MyPet Website

오픈소스SW활용 1분반

Dpt. of Mobile System Engineering
32192530 양윤성

2023.05.12.

● 목차

1. 서비스 기능 소개

MyPet 웹사이트는 제 애완묘인 "루티"를 소개하는 홈페이지입니다. 마침 기말 프로젝트를 위해 Django, HTML, CSS, Bootstrap을 공부할 필요가 있었는데, 이번 과제가 마침 이들을 연습하기 좋은 기회라 생각했습니다. 지금까지 Front-End 관련 내용들을 이론적으로만 배우고 한번도 실습을 해보지 않았기 때문입니다. 간단한 MyPet 소개 홈페이지를 만들어보며 HTML, CSS 파일 작성법 및 프론트엔드 관련 오픈소스인 Bootstrap 사용법을 깨닫고, Django로 홈페이지를 어떻게 구성하고 만들고 배포하는지 배우는 것을 목표로 구현했습니다.

홈페이지에 접속하면 가장 먼저 index.html이 표시됩니다. 이 페이지는 MyPet 홈페이지의 메인 화면으로, MyPet 사이트를 간단하게 소개하고, 제 애완묘인 "루티"를 더 자세히 알아볼 수 있는 시작버튼과 루티의 사진들 및 정보들을 더 자세히 볼 수 있는 공식 인스타그램으로 연결되는 버튼을 구현했습니다. 메인 화면의 시작 버튼을 누르거나 네비게이션 바의 "Intro"를 클릭하면 intro.html이 표시됩니다. 여기에는 루티 보호자인 저에 대한 정보와 루티의 정보를 소개합니다. 제 소개는 이름, 학교, 학과로 간단하게 하고 연락을 할 수 있도록 이메일과 인스타그램 아이디를 추가로 표기했습니다. 루티 소개는 고양이 종과 성별, 생년월일, 입양해온 날짜, 공식 인스타그램 아이디를 표기했습니다. 루티 소개 파트에서 "Check My Life"를 클릭하거나 네비게이션 바의 "Daily"를 누르면 daily.html이 표시됩니다. Daily 페이지는 루티에 대해 더욱 깊게 소개합니다. 하루 일과표, 좋아하는 것, 싫어하는 것, 좋아하는 음식, 취미 및 특기, 기타 특징들을 소개하여 사람들이 루티를 상세히 알 수 있도록 합니다. 네비게이션 바의 "Photo"를 누르면 마지막 화면인 photo.html이 표시됩니다. 여기에는 루티의 사진들과 각각 한줄 설명이 담긴 루티 갤러리입니다. 이 홈페이지에 나오는 사진들은 관리자 페이지에서 사진과 한줄 설명을 기입하며 데이터베이스에 저장하면 설명과 함께 사진이 추가됩니다. 지금까지 간략히 소개한 내용들은 이후 UI스케치 및 실제 구현 사진, 코드 원리를 포함하여 보다 자세히 설명할 것입니다.

