민선아@DESKTOP-0MVMCF2 MINGW64 ~/Desktop/멋쟁이사자처럼9기/studywithLikeLion/Classlion/Model실습 (master)

$ source myvenv/Scripts/activate

bash: myvenv/Scripts/activate: No such file or directory

민선아@DESKTOP-0MVMCF2 MINGW64 ~/Desktop/멋쟁이사자처럼9기/studywithLikeLion/Classlion/Model실습 (master)

$ python -m venv myvenv

민선아@DESKTOP-0MVMCF2 MINGW64 ~/Desktop/멋쟁이사자처럼9기/studywithLikeLion/Classlion/Model실습 (master)

$ source myvenv/Scripts/activate

(myvenv)

민선아@DESKTOP-0MVMCF2 MINGW64 ~/Desktop/멋쟁이사자처럼9기/studywithLikeLion/Classlion/Model실습 (master)

$ pip install django

Collecting django

Using cached Django-3.2.3-py3-none-any.whl (7.9 MB)

Collecting pytz

Using cached pytz-2021.1-py2.py3-none-any.whl (510 kB)

Collecting sqlparse>=0.2.2

Using cached sqlparse-0.4.1-py3-none-any.whl (42 kB)

Collecting asgiref<4,>=3.3.2

Using cached asgiref-3.3.4-py3-none-any.whl (22 kB)

Installing collected packages: sqlparse, pytz, asgiref, django

Successfully installed asgiref-3.3.4 django-3.2.3 pytz-2021.1 sqlparse-0.4.1

WARNING: You are using pip version 21.1.1; however, version 21.1.2 is available.

You should consider upgrading via the 'c:\users\민용진\desktop\멋쟁이사자처럼9기\studywithlikelion\classlion\model실습\myvenv\scripts\python.exe -m pip install --upgrade pip' command.

(myvenv)

민선아@DESKTOP-0MVMCF2 MINGW64 ~/Desktop/멋쟁이사자처럼9기/studywithLikeLion/Classlion/Model실습 (master)

$ django-admin startproject lionproject

(myvenv)

민선아@DESKTOP-0MVMCF2 MINGW64 ~/Desktop/멋쟁이사자처럼9기/studywithLikeLion/Classlion/Model실습 (master)

$ ls

lionproject myvenv

(myvenv)

민선아@DESKTOP-0MVMCF2 MINGW64 ~/Desktop/멋쟁이사자처럼9기/studywithLikeLion/Classlion/Model실습 (master)

$ cd lionproject/

(myvenv)

민선아@DESKTOP-0MVMCF2 MINGW64 ~/Desktop/멋쟁이사자처럼9기/studywithLikeLion/Classlion/Model실습/lionproject (master)

$ ls

lionproject manage.py

(myvenv)

민선아@DESKTOP-0MVMCF2 MINGW64 ~/Desktop/멋쟁이사자처럼9기/studywithLikeLion/Classlion/Model실습/lionproject (master)

$ python manage.py startapp blog

(myvenv)

민선아@DESKTOP-0MVMCF2 MINGW64 ~/Desktop/멋쟁이사자처럼9기/studywithLikeLion/Classlion/Model실습/lionproject (master)

$

<models.py>

from django.db import models

# Create your models here.

class Blog(models.Model):

    title = models.CharField(max\_length=200)

    writer = models.CharField(max\_length=100)

    pub\_date = models.DateTimeField()

    body = models.TextField()

    def \_\_str\_\_(self):

        return self.title

class 만들면서 model에서 상속받음

민선아@DESKTOP-0MVMCF2 MINGW64 ~/Desktop/멋쟁이사자처럼9기/studywithLikeLion/Classlion/Model실습 (master)

$ cd lionproject

(myvenv)

민선아@DESKTOP-0MVMCF2 MINGW64 ~/Desktop/멋쟁이사자처럼9기/studywithLikeLion/Classlion/Model실습/lionproject (master)

$ ls

blog lionproject manage.py

(myvenv)

민선아@DESKTOP-0MVMCF2 MINGW64 ~/Desktop/멋쟁이사자처럼9기/studywithLikeLion/Classlion/Model실습/lionproject (master)

