', {'form': form})
```

**instance**는 우리가 수정할 글이 어떤 글인지  
글의 id를 함수에게 설명해주는 코드입니다 ~

## crudapp / views.py

```
## UD 실습
```

```
def update(request, id):
```

```
    blog = get_object_or_404(Blog, id = id)
```

```
    if request.method == 'POST':
```

```
        form = BlogForm(request.POST, instance = blog)
```

```
        if form.is_valid():
```

```
            form = form.save(commit = False)
```

```
            form.pub_date = timezone.now()
```

```
            form.save()
```

```
            return redirect('read')
```

```
    else:
```

```
        form = BlogForm(instance = blog)
```

```
        return render(request, 'update.html', {'form': form})
```

**form이 유효하다면 (잘 입력 되었다면)**

## crudapp / views.py

```
## UD 실습
def update(request, id):
    blog = get_object_or_404(Blog, id = id)
    if request.method == 'POST':
        form = BlogForm(request.POST, instance = blog)
        if form.is_valid():
            form = form.save(commit = False)
            form.pub_date = timezone.now()
            form.save()
            return redirect('read')
    else:
        form = BlogForm(instance = blog)
    return render(request, 'update.html', {'form': form})
```

**form 을 저장 대기 상태에 두고  
pub\_date field 에 현재 시간을 저장해주고  
form 을 전체 저장해줘요**

## crudapp / views.py

```
## UD 실습
def update(request, id):
    blog = get_object_or_404(Blog, id = id)
    if request.method == 'POST':
        form = BlogForm(request.POST, instance = blog)
        if form.is_valid():
            form = form.save(commit = False)
            form.pub_date = timezone.now()
            form.save()
            return redirect('read')
    else:
        form = BlogForm(instance = blog)
    return render(request, 'update.html', {'form': form})
```

그리고 read 페이지로 redirect 합니다



## crudapp / views.py

```
## UD 실습
```

```
def update(request, id):
```

```
    blog = get_object_or_404(Blog, id = id)
```

```
    if request.method == 'POST':
```

```
        form = BlogForm(request.POST, instance = blog)
```

```
        if form.is_valid():
```

```
            form = form.save(commit = False)
```

```
            form.pub_date = timezone.now()
```

```
            form.save()
```

```
            return redirect('read')
```

```
    else:
```

```
        form = BlogForm(instance = blog)
```

```
        return render(request, 'update.html', {'form': form})
```

정보를 받는 방식이 POST 가 아니라면 이 코드블럭이 실행돼요.  
Form 을 보여주는 코드입니다 ~



## crudapp / views.py

```
## UD 실습
```

```
def update(request, id):
```

```
    blog = get_object_or_404(Blog, id = id)
```

```
    if request.method == 'POST':
```

```
        form = BlogForm(request.POST, instance = blog)
```

```
        if form.is_valid():
```

```
            form = form.save(commit = False)
```

```
            form.pub_date = timezone.now()
```

```
            form.save()
```

```
            return redirect('read')
```

```
    else:
```

```
        form = BlogForm(instance = blog)
```

```
    return render(request, 'update.html', {'form': form})
```

여기서도 동일하게 글 정보를 가져오기 위해  
instance = blog 코드를 작성해줘요

# crudapp / templates / update.html

▼ CRUD2

> crud2project

▼ crudapp

> \_\_pycache\_\_

> migrations

▼ templates

<> create.html

<> detail.html

<> index.html

<> read.html

<> update.html

🔗 \_\_init\_\_.py

🔗 admin.py

🔗 apps.py

🔗 forms.py

🔗 models.py

crudapp > templates > <> update.html > ...

```
1 {% extends 'index.html' %}
2
3 {% block title %}
4     <title>Update</title>
5 {% endblock %}
6
7 {% block content %}
8     <h1>Update Page</h1>
9     <hr>
10    <form method="POST">
11        {% csrf_token %}
12        <table>
13            {{ form.as_table }}
14        </table>
15        <button type = "submit">작성</button>
16    </form>
17 {% endblock %}
```

crudapp / templates 폴더에 update.html을 만들고 작성해주세요

## crud2project / urls.py

```
20 urlpatterns = [  
21     path('admin/', admin.site.urls),  
22     path('', crudapp.views.index, name = 'index'),  
23     path('create/', crudapp.views.create, name = 'create'),  
24     path('read/', crudapp.views.read, name = 'read'),  
25     path('detail/<str:id>', crudapp.views.detail, name = 'detail'),  
26  
27     ## UD 실습  
28     path('update/<str:id>', crudapp.views.update, name = 'update'),  
29 ]
```

**detail url 아래에  
url 이동을 위한 코드를 작성해줄게요.**

# crudapp / templates / detail.html

crudapp > templates > <> detail.html >  a