2. 설계

2-1. UI 스케치

그림1. 메인 페이지

그림2. 소개(Intro) 페이지

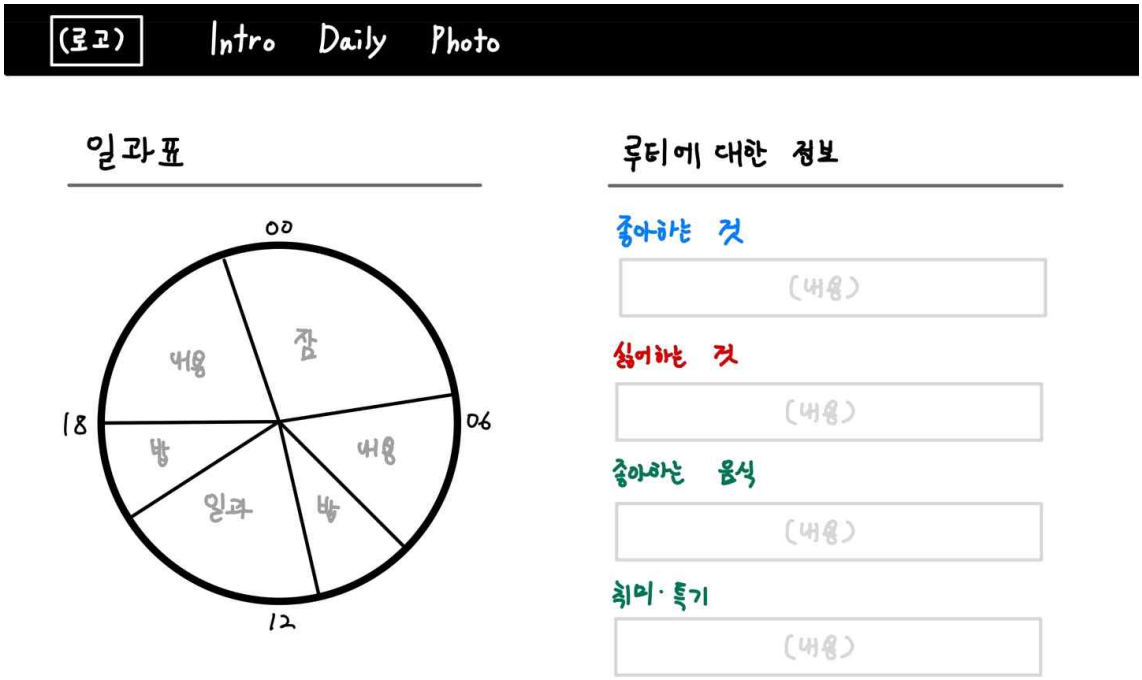


그림3. 일과(Daily) 페이지

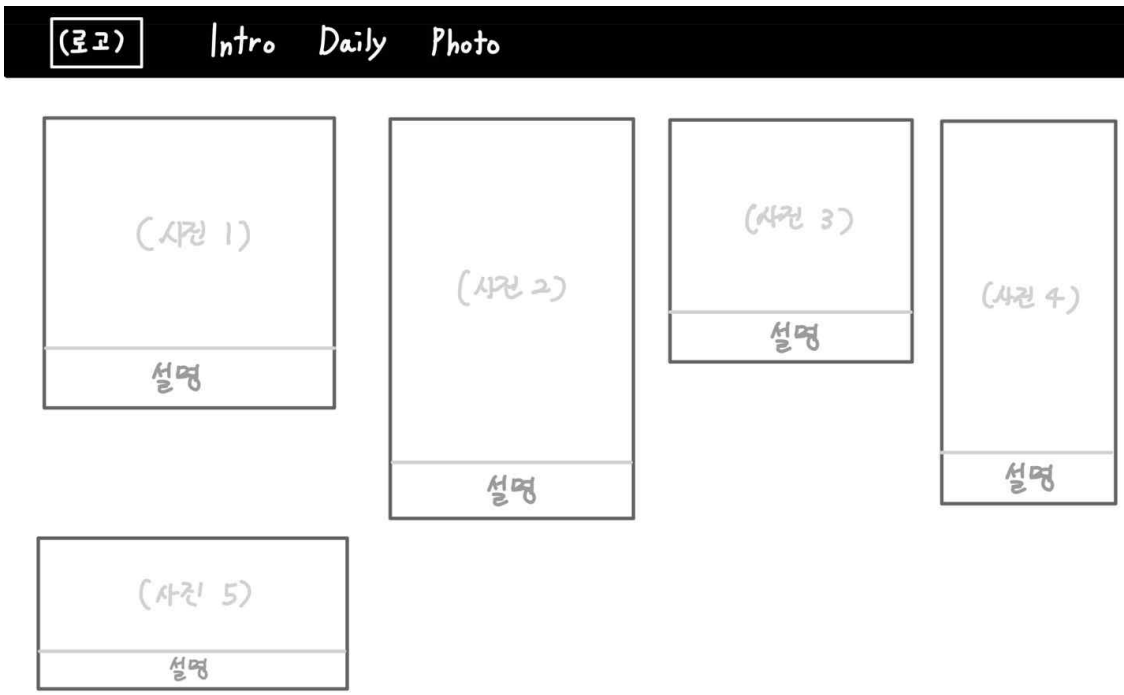


그림4. 갤러리(Photo) 페이지

2-2. 개발환경 및 사용 오픈소스

그림5. 사용할 Tech Stack

이번 프로젝트는 기말 프로젝트를 위한 연습이라는 목표도 있기 때문에 기말 프로젝트를 계획할 시 예정했던 개발 환경을 똑같이 맞춰 진행했습니다. Windows 10 Education x64 OS에서 진행하며, 개발 환경은 IDE로는 Visual Studio Code 1.76.2를, 웹 프레임워크로는 Django 4.1.4로 똑같이 사용합니다. VS Code 내에서 Python 확장자를 따로 설치하고, 터미널에서 Django 4.1.4를 pip install로 설치하여 Python과 Django 명령어가 사용 가능하게 개발 환경을 구축했습니다. 웹 사이트를 디자인하는 데에 HTML와 CSS 파일을 작성했습니다. HTML로 웹사이트의 구조를 작성하고 CSS 파일에 HTML 파일을 위한 디자인을 정의하여 MyPet Website를 구현했습니다. 일부 디자인은 Bootstrap 오픈소스를 가져와 사용했습니다. Bootstrap은 오픈소스 프론트엔드 프레임워크로, 웹사이트를 디자인하고 구현하는 데에 필요한 컴포넌트를 개발자에게 제공하여 시간과 비용을 절약하게 해줍니다. 특히 Bootstrap만의 쉬운 CSS Class 기능과 JavaScript 플러그인을 사용하여 사용자 정의 디자인과 동작을 구현할 수 있다는 특징이 있습니다. Django는 후술할 2-3에서 설명합니다.