$ python manage.py makemigrations

Migrations for 'blog':

blog\migrations\0001\_initial.py

- Create model Blog

🡪 앱 내의 migration 폴더를 만들어서 models.py의 변경사항 저장

민선아@DESKTOP-0MVMCF2 MINGW64 ~/Desktop/멋쟁이사자처럼9기/studywithLikeLion/Classlion/Model실습/lionproject (master)

$ python manage.py migrate

Operations to perform:

Apply all migrations: admin, auth, blog, contenttypes, sessions

Running migrations:

Applying blog.0001\_initial... OK

* Migration 폴더를 실행시켜 데이터베이스에 적용

Operations to perform:

Apply all migrations: admin, auth, contenttypes, sessions

Running migrations:

Applying contenttypes.0001\_initial... OK

Applying auth.0001\_initial... OK

Applying admin.0001\_initial... OK

Applying admin.0002\_logentry\_remove\_auto\_add... OK

Applying admin.0003\_logentry\_add\_action\_flag\_choices... OK

Applying contenttypes.0002\_remove\_content\_type\_name... OK

Applying auth.0002\_alter\_permission\_name\_max\_length... OK

Applying auth.0003\_alter\_user\_email\_max\_length... OK

Applying auth.0004\_alter\_user\_username\_opts... OK

Applying auth.0005\_alter\_user\_last\_login\_null... OK

Applying auth.0006\_require\_contenttypes\_0002... OK

Applying auth.0007\_alter\_validators\_add\_error\_messages... OK

Applying auth.0008\_alter\_user\_username\_max\_length... OK

Applying auth.0009\_alter\_user\_last\_name\_max\_length... OK

Applying auth.0010\_alter\_group\_name\_max\_length... OK

Applying auth.0011\_update\_proxy\_permissions... OK

Applying auth.0012\_alter\_user\_first\_name\_max\_length... OK

Applying sessions.0001\_initial... OK

민선아@DESKTOP-0MVMCF2 MINGW64 ~/Desktop/멋쟁이사자처럼9기/studywithLikeLion/Classlion/Model실습/lionproject (master)

$ python manage.py runserver

Watching for file changes with StatReloader

Performing system checks...

