Sriram Sunderrajan

220962444

WP Lab

Week-9

20th March 2025

**PROJECT FILES**

**settings.py**

from pathlib import Path

import os

# Build paths inside the project like this: BASE\_DIR / 'subdir'.

BASE\_DIR = Path(\_\_file\_\_).resolve().parent.parent

# Quick-start development settings - unsuitable for production

# See https://docs.djangoproject.com/en/5.1/howto/deployment/checklist/

# SECURITY WARNING: keep the secret key used in production secret!

SECRET\_