2-3. Django

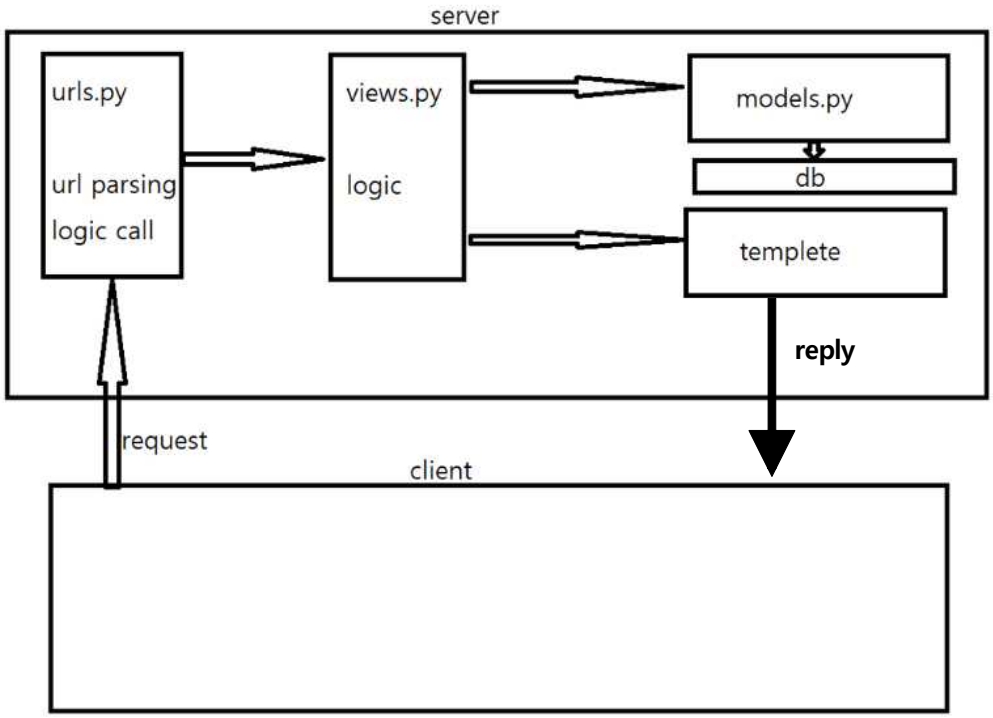
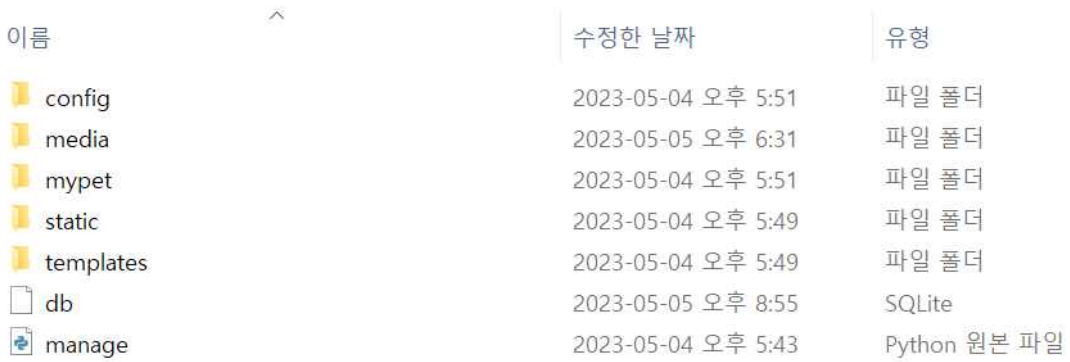


그림6. Django의 전체적인 흐름도

Django는 Python으로 작성 가능한 무료 오픈소스 웹 프레임워크로 데이터베이스, URL 라우팅, 템플릿, 관리자 페이지 등 다양한 통합 개발 환경을 제공합니다. 장고는 MTV 패턴으로 구성되는데, M은 Model(Database), T는 Template(HTML), V는 View(Logic)입니다. 이 3가지를 Django가 묶어서 웹사이트를 실행할 수 있게 됩니다. 우선 client(사용자)가 특정 사이트에 접속하면 client가 서버에 request(요청)을 보내게 되고 서버는 이를 받고 확인합니다. 요청에 따라 Django 내의 urls.py에서 어떤 url인지 판단하여 views.py와 연계합니다. views.py에서는 해당 url과 맞는 Logic을 결정하게 됩니다. 이 Logic에 따라 client에게 보내는 응답에서의 html 파일과 방식이 모두 다릅니다. html 파일을 찾을 때는 Django 내의 template 디렉토리에서 찾으며, 데이터는 models.py에서 정의된 모델을 활용합니다. Reply(응답)은 Reply Header와 Reply Body로 구성되는데, Header는 Django에서 자동으로 설정되고 Body에 디자인한 HTML파일 등이 들어갑니다.