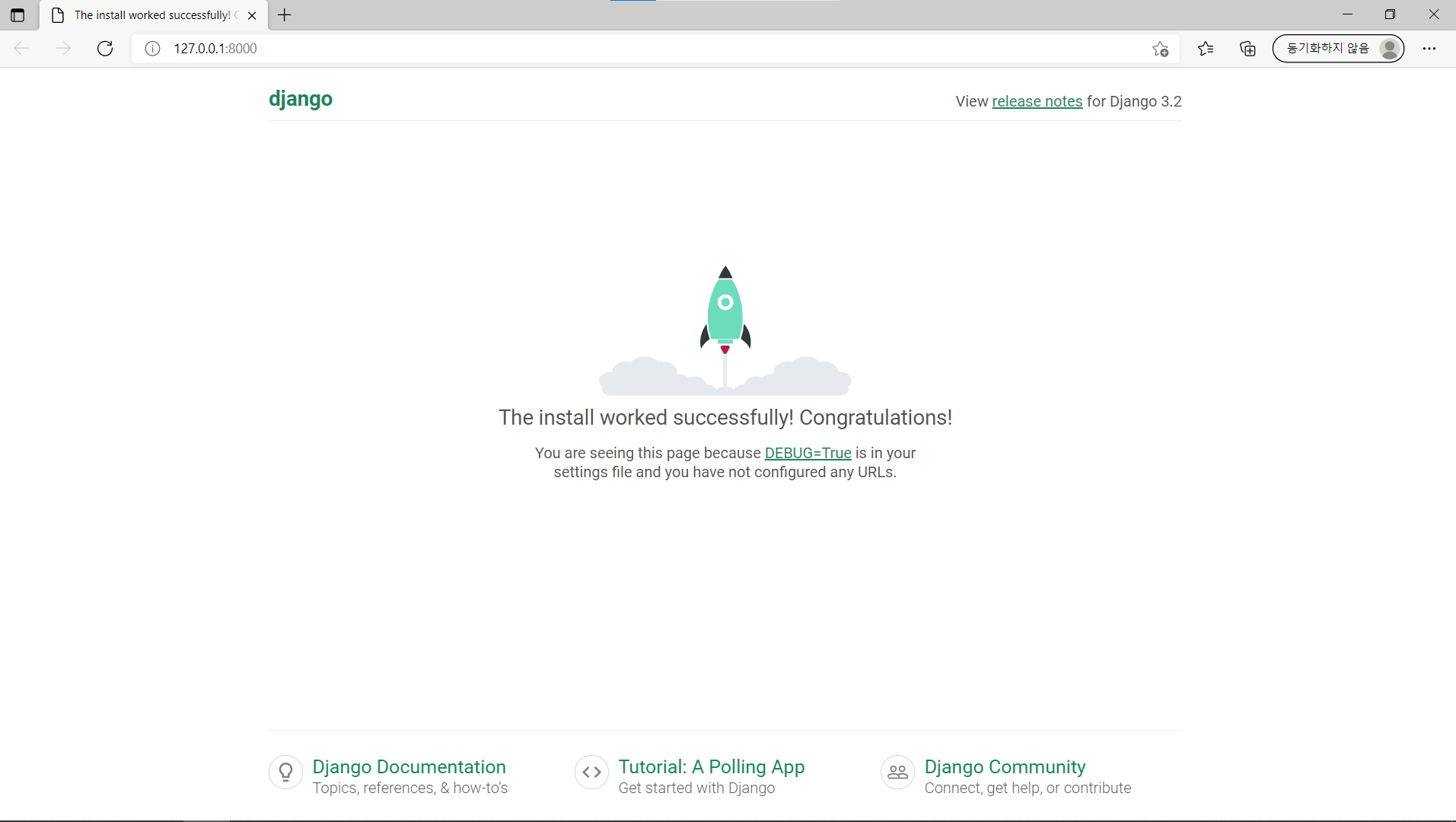
System check identified no issues (0 silenced).

June 01, 2021 - 19:52:54

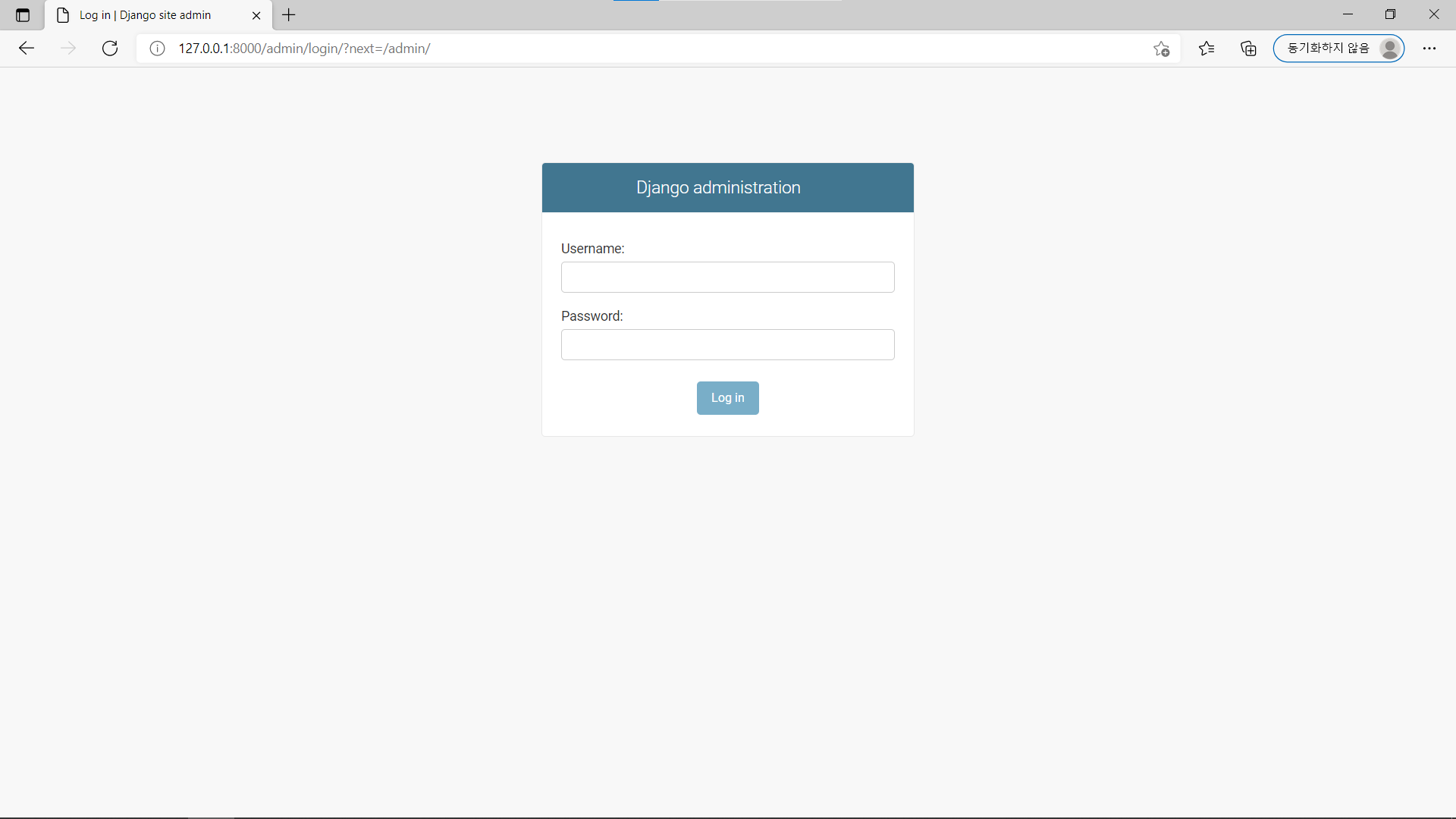
Django version 3.2.3, using settings 'lionproject.settings'