```
1  {% extends 'index.html' %}
2
3  {% block title %}
4      <title>Detail</title>
5  {% endblock %}
6
7  {% block content %}
8      <h1>Detail Page</h1>
9      <hr>
10     <h3>{{ blog.title }}</h3>
11     <p>{{ blog.pub_date }}</p>
12     <p>{{ blog.body }}</p>
13     <br>
14     <a href="{% url 'update' blog.id %}">글 수정하기</a>
15 {% endblock %}
```

**detail 페이지에 글 수정을 위한 a 태그를 작성해주세요**

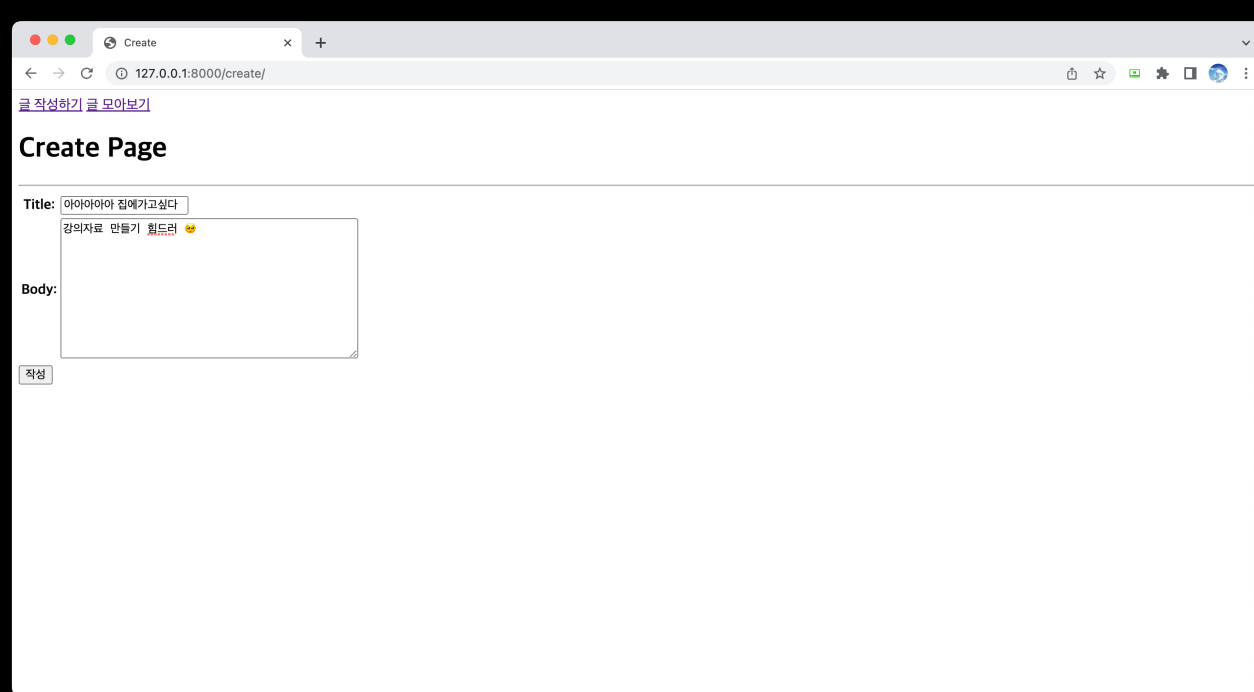
# 런 서버 해주기

**Windows :: python manage.py runserver**

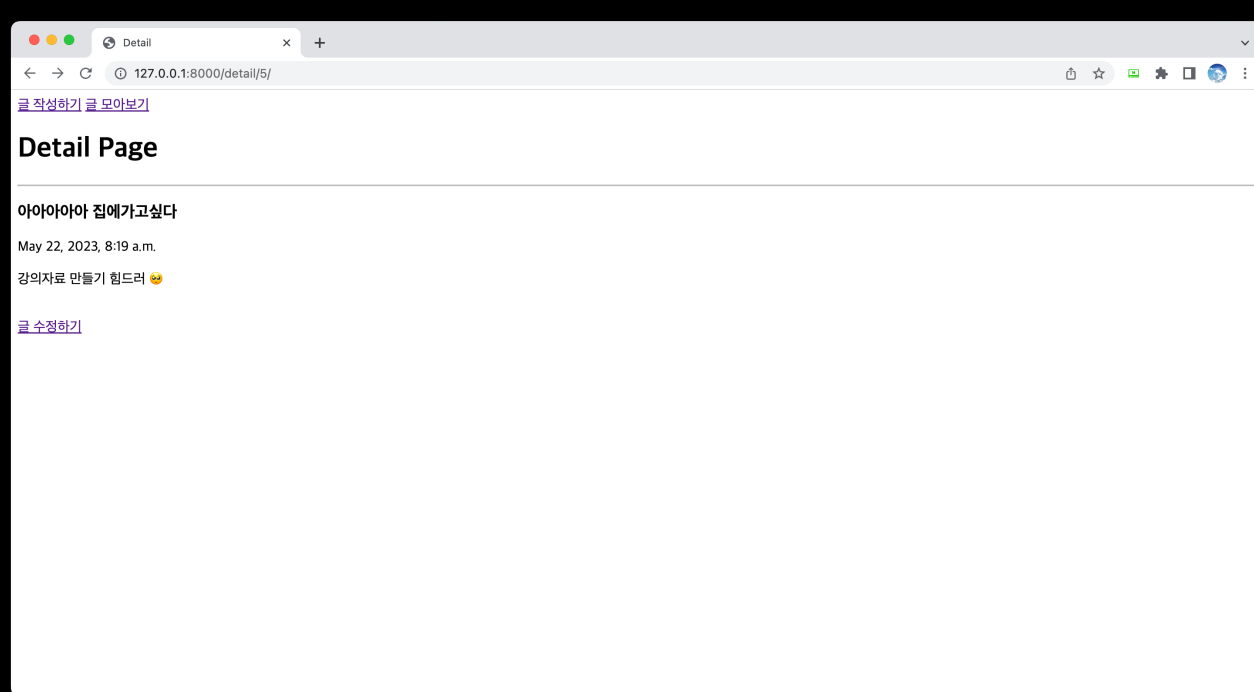
**Mac :: python3 manage.py runserver**

**런서버 해줄게요**

# 실습 확인

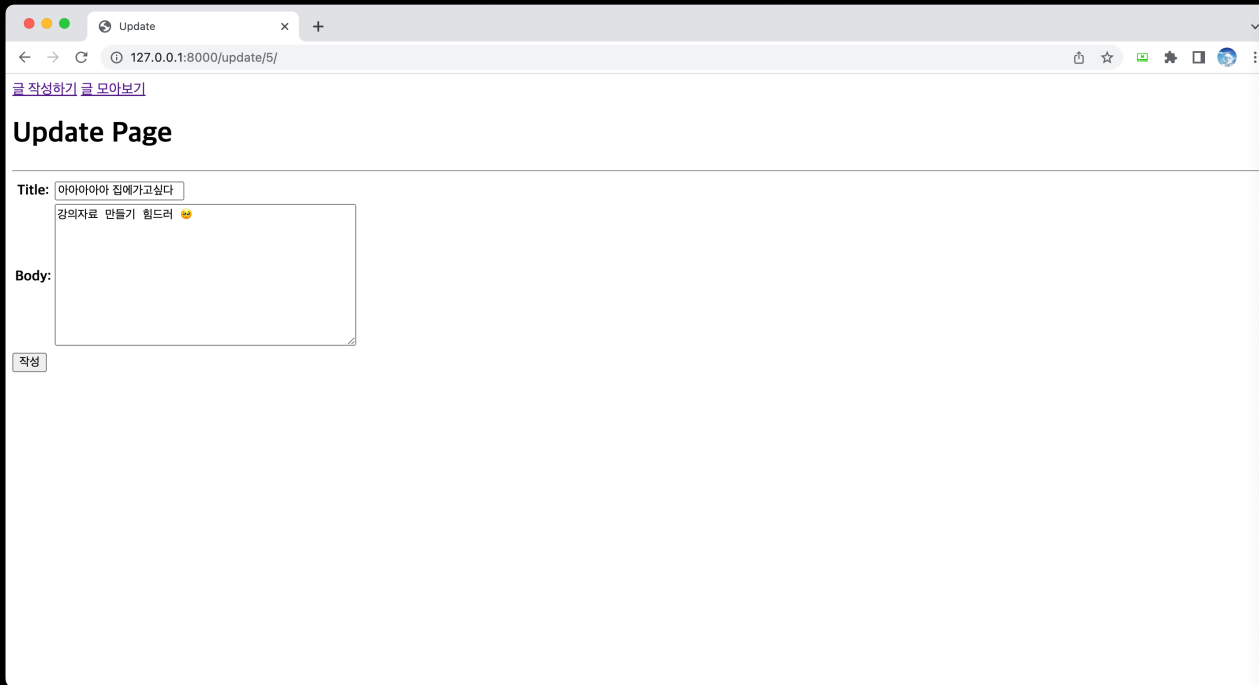


지난 실습이 잘 되는지 확인차  
글을 새로 작성해주세요

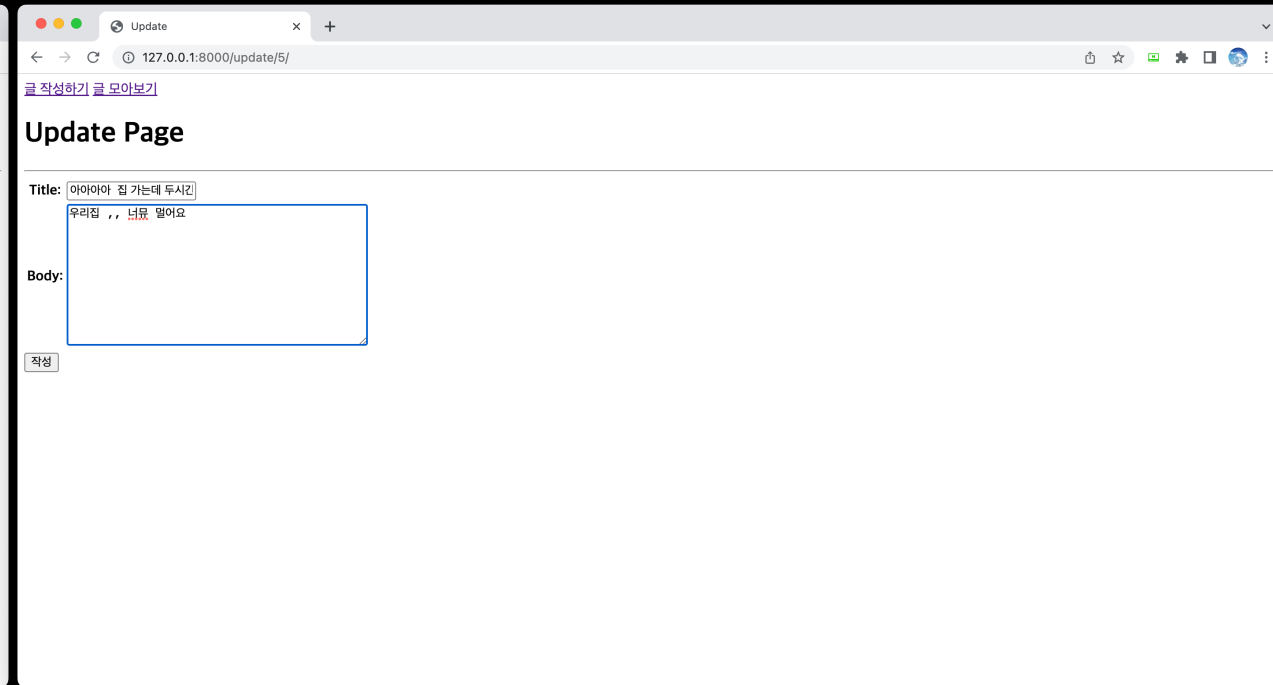


read 페이지에서 작성한 글을 클릭해  
detail 페이지로 이동하면  
이렇게 작성해준 a 태그가 잘 뜨죠?  
그러면 글 수정하기를 클릭해줄게요