3. 구현

3-1. 각 디렉토리 소개



이름	수정한 날짜	유형
config	2023-05-04 오후 5:51	파일 폴더
media	2023-05-05 오후 6:31	파일 폴더
mypet	2023-05-04 오후 5:51	파일 폴더
static	2023-05-04 오후 5:49	파일 폴더
templates	2023-05-04 오후 5:49	파일 폴더
db	2023-05-05 오후 8:55	SQLite
manage	2023-05-04 오후 5:43	Python 원본 파일

그림7. Django 프로젝트 구성

Django를 시작할 때, 터미널에 django-admin startproject config . 명령어를 입력하여 프로젝트를 만들고 django-admin startapp mypet 으로 config 프로젝트 내에 mypet 앱을 만들었습니다. 그 외에 직접 templates, static, media 디렉토리를 만들었습니다. config 디렉토리에는 프로젝트 관련한 전체적인 설정을 할 수 있는 파이썬 파일들이 담겨져 있고, mypet에는 웹사이트와 관련된 logic들을 처리하는 views.py와 웹사이트의 데이터를 저장하기 위한 데이터베이스 모델이 설계된 models.py등이 있습니다. templates에는 HTML파일들을, static에는 부트스트랩을 포함한 여러 css파일들이 있습니다. 마지막으로 media는 데이터베이스에서 저장된 사진을 HTML로 불러오기 위한 디렉토리이며 model에 저장된 사진들이 이 디렉토리에 위치합니다. mypet 앱과 templates, static, media 디렉토리를 사용하기 위해서 config/settings.py에서 아래와 같은 추가적인 설정을 해주었습니다.

```python
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'mypet.apps.MypetConfig',
]
```

```python
TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.
        'DIRS': [BASE_DIR / 'templates'],
STATIC_URL = 'static/'
STATICFILES_DIRS = [
    BASE_DIR / 'static',
]
# 이미지 불러오기를 위한 media_url
MEDIA_URL = '/media/'
MEDIA_ROOT = os.path.join(BASE_DIR,'media')
```

그림8. Django 프로젝트 Setting

3-2. urls.py & views.py

```python
from django.contrib import admin
from django.urls import path, include

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

그림9. config/urls.py

```python
from django.urls import path
from django.conf.urls.static import static
from django.conf import settings
from . import views

app_name = 'mypet'

urlpatterns = [
    path('', views.index, name='index'),
    path('intro/', views.intro, name='intro'),
    path('daily/', views.daily, name='daily'),
    path('photo/', views.photo, name='photo'),
] + static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)
```

그림10. mypet/urls.py

```python
from django.shortcuts import render
from .models import Photo

def index(request):
    return render(request, 'mypet/index.html')

def intro(request):
    return render(request, 'mypet/intro.html')

def daily(request):
    return render(request, 'mypet/daily.html')

def photo(request):
    photo_list = Photo.objects.all()
    context = {'photo_list': photo_list}
    return render(request, 'mypet/photo.html', context)
```

그림11. mypet/views.py

프로젝트 urls.py에서 각 앱의 views.py로 바로 연결할 수도 있지만, 확장성 및 분리 관리를 위하여 mypet app에 urls.py를 추가한 다음, include를 이용하여 mypet의 urls.py로 연결하게 했습니다. mypet의 urls.py에는 각 logic에 맞는 views.py의 function을 찾아 실행됩니다. function은 각 페이지 별로 index, intro, daily, photo가 있는데 photo에는 후에 설명할 Photo 모델 객체들을 불러와 페이지에 표시해야 하기 때문에, Photo.objects.all()로 모든 객체들을 불러와 context에 담고 이를 photo.html에 담아 client에게 reply합니다.

3-3. base.html

```
<nav class="navbar navbar-expand-lg navbar-dark" style="background-color: #0c0902;">
    <div class="container">
        <a id="logo" class="navbar-brand" href="{% url 'mypet:index' %}">
            <img width="110%" height="110%" src="{% static 'images/ruti_logo.jpg' %}"/>
        </a>
        <button class="navbar-toggler" type="button" data-bs-toggle="collapse"
            data-bs-target="#navbarSupportedContent" aria-controls="navbarSupportedContent"
            aria-expanded="false" aria-label="Toggle navigation">
            <span class="navbar-toggler-icon"></span>
        </button>
        <div class="collapse navbar-collapse" id="navbarSupportedContent">
            <ul class="navbar-nav mx-5 mb-2 mb-lg-0 navs">
                <li class="nav-item mx-1"><a class="nav-link active text-white" href="{% url 'mypet:intro' %}">
                    Intro</a></li>
                <li class="nav-item mx-1"><a class="nav-link active text-white" href="{% url 'mypet:daily' %}">
                    Daily</a></li>
                <li class="nav-item mx-1"><a class="nav-link active text-white" href="{% url 'mypet:photo' %}">
                    Photos</a></li>
            </ul>
        </div>
    </div>
</nav>
```

그림12. base.html의 네비게이션바 부분

base.html에서는 HTML5의 기본 구조 및 각종 라이브러리 호출, 네비게이션 바를 작성했습니다. 이 파일의 목적은 각 html 파일에서 공통적인 부분을 넣기 위함입니다. HTML5 기본 구조는 모두 공통적으로 들어가기에 html 파일마다 해당 부분이 들어가면 코드가 다소 길어질 수 있습니다. 그러므로 이를 base.html로 따로 분리한 후 static 문법을 이용하여 각 파일에서 {% extends 'base.html' %}로 상속받아 사용했습니다. 네비게이션 바는 nav 속성과 부트스트랩의 nav, navbar 관련 클래스를 사용하여 구현했습니다. 네비게이션 바의 가장 왼쪽의 고양이 로고를 누르면 메인 페이지로 이동하고 Intro, Daily, Photos는 각각의 페이지로 연결됩니다.

3-4. index, intro, daily.html