Starting development server at http://127.0.0.1:8000/

Quit the server with CTRL-BREAK.



/admin 치면



민선아@DESKTOP-0MVMCF2 MINGW64 ~/Desktop/멋쟁이사자처럼9기/studywithLikeLion/Classlion/Model실습/lionproject (master)

$ python manage.py createsuperuser

Username: sss

Email address: ss@likelion.org

Password:

Password (again):

This password is too short. It must contain at least 8 characters.

This password is too common.

This password is entirely numeric.

Bypass password validation and create user anyway? [y/N]: y

Superuser created successfully.

민선아@DESKTOP-0MVMCF2 MINGW64 ~/Desktop/멋쟁이사자처럼9기/studywithLikeLion/Classlion/Model실습/lionproject (master)

$ python manage.py runserver

Watching for file changes with StatReloader

Performing system checks...

System check identified no issues (0 silenced).

June 02, 2021 - 00:02:21

Django version 3.2.3, using settings 'lionproject.settings'

Starting development server at http://127.0.0.1:8000/

Quit the server with CTRL-BREAK.

[02/Jun/2021 00:02:25] "GET / HTTP/1.1" 200 10697

[02/Jun/2021 00:02:34] "GET /admin/ HTTP/1.1" 302 0

[02/Jun/2021 00:02:34] "GET /admin/login/?next=/admin/ HTTP/1.1" 200 2214

[02/Jun/2021 00:02:34] "GET /admin/ HTTP/1.1" 302 0

[02/Jun/2021 00:02:34] "GET /admin/login/?next=/admin/ HTTP/1.1" 200 2214

[02/Jun/2021 00:02:39] "POST /admin/login/?next=/admin/ HTTP/1.1" 302 0

[02/Jun/2021 00:02:39] "GET /admin/ HTTP/1.1" 200 3324

[02/Jun/2021 00:02:40] "GET /static/admin/css/dashboard.css HTTP/1.1" 200 380

[02/Jun/2021 00:02:40] "GET /static/admin/img/icon-addlink.svg HTTP/1.1" 200 331

[02/Jun/2021 00:02:40] "GET /static/admin/img/icon-changelink.svg HTTP/1.1" 200 380