# 실습 확인

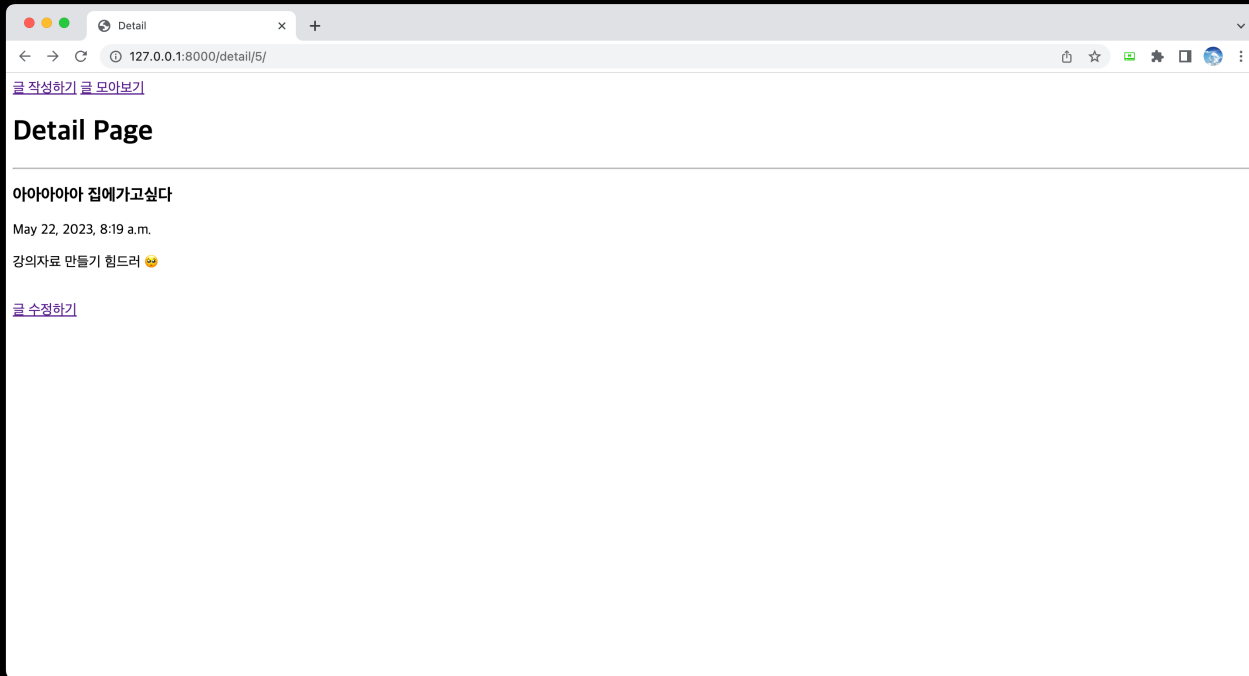


아까 detail 페이지에서 보인 내용과 동일하게  
Update 페이지에서도 잘 보여요

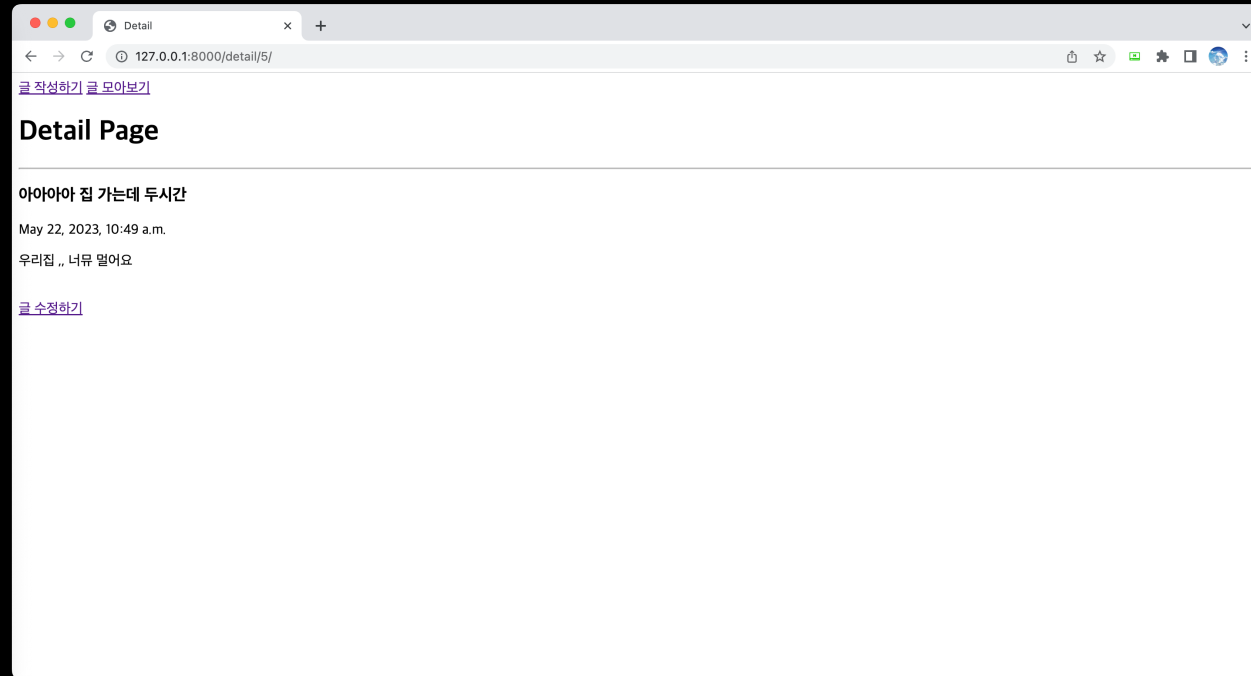


이렇게 글을 수정해준 후  
작성 버튼을 눌러줄게요

# 실습 확인



이랬던 글이 이렇게 바뀌어요



그러면  
이렇게 아까 수정해준 것처럼  
detail page에도 내용이 잘 떠요



**서버 종료**

**잘 되신다면**

**Windows :: ctrl + c**

**Mac :: control + c**

**서버 종료!**

〈CRUD use form〉

CRUD

## crudapp / views.py

