4장. 폼(form) & 네비게이션

목 차

장고 폼으로 등록 페이지 만들기

- **질문 등록 페이지 만들기**

1. question_create 함수 정의 – pybo/views.py

```
def question_create(request):
    # 질문 등록
    form = QuestionForm()
    context = {'form':form}
    return render(request, 'pybo/question_form.html', context)
```

2. 장고 폼 만들기 – pybo/forms.py(새로 생성)

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

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

froms.Model 폼 상속
- 모델 폼 객체를 저장하면 연결된 모델의 데이터를 저장할 수 있다.

- 내부 클래스로 반드시 Meta Class를 가져야 한다.

장고 폼으로 등록 페이지 만들기

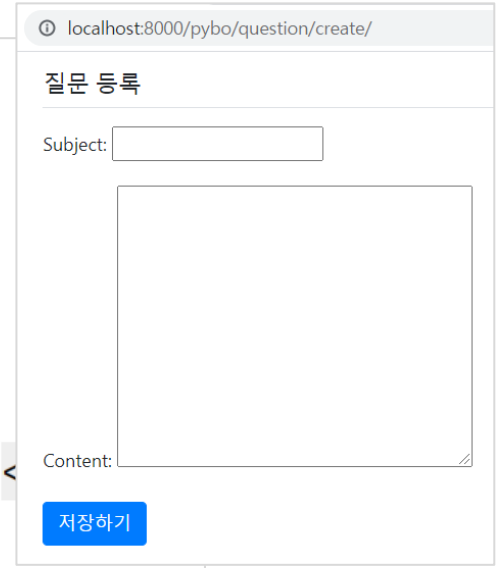
3. 질문 등록 URL 매핑하기 – pybo/views.py

```
urlpatterns = [
    ....
    path('/question/create/', views.question_create, name='question_create'),
]
```

장고 폼으로 등록 페이지 만들기

4. question_form.html 만들기

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



장고 폼으로 등록 페이지 만들기

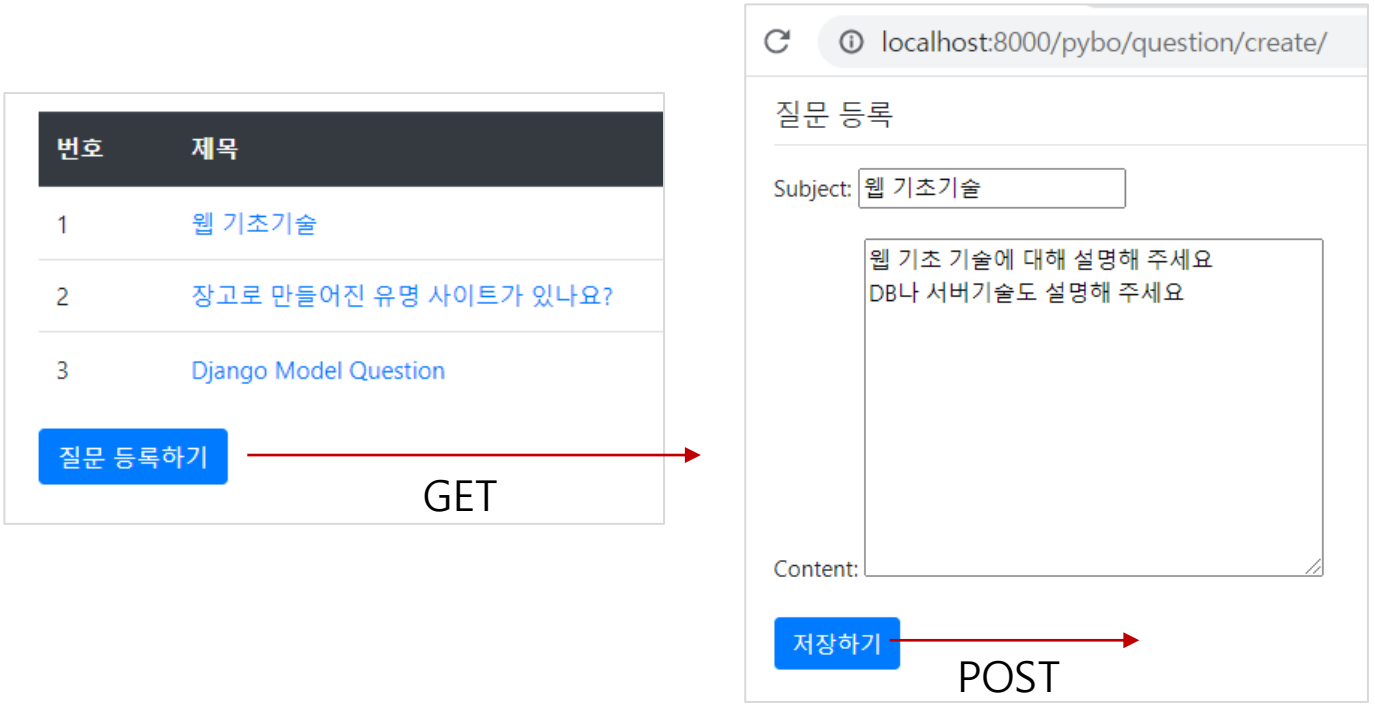
4. question_create 함수 완성하기

```
def question_create(request):
    # 질문 등록
    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('pybo:index')
    else:
        form = QuestionForm()
    context = {'form': form}
    return render(request, 'pybo/question_form.html', context)
```

request.method == 'GET"

GET 방식과 POST 방식 비교

➢ GET 방식과 POST 방식 비교



GET 방식 요청 : 질문 등록하기 -> question/create 페이지

POST 방식 요청: 폼에 내용 입력 -> 저장하기

Meta 클래스

- Meta 클래스

  클래스를 만드는 클래스라고 정의 할 수 있다.

```
# type()함수를 이용하여 동적으로 클래스 생성하기
Hello = type("Hello", (), {}) # 문자열, 튜플, 딕셔너리
print(Hello)

hello = Hello()
print(hello)
```

```
<class '__main__.Hello'>
<__main__.Hello object at 0x0000011681C53FC8>
```

Meta 클래스

- Meta 클래스

```
class MakeCalc(type):    # type을 상속받음
    # 새 클래스를 만들 때 호출되는 메서드
    def __new__(metacls, name, bases, namespace):
        # 새 클래스에 속성 추가
        namespace['desc'] = '계산 클래스'
        # 새 클래스에 메서드 추가
        namespace['add'] = lambda self, a, b : a + b
        # type의 __new__ 호출
        return type.__new__(metacls, name, bases, namespace)

Calc = MakeCalc('Calc', (), {})
c = Calc()
print(c.desc)
print(c.add(3, 4))
```

유효성 오류 처리

- label 이름 한글로 변경하기

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

답변 등록 폼 만들기

- **답변 등록 폼 만들기**

1. AnswerForm 클래스 정의 – pybo/forms.py(새로 생성)

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

답변 등록 페이지 만들기

2. 답변 등록 URL 매핑하기 – pybo/views.py

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

답변 등록 페이지 만들기

3. answer_create 함수 재정의 – pybo/views.py

```
def answer_create(request, question_id):
    # 답변 등록
    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.question = question
            answer.create_date = timezone.now()
            answer.save()
            return redirect('pybo:detail', question_id=question.id)
    else:
        form = AnswerForm()
    context = {'question':question, 'form': form}
    return render(request, 'pybo/question_detail.html', context)
```