```
{% extends 'base.html' %}
{% load static %}
{% block content %}
<div class="container my-4 mt-5">
    <img class="rounded mx-auto d-block" width="30%" height="30%" src="{% static 'images/ruti_main.jpg' %}"/>
</div>
<div class="container my-3">
    <p id="title" class="text-center">My Pet Ruti!</p>
</div>
<div class="container my-3">
    <p class="text-center content">Ruti is very cute cat in our house.<br>
        We introduce Ruti and provide information&amp;phothos. <br>
        Read more deatiled with navbar above or start button below.
    </p>
</div>
<br>
<div class="d-flex justify-content-center">
    <a href="https://www.instagram.com/lutyvely/" target="_blank">
        <button type="button" class="btn btn-lg btn-secondary mx-5 index-btn">
            <i class="fa fa-instagram"></i>  Instagram
        </button>
    </a>
    <a href="{% url 'mypet:intro' %}">
        <button type="button" class="btn btn-lg btn-dark mx-5 index-btn">Get Started</button>
    </a>
</div>
{% endblock %}
```

그림13. index.html

메인 페이지를 나타내며, 간단한 사진 및 설명과 함께 실제 운영하는 인스타그램 사이트로 접속할 수 있는 버튼과 시작 버튼을 구현했습니다. 여기서 밑에 i속성은 아이콘으로, base.html에서 fontawesome을 불러와 fa 클래스를 사용했습니다. 시작 버튼을 누르면 intro.html로 연결됩니다. 인스타그램 사이트는 로그인을 해야 접속되기에 아이디가 없으시다면 아래 사진을 확인하시면 됩니다. intro.html와 daily.html 역시 비슷한 구조이지만 전체적인 화면을 둘로 분할하여 구현했다는 점이 다릅니다.

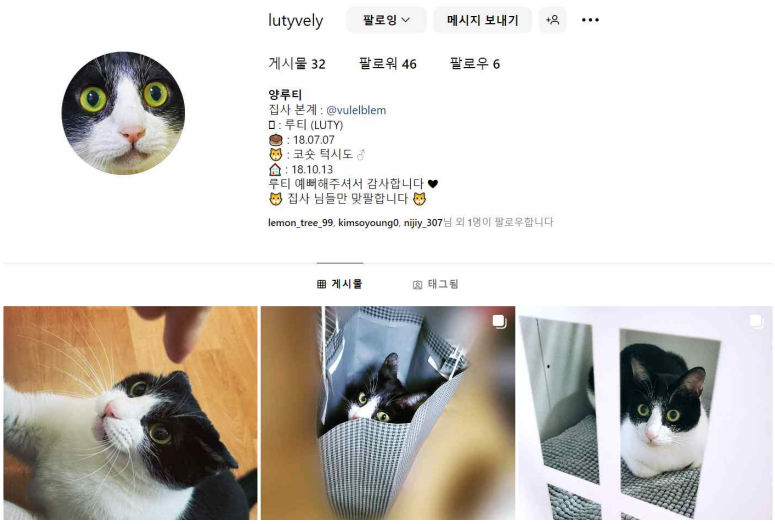


그림14. 루티 공식 인스타그램 페이지 (가족이 관리 중)

3-5. photo.html