```
45 def delete(request, id):  
46     blog = get_object_or_404(Blog, id = id)  
47     blog.delete()  
48     return redirect('read')
```

## crud2project / urls.py

```
## UD 실습  
path('update/<str:id>', crudapp.views.update, name = 'update'),  
path('delete/<str:id>', crudapp.views.delete, name = 'delete'),  
]
```

이렇게 view 와 url 코드를 작성해주세요

# crudapp / templates / detail.html

crudapp > templates > <> detail.html > a

```
1  {% extends 'index.html' %}
2
3  {% block title %}
4  <title>Detail</title>
5  {% endblock %}
6
7  {% block content %}
8  <h1>Detail Page</h1>
9  <hr>
10 <h3>{{ blog.title }}</h3>
11 <p>{{ blog.pub_date }}</p>
12 <p>{{ blog.body }}</p>
13 <br>
14 <a href="{% url 'update' blog.id %}">글 수정하기</a>
15 <br>
16 <a href="{% url 'delete' blog.id %}">글 삭제하기</a>
17 {% endblock %}
18
```

이렇게 삭제를 위한 a 태그를 작성해줄게요

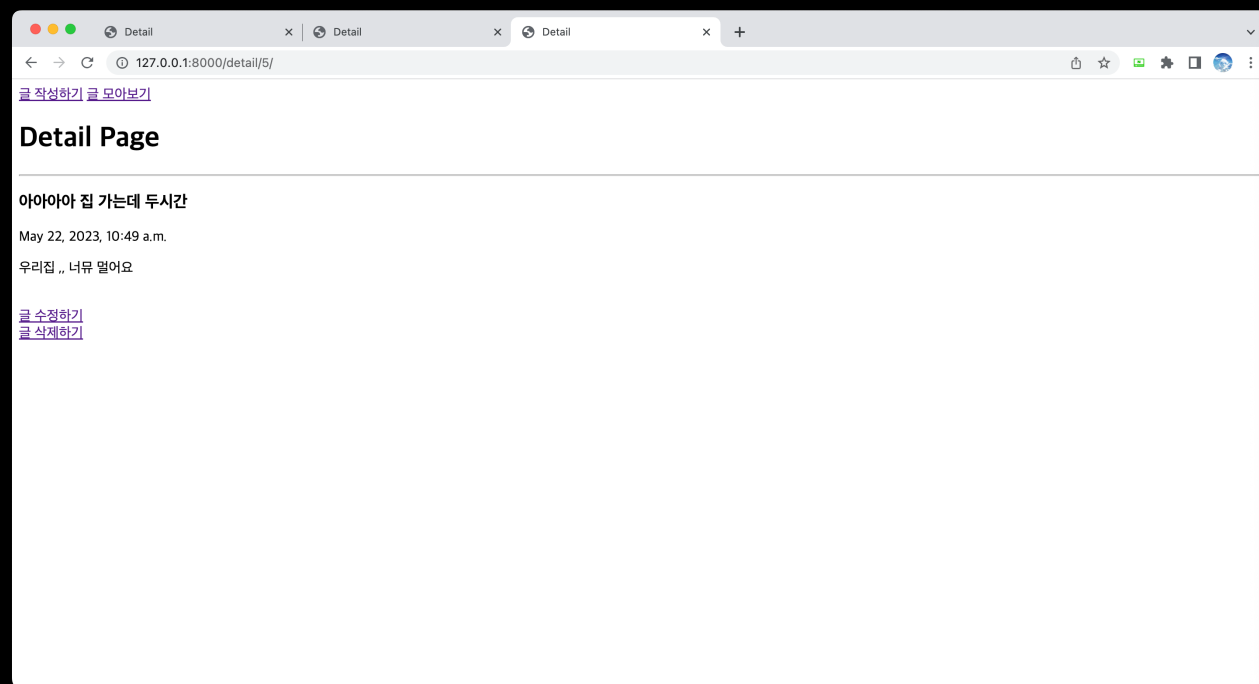
# 런 서버 해주기

**Windows :: python manage.py runserver**

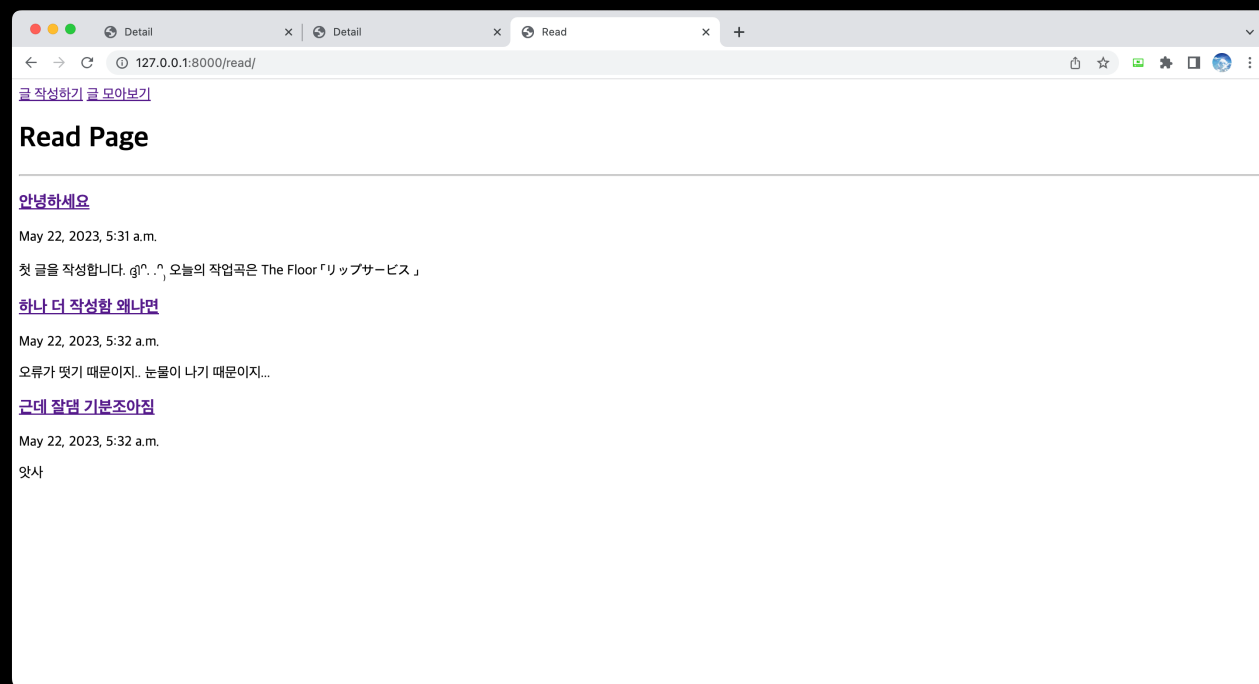
**Mac :: python3 manage.py runserver**

**런서버 해줄게요**

# 실습 확인



삭제하기를 눌러 글이 잘 삭제되는지  
확인해주세요  
글이 삭제됨과 동시에  
Read 페이지로 이동해야해요



이렇게 글이 잘  
삭제되면 성공 ~

**서버 종료**

**잘 되신다면**

**Windows :: ctrl + c**

**Mac :: control + c**

**서버 종료!**

오늘은 금요일이니까

과제 있습니다

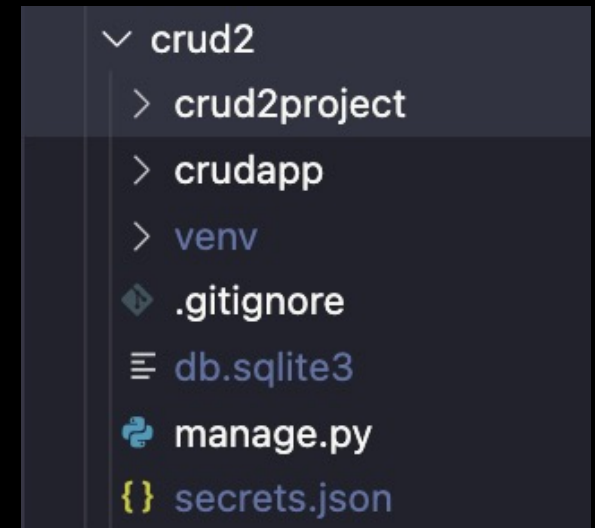
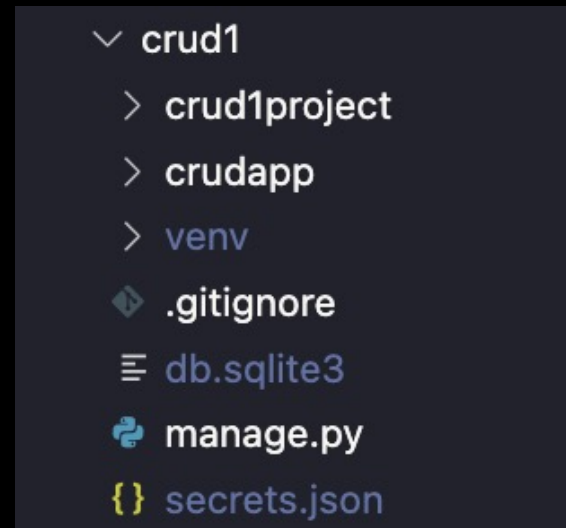
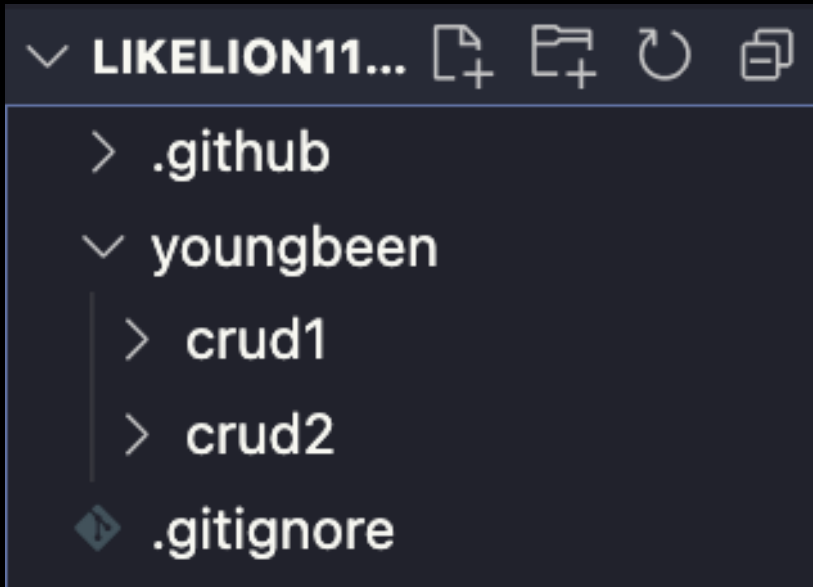


## 과제 전반 가이드라인

과제 제출 :: 6월 1일 목요일 오후 11시 59분까지

지난 수업과 오늘 수업에 배운

**form 을 사용하지 않는 crud 구현 (폴더명 crud1) 과  
modelForm 을 사용하는 crud를 구현 (폴더명 crud2) 하여 제출해주세요.**



이렇게 crud1 과 crud2 모두  
가상환경 / db / secrets.json 이  
gitignore 되어야 해요  
(이거 안 되면 가이드라인 미준수)

평소 하시던 것처럼 영문이름 폴더 하위에  