<admin.py>

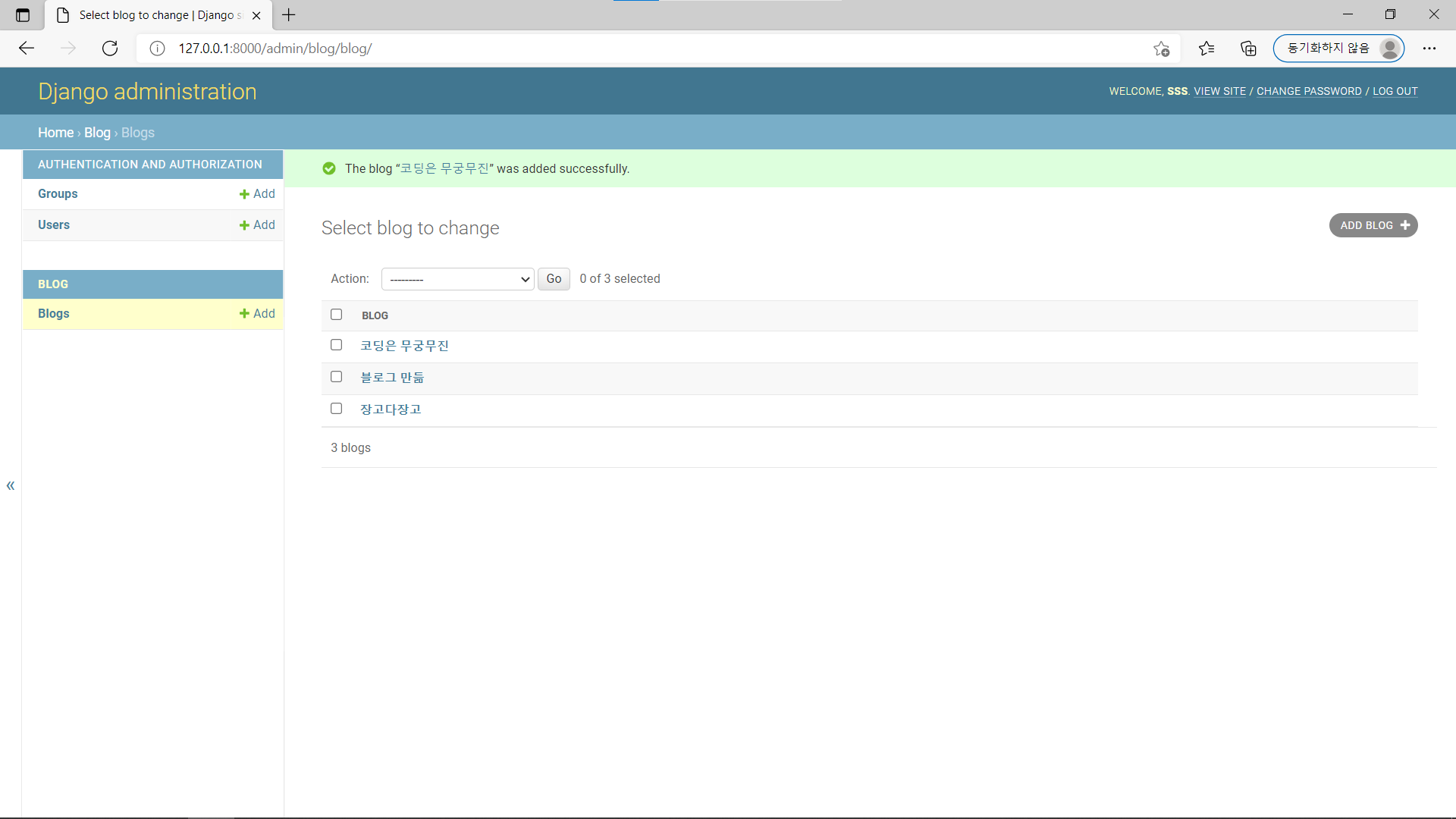
from django.contrib import admin

from .models import Blog

# Register your models here.

admin.site.register(Blog)