```
{% extends 'base.html' %}
{% load static %}
{% block content %}
<div class="album py-4 mt-1 bg-light">
    <div class="container">
        <div class="row row-cols-1 row-cols-sm-2 row-cols-md-4 g-3">
            {% for photo in photo_list %}
            <div class="col">
                <div class="card shadow-sm fs-5 nav-link text-dark border border-3 border-drak">
                    <img class="rounded" width="100%" height="100%" src="/media/{{photo.image}}">
                    <div class="card-body">
                        <div class="d-flex justify-content-between align-items-center">
                            <small class="photo-content card-text fs-5">{{ photo.detail }}</small>
                        </div>
                    </div>
                </div>
            </div>
            {% endfor %}
        </div>
    </div>
</div>
{% endblock %}
```

그림15. photo.html

views.py에서 context에 photo_list를 넣었기에 photo.html에서 for문을 활용하여 photo_list에 있는 모든 객체들을 표시할 수 있습니다. 각 객체들은 card속성으로 표시되며 카드의 가운데에는 객체별 고양이 사진이, 카드의 아래쪽에는 객체별 사진 설명이 담겨져 있습니다. 객체의 사진들은 3-1에서 설명했던 media 디렉토리에 저장되므로 이미지를 불러오는 경로를 media로 설정해 줍니다.

3-6. style.css

mypet 웹사이트를 디자인할 때 부트스트랩으로 구현하기에 한계가 있는 부분을 css파일을 정의하여 만들었습니다. 우선 Google Font를 사용하기 위해 @import로 특정 글꼴들을 불러온 다음, font-family로 설정했습니다. 또한 font-size, font-weight 등 글꼴과 관련된 속성들은 직접 사용자 설정을 해주는 것이 더 디테일하고 편하므로 부트스트랩을 이용하지 않았습니다. 이 외에도 네비게이션 바의 로고나 버튼 등의 너비나 높이를 직접 정해주기 위해 width나 height를 따로 정해준 경우도 있습니다.

3-7. models.py

```python
from django.db import models

# Create your models here.
class Photo(models.Model):
    image = models.ImageField(blank=True, upload_to='media/') # 사진
    detail = models.CharField(max_length=200) # 사진 설명
    def __str__(self):
        return self.detail
```

그림16. models.py

models.py에 루티의 사진을 불러오기 위한 객체인 Photo를 정의했습니다. image 필드는 사진을 넣어두는 필드로, 프로젝트 내의 media 디렉토리 내에 저장됩니다. detail은 사진에 대한 설명을 넣는 필드로 최대 길이가 200장입니다. __str__ function은 아래에서 설명할 관리자 페이지에서 설명으로 사진을 검색하게 하기 위해 만들어 주었습니다. 이 model은 터미널에서 python manage.py makemigrations와 python manage.py migrate를 입력하여 모델을 최종 정의해 줍니다.

3-8. 관리자 페이지

```python
from django.contrib import admin
from .models import Photo
# Register your models here.

class PhotoAdmin(admin.ModelAdmin):
    search_fields = ['detail']