crud1 / crud2 를 넣어주세요

**과제 전반 가이드라인 (crud1 / crud2 모두 적용)**

**.gitignore / secret key 분리 모두 하셔야 해용**

**그리고 pull request 하심과 동시에**

**조지미에게 secrets.json 파일도 전달해주세요 !!!**

**(이 파일이 없다면 조지미가 런서버가 안 돼서 과제 확인이 불가해요 ...  
따라서 가이드라인 미준수가 될 수 밖에 없습니다 ㅠ )**

**백엔드는 원래 오류가 많이 뜨는데**

**오류가 뜰 때 바로 조지미에게 물어보기 보다는**

**오류가 뜬다면 해당 오류를 자세히 읽어보고**

**(오류에는 항상 어디서 났는지 / 왜 났는지 힌트가 있어요)**

**그래도 모르겠다면 구글링을 해보고**

**그래도그래도그래도 안 되면 조지미에게 물어봐주세요.**

**과제 전반 가이드라인 (crud1 / crud2 모두 적용)**

**crud1의 프로젝트명 :: crud1prj / 앱명 :: crud1app**  
**crud2의 프로젝트명 :: crud2prj / 앱명 :: crud2app**  
**그리고 모델명은 Post로 해주세요.**

**런서버 하자마자 보이는 index 페이지를 만들고  
CRUD 페이지에서 이 index 페이지를 상속 받도록 해주세요.**

**모든 페이지에서 index.html 을 상속받아야 합니다.**

**오늘 배운 것과 동일하게 제목 / 작성시간 / 내용 field를 가지는 것과 동시에  
작성자를 입력할 수 있는 field를 가지고  
작성자는 최대 20자까지만 작성할 수 있게 해주세요.**

