

blog 사이트 만들기



블로그 포스트 페이지 만들기

블로그 게시글

- 곰 인형을 소개합니다.



- 무더위를 이기는 방법은 올림픽 경기를 관람하는 것
- 커피를 마시면 좋아요

글쓰기

블로그 포스트

제목:

사진: 선택된 파일 없음

blog 앱 프로젝트 생성하기

❖ blog 프로젝트 만들기

1. projects 디렉터리 하위에 blog 디렉터리에 만들기

```
(mysite) > C:\projects>mkdir blog
```

2. blog 디렉터리에 새 프로젝트 config 앱 설치하기

```
(mysite) > C:\projects\blog>django-admin startproject config .
```

blog 앱 프로젝트 생성하기

3. 서버를 구동해서 127.0.0.1:8000 페이지 확인

```
(mysite) > C:\projects\blog>python manage.py runserver
```

4. DB 사용을 위한 migrate 실행하기

```
(mysite) > C:\projects\blog>python manage.py migrate
```

5. 관리자 페이지 사용을 위한 슈퍼유저 생성하기

```
(mysite) > C:\projects\blog>python manage.py createsuperuser
```

blog 앱 프로젝트 생성하기

❖ config/settings.py에서 설정 변경하기

1. 언어와 시간 변경하기

```
LANGUAGE_CODE = 'ko-kr'  
  
TIME_ZONE = 'Asia/Seoul'
```

2. 'blog'앱 등록하기 – DB 사용을 위한 필수 설정

```
INSTALLED_APPS = [  
    'blog',  
    'django.contrib.admin',  
    'django.contrib.auth',  
]
```

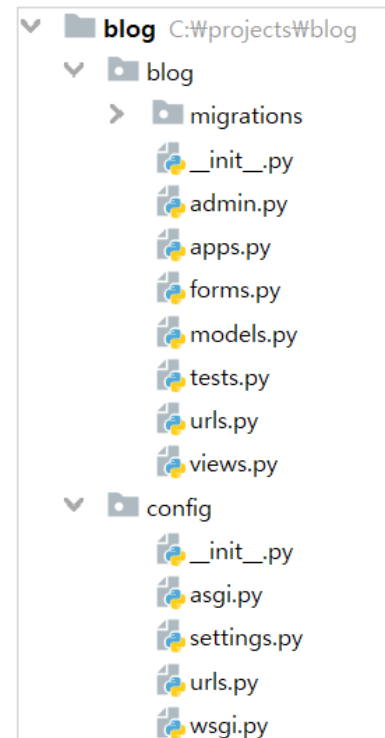
blog 앱 생성하기

❖ blog app(앱) 생성하기

1. blog프로젝트 하위에 blog 앱 생성하기

```
(mysite) > C:\projects\blog>django-admin startapp blog
```

2. blog앱 하위에 urls.py 생성하기



index 페이지 – 요청과 응답

❖ index 페이지 만들기

1. config/urls.py 경로 설정

```
from poll import views
urlpatterns = [
    path('admin/', admin.site.urls),
    path('blog/', include('blog.urls')),
]
```

2. blog/urls.py 작성

```
from poll import views
urlpatterns = [
    path("", views.index)
]
```

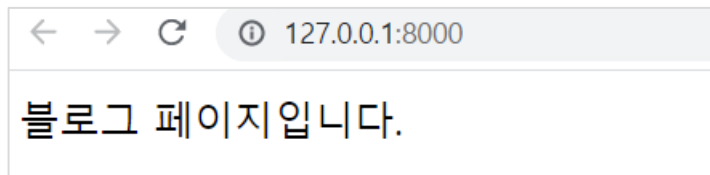
index 페이지 – 요청과 응답

❖ index 페이지 만들기

3. blog/views.py 작성

```
from django.http import HttpResponse  
def index(request):  
    return HttpResponse("블로그 페이지입니다.")
```

4. 127.0.0.1:8000 접속



DB 모델 만들기

▪ post 모델 만들기

1. blog/models.py에 Post 모델 생성

```
class Post(models.Model):  
    title = models.CharField(max_length=100)  
    photo = models.ImageField(upload_to="blog/images/%Y/%m/%d/",  
                              null=True, blank=True)
```

2. 변경된 DB 적용을 위한 makemigrations와 migrate 실행
3. Pillow 라이브러리 설치해야 함

```
(mysite) > C:\projects\blog>pip install Pillow
```