admin.site.register(Photo, PhotoAdmin)
```

그림17. mypet/admin.py

프로젝트 터미널에서 python manage.py createsuperuser 명령어를 입력하면 superuser을 만들고 Django의 관리자 페이지가 만들어집니다. mypet의 superuser ID는 abee3417, PW는 1111입니다. 로컬 서버 주소에 /admin을 붙이면 관리자 페이지로 이동하여 로그인할 수 있습니다. 또한 mypet의 admin.py에 위 사진처럼 권한을 설정해주면 관리자 페이지에서 Photo객체를 사진, 설명과 함께 추가하거나 수정, 삭제 등이 가능합니다. 관리자 페이지는 아래 12쪽부터 확인할 수 있습니다.

4. 테스트 결과

그림18. 관리자페이지 메인화면

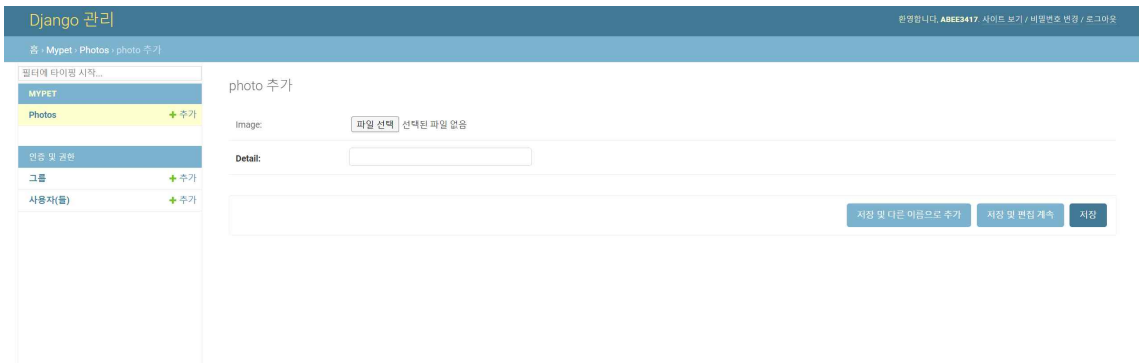


그림19. 관리자페이지 객체 추가 화면

그림20. 관리자페이지 객체 관리 화면

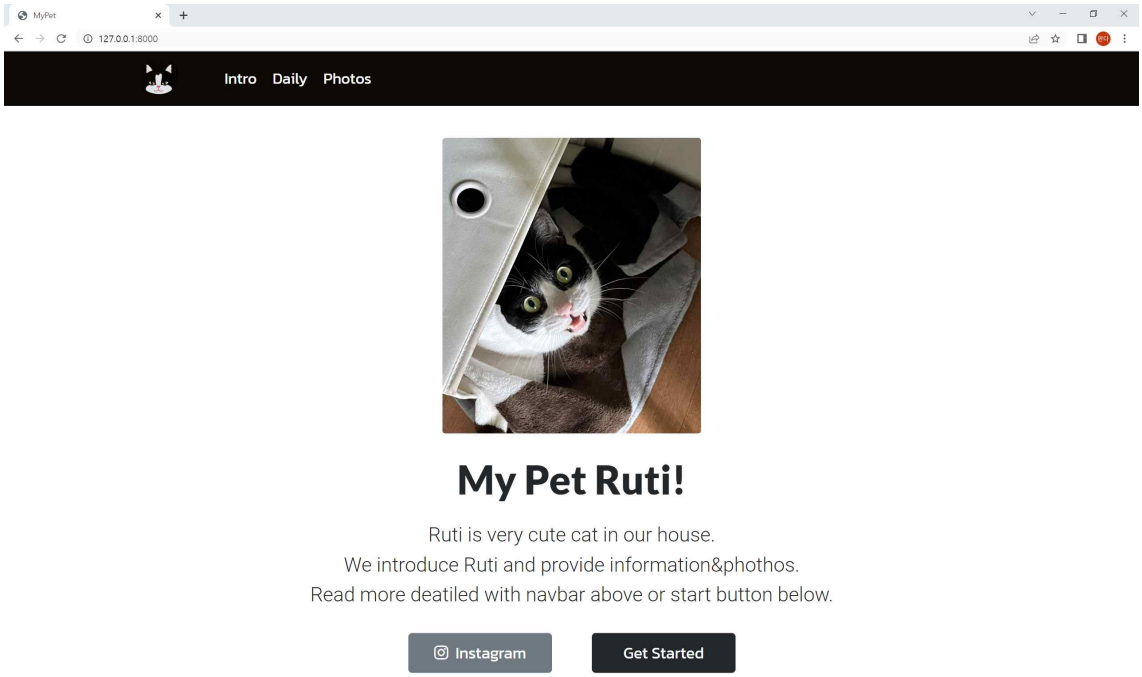