* Admin 사이트에 등록해줘야함
* 다시 웹 페이지에서 새로고침



<models.py>

    def \_\_str\_\_(self):

        return self.title

글 제목 보이게 override 하는 것

CRUD = Create, Read, Update, Delete

**1. READ**

from django.shortcuts import render

from .models import Blog

# Create your views here.

def home(request):

    blogs = Blog.objects.all()

    return render(request, 'home.html',{'blogs':blogs})

<urls.py>

from django.contrib import admin

from django.urls import path

from blog.views import \*

urlpatterns = [

    path('admin/', admin.site.urls),

    path('',home,name="home")

]

<home.html>

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>BLOG</title>

</head>

<body>

    <h1>Blog Project</h1>

    {{blogs}}

</body>

</html>

Views.py에 만들어놨던 key값 {{blogs}} 사용

Views.py urls.py home.html templates파일 다 만들었으면

Python manage.py runserver 🡪 runserver 하면 됨!!

<home.html> css로 조금 꾸미는 과정

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>BLOG</title>

    <style>

        body{text-align: center;}

    </style>

</head>

<body>

    <h1>Blog Project</h1>

    <div class="container">

    {% for blog in blogs %}

        <div>

        <h3> {{blog.title}}</h3>

        {{blog.writer}}

        {{blog.body}}

        <br>

        </div>

    {% endfor %}

    </div>

</body>

</html>

<models.py>

본문이 너무 길 때 첫화면에 뜨지 않게 summary하는 함수 기능

    def summary(self):

        return self.body[:100]

100번째 index까지 슬래싱 slashing

**Path Converter: Detail 상세페이지 만드는 법**

Details.html 하나 만들어

<views.py>

from django.shortcuts import render,get\_object\_or\_404

* 찾을 수 없는 페이지 요청했을 때 뜨는 에러 페이지

def detail(request,id):

    blog = get\_object\_or\_404(Blog, pk = id)    # Blog.objects.get(id = id)

    return render(request,'detail.html'),{'blog':blog}

Primary Key: 식별자 - Blog라는 class(테이블)에서 각각 row, column을 구별해주는 id값을 pk라고 한다.

<urls.py> path 추가

urlpatterns = [

    path('admin/', admin.site.urls),

    path('',home,name="home")

    path('<str:id>',detail,name="detail"),

]

<home.html>

위에 id를 어디서 얻어오는거냐면

<body>

    <h1>Blog Project</h1>

    <div class="container">

    {% for blog in blogs %}

        <div>

        <h3> {{blog.title}}</h3>

        {{blog.id}}

        {{blog.writer}}

        {{blog.summary}} <a href ="{%url 'detail' blog.id %}">...more</a>

        <br>

        </div>

    {% endfor %}

    </div>

</body>

</html>

Summary 옆에다가 a 태그 걸어서

Views.py, urls.py, templates – detail.html 완료

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Detail</title>

    <style>

        body{text-align: center;}

    </style>

</head>

<body>

    <h1>{{blog.title</h1>

    <div>

    {{blog.writer}}

    {{blog.pub\_date}}

    </div>

    <hr>

    <p>{{blog.body}}</p>

</body>

</html>

Runserver 돌리면 됨

**2. CREATE**

NEW: new.html 보여줌 - create하기 위한 정보를 받는 기능

Create: 데이터베이스에 저장

templates안에 New.html 파일 하나 만들어

<views.py>

def new(request):

    return render(request,'new.html')

함수 하나 만들고

<urls.py>

함수 새로 만들 때마다 path 하나 추가

    path('new/',new,name="new"),

<home.html>

버튼 같은 거 만들어야지

<body>

    <h1>Blog Project</h1>

    <div>

        <a href="{%url 'new'%}">write blog</a>

    </div>

    <div class="container">

    {% for blog in blogs%}

<new.html>

Form post + tab

http 요청 방식

GET vs POST

* Get method: data를 얻기 위한 요청, 데이터가 url에 보임
  + 보안에 굉장히 안좋음 ex. Word Count
  + 이거 안씀!! 무조건 Post 방식
* Post method: 데이터 생성하기 위한 요청, 데이터 url 안보임, Csrf 공격 방지

🡪 csrf 공격: 웹사이트 취약점 공격, 사이트 간 요청 위조, 공격자가 수정 또는 삭제함

<h1>Write Your Own Blog</h1>

<form action="" method="post">

    {%csrf\_token%}

    <p>제목:<input type="text" name="title"></p>

    <p>작성자:<input type="text" name="writer"></p>

    본문:<textarea name="" id="" cols="30" rows="10"></textarea>

    <button type="submit"> submit</button>

</form>

Post method 사용시

{%csrf\_token%}

라는 template 변수 꼭 써줘야 함.