미디어 파일 관리하기

- 이미지 파일 업로드를 위한 설정

1. config/settings.py에 Post 모델 생성

```
import os
from pathlib import Path

STATIC_URL = '/static/'

MEDIA_URL = '/media/'
# 프로젝트 폴더 아래 media 폴더가 생성됨
MEDIA_ROOT = os.path.join(BASE_DIR, 'media')
```

미디어 파일 관리하기

- 이미지 파일 업로드를 위한 설정

2. config/urls.py에 미디어 경로 추가하기

```
from django.conf.urls.static import static
from django.conf import settings

urlpatterns = [
    path('admin/', admin.site.urls),
    path('blog/', include('blog.urls')),
]

urlpatterns += static(settings.MEDIA_URL,
                      document_root=settings.MEDIA_ROOT)
```

템플릿으로 index 페이지 만들기

- 템플릿(templates) 만들기

1. blog 프로젝트 하위에 **templates** 디렉토리를 바로 밑에 만든다.

```
(mysite) C:\projects\blog>mkdir templates
```

2. 템플릿 디렉토리 위치를 config/settings.py에 등록하기

```
TEMPLATES = [  
    .....  
    'DIRS': [BASE_DIR / 'templates'],  
    .....  
]
```

템플릿으로 index 페이지 만들기

3. index 함수 작성하기 - blog/views.py

```
def index(request):  
    # post(데이터) 목록  
    post_list = Post.objects.all()  
    context = {'post_list': post_list}  
    return render(request, 'blog/post_list.html', context)  
    #return HttpResponse("블로그 페이지입니다.")
```

템플릿으로 질문 목록 페이지 만들기

3. 템플릿 파일 만들기

– 경로 : C:/projects/polls/templates/blog/index.html

```
<body>
  <h3>블로그 게시물</h3>
  {% if post_list %}
  <ul>
    {% for post in post_list %}
    <li>{{ post.title }}</li>
    <div>
      {% if post.photo %}
      
      {% endif %}
    </div>
    {% endfor %}
  </ul>
  {% endif %}
  <a href="{% url 'blog:post_create' %}">글쓰기</a>
</body>
```

포스트 등록 폼 만들기

- 포스트 등록 폼 만들기

1. 포스트 등록 URL 매핑하기 – blog/urls.py

```
urlpatterns = [  
    ...  
    path('post/create/', views.post_create, name='post_create')  
]
```

포스트 등록 폼 만들기

2. post_create 함수 정의 – blog/views.py

```
def post_create(request):  
    if request.method == "POST":  
        form = PostForm(request.POST, request.FILES)  
        if form.is_valid():  
            form.save()  
            return redirect('blog:index')  
    else:  
        form = PostForm()  
    context = {'form': form}  
    return render(request, 'blog/post_form.html', context)
```


포스트 등록 폼 만들기

3. 포스트 등록 폼 만들기- blog/forms.py

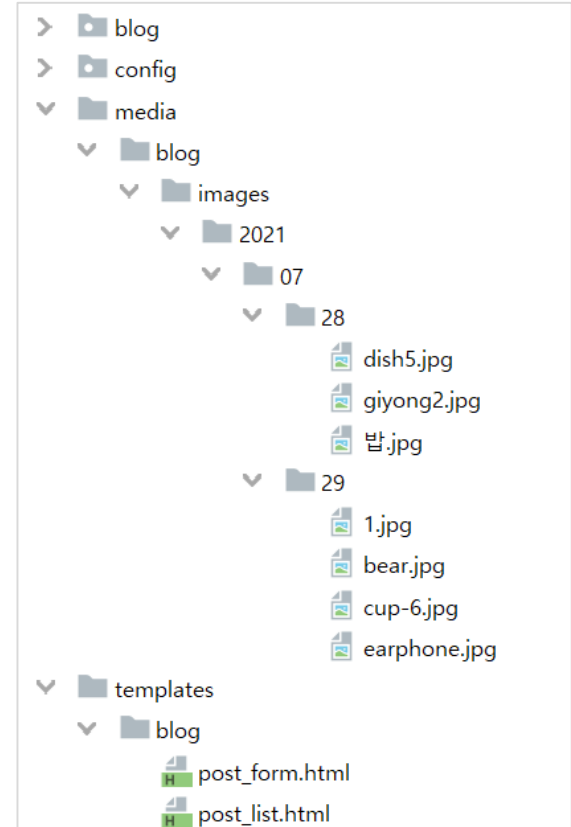
```
from django import forms
from blog.models import Post

class PostForm(forms.ModelForm):
    class Meta:
        model = Post
        fields = ['title', 'photo']
        labels = {
            'title': '제목',
            'photo': '사진'
        }
```