## 과제 전반 가이드라인 (crud1 / crud2 모두 적용)

그리고 구글링을 통해 django 에서 사용할 수 있는 필드를 공부한 후  
crud1 / crud2 각각 다른 필드를 1개 이상 추가해주세요. (최소 1개)  
이 field는 지금까지 작성한 field와 중복되면 안 돼요 !

즉, crud1 :: 제목 / 작성시간 / 내용 / 작성자 + 최소 1개이상의 추가 field  
crud2 :: 제목 / 작성시간 / 내용 / 작성자 + 최소 1개이상의 추가 field

과제를 하시며 해당 Field 가 어떤 형태의 입력을 받을 때 사용하는 것이 좋은지  
꼭 이해하고 사용하시길 바랍니다

(FileField / ImageField 는 사용하지 않습니다)

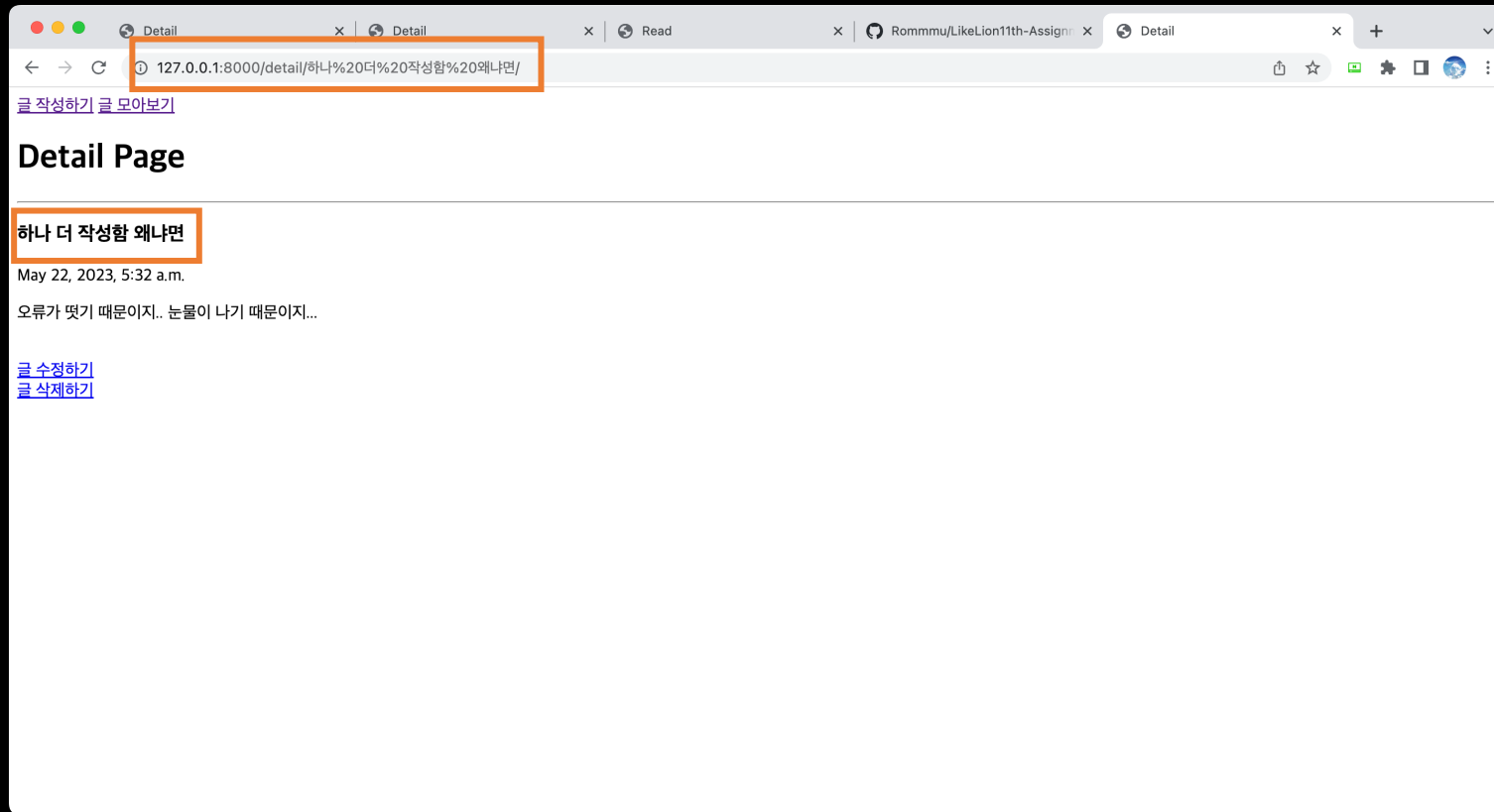
## 과제 전반 가이드라인 (crud1 에만 적용)

**우리가 detail 과 같이 특정 게시글을 식별하기 위해  
id 값을 설정해주었죠??**

**우리는 실습할 때 이 id 값을 문자열 타입으로 설정해주었지만,  
과제에서는 해당 타입을 정수 타입으로 설정해주세요.**

## 과제 전반 가이드라인 (crud2 에만 적용)

이번에는 crud1 과 달리  
crud2 에는 특정 게시글을 식별하기 위한 값을 id 가 아닌 제목으로 설정해주세요.  
어려우실 수 있어 예시를 들어드릴게요.



이와 같이 제목이 url 주소창에 나타나도록 ! 해주시면 돼요.

**오늘은 가이드라인을  
간단히 드렸습니당 ㅎㅎ ..**

**다들 너무 뻥세다구 하시길래 ..  
과제 모두 화이팅 하시구  
다음주에 뵈어용 31^..^)**