<views.py>

Create 하기 위해서 views.py로 가야함

from django.utils import timezone

* Pub\_date처럼 작성한 시간을 저장 - django에서 제공하는module 사용

def create(request):

    new\_blog = Blog()

    new\_blog.title = request.POST['title']

    new\_blog.writer = request.POST['writer']

    new\_blog.body = request.POST['body']

    new\_blog.pub\_date = timezone.now()

    new\_blog.save()

    return redirect('detail',new\_blog.id)

New.html에서 정보를 받는 것 request.POST

* 객체지향적으로 coding 하는 것

<urls.py>

Path 추가

path('create/',create,name="create"),

<new.html>

<form action="{%url 'create'%}" method="post">

<TERMINAL>

민선아@DESKTOP-0MVMCF2 MINGW64 ~/Desktop/멋쟁이사자처럼9기/studywithLikeLion/Classlion/Model실습/lionproject (master)

$ python manage.py runserver

Watching for file changes with StatReloader

Performing system checks...

System check identified no issues (0 silenced).

June 02, 2021 - 15:03:34

Django version 3.2.3, using settings 'lionproject.settings'

Starting development server at http://127.0.0.1:8000/

Quit the server with CTRL-BREAK.

[02/Jun/2021 15:03:36] "GET / HTTP/1.1" 200 1378

[02/Jun/2021 15:03:40] "GET /new/ HTTP/1.1" 200 422

[02/Jun/2021 15:04:06] "POST /create/ HTTP/1.1" 302 0

[02/Jun/2021 15:04:06] "GET /5 HTTP/1.1" 200 551

[02/Jun/2021 15:04:16] "GET / HTTP/1.1" 200 1625

(myvenv)

**3. UPDATE 수정하기**

Edit: edit.html 보여줌

Update: 데이터베이스에 적용

\*수정할 데이터의 id값을 받아야 함 – path converter 사용

<views.py>

def edit(request, id):

    edit\_blog = Blog.objects.get(id = id)

    return render(request, 'edit.html',{'blog':edit\_blog})

<urls.py>

path('edit/<str:id>',edit,name="edit")

<str:id> id값을 받아올 path converter 만들어야 함

<detail.html>

<body>

    <h1>{{blog.title}}</h1>

    <div>

    작성자: {{blog.writer}}

    <br>

    날짜: {{blog.pub\_date}}

    </div>

    <hr>

    <p>{{blog.body}}</p>

    <a href="{% url 'edit' blog.id %}">수정하기</a>

</body>

</html>

<views.py>

def update(request,id):

    update\_blog = Blog.objects.get(id = id)

    update\_blog.title = request.POST['title']

    update\_blog.writer = request.POST['writer']

    update\_blog.body = request.POST['body']

    update\_blog.pub\_date = timezone.now()

    update\_blog.save()

    return redirect('detail',update\_blog.id)

<edit.html>

New.html이랑 비슷하게 만들기

<h1>Update Your Blog</h1>

<form action="" method="post">

    {%csrf\_token%}

    <p>제목: <input type="text" name="title" value ="{{blog.title}}"></p>

    <p>작성자: <input type="text" name="writer" value="{{blog.writer}}"></p>

    본문: <textarea name="body" id="" cols="30" rows="10"{{blog.body}}></textarea>

    <button type="submit"> submit</button>

</form>

<form action="{%url 'update' blog.id %}" method="post">

매개변수 blog.id 설정

**4. DELETE 삭제**

<views.py>

def delete(request,id):

    delete\_blog = Blog.objects.get(id = id)

    delete\_blog.delete()

    return redirect('home')

<urls.py>

    path('delete/<str:id>',delete,name="delete"),

<details.html>

    <a href="{% url 'delete' blog.id %}">삭제하기</a>

그 다음에 runserver