블로그 포스트

제목:

사진: 선택된 파일 없음



포스트 등록 폼 만들기

3. 포스트 등록 템플릿 만들기 – blog/post_form.html

```
<head>
  <meta charset="UTF-8">
  <title>글쓰기 창</title>
</head>
<body>
  <h5>글쓰기 창</h5>
  <form method="post" enctype="multipart/form-data">
    {% csrf_token %}
    {{ form.as_p }}
    <button type="submit">저장하기</button>
  </form>
</body>
```

포스트 상세 페이지 만들기

글쓰기

곰 인형을 소개합니다.



무더위를 이기는 방법은 올림픽 경기를

클릭 이동

← → ↻ ⓘ 127.0.0.1:8000/blog/9/

곰 인형을 소개합니다.



수정 삭제

포스트 상세 페이지 만들기

- 포스트 상세 페이지 만들기

1. 메인 템플릿에 상세페이지 링크 추가 – blog/post_list.html

```
<h4>
    <a href="{% url 'blog:post_create' %}">글쓰기</a>
</h4>
{% if post_list %}
    {% for post in post_list %}
        <h3><a href="{% url 'blog:detail' post.id %}">{{ post.title }}</a></h3>
        <div>
            {% if post.photo %}
                
            {% endif %}
        </div>
        <hr>
    {% endfor %}
</ul>
{% endif %}
```

포스트 상세 페이지 만들기

2. 상세페이지 URL 매핑 추가 – blog/urls.py

```
urlpatterns = [  
    ....  
    path('<int:post_id>/', views.detail, name='detail')  
]
```

3. detail 함수 추가 – blog/views.py

```
def detail(request, post_id):  
    post = Post.objects.get(id=post_id)  
    context = {'post': post}  
    return render(request, 'blog/post_detail.html', context)
```

포스트 상세 페이지 만들기

4. 상세페이지 템플릿 만들기 – blog/post_detail.html

```
<body>
  <h3>{{ post.title }}</h3>
  <div>
    
  </div>
  <div>
    <a href="#">수정</a>
    <a href="#">삭제</a>
  </div>
</body>
```

로그인 페이지 만들기

로그인 글쓰기

곰 인형을 소개합니다.



무더위를 이기는 방법은 올림픽 경기를

클릭 이동

← → ↻ ⓘ 127.0.0.1:8000/common/login/

사용자ID

비밀번호

common 앱 생성하기

❖ 회원관리를 위한 common app(앱) 생성하기

1. blog 프로젝트 하위에 common 앱 생성하기

```
(mysite) > C:\projects\blog>django-admin startapp common
```

2. common앱 등록 – config/setting.py

```
INSTALLED_APPS = [  
    'blog',  
    'common',  
    'django.contrib.admin',  
    'django.contrib.auth',
```

3. common앱 하위에 urls.py 생성하기

common 앱 urls.py 설정

1. config/urls.py 경로 설정

```
urlpatterns = [  
    path("", views.index),  
    path('admin/', admin.site.urls),  
    path('blog/', include('blog.urls')),  
    path('common', include('common.urls')),  
]
```

2. common/urls.py 작성

```
from django.contrib.auth import views as auth_views  
app_name = 'common'  
urlpatterns = [  
    path('login/', auth_views.LoginView.as_view(  
        template_name = 'common/login.html'), name='login'),  
]
```

login 템플릿 만들기

- login 템플릿 만들기 – templates/common/login.html

```
<form action="{% url 'common:login' %}" method="post">
    {% csrf_token %}
    <p><label for="username">사용자ID</label>
    <input type="text" name="username" id="username"
        value="{% form.username.value|default_if_none:'' %}"></p>
    <p><label for="password">비밀번호</label>
    <input type="password" name="password" id="password"
        value="{% form.password.value|default_if_none:'' %}"></p>
    <button type="submit">로그인</button>
</form>
```