그림21. index.html 화면

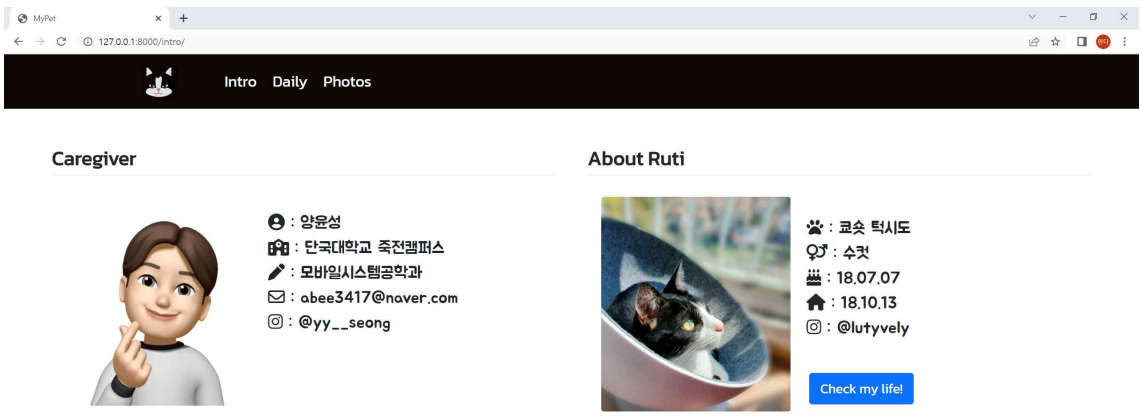


그림22. intro.html 화면

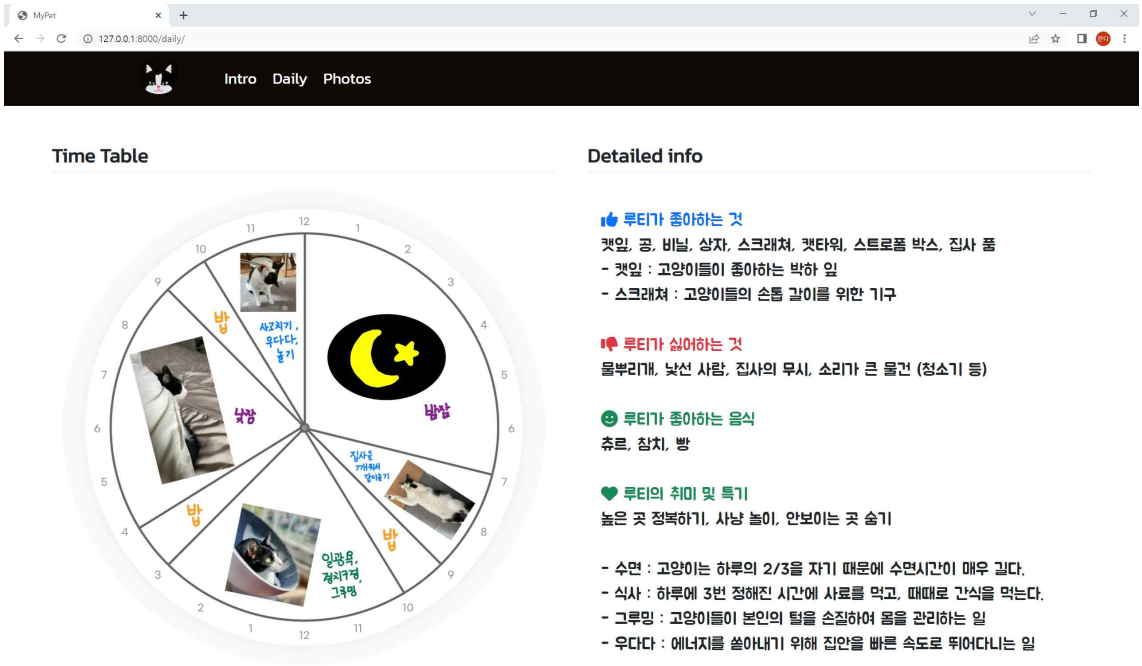


그림23. daily.html 화면

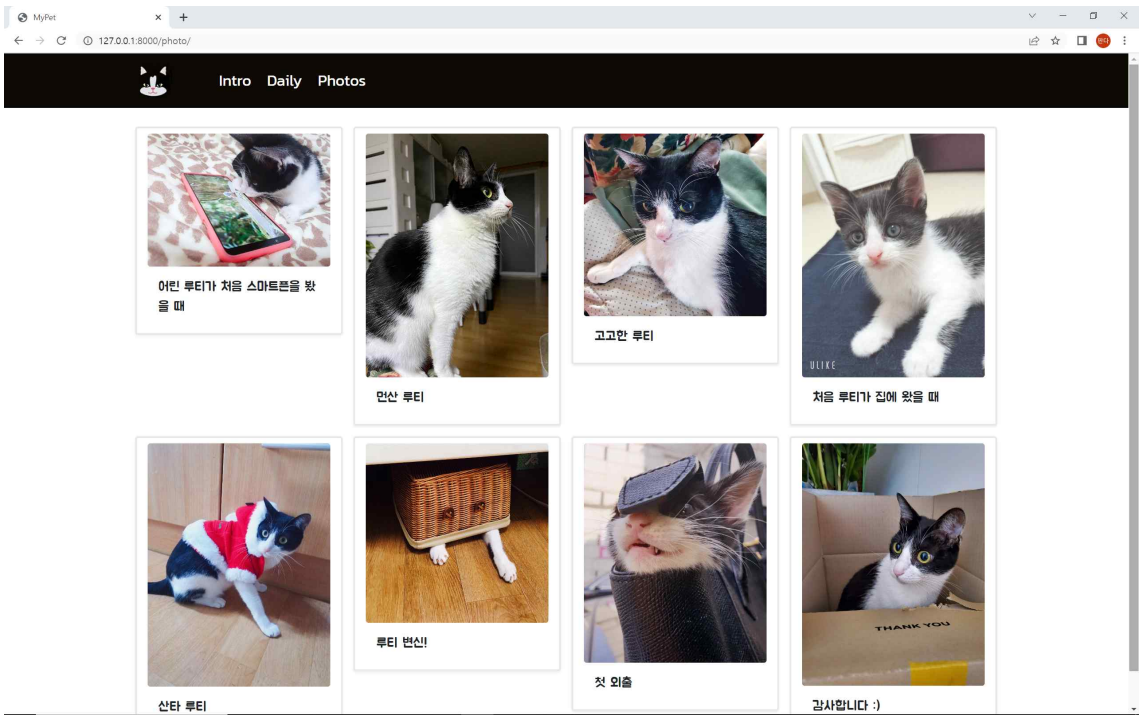


그림24. photo.html 화면

5. 후기

대학교를 다니며 웹 프레임워크와 실제 오픈소스를 활용해보는 건 처음이었습니다. 1, 2학년동안 VS나 VS Code로 기본적인 코딩 및 단순 프로젝트만 진행해 보고, Android studio로 간단한 어플을 만들어본 것이 전부였습니다. 그러나 수업을 듣고 mypet 홈페이지를 직접 구현 해보며 웹 개발이 어떤 방식으로 진행해야 하는지에 대한 흐름을 알 수 있었습니다. Django를 사용하고 Bootstrap을 활용하는 방법을 공부했으므로 후에 있을 기말 프로젝트나 더 나아가 현업에서도 이번 과제가 확실한 밑거름이 될 거라 생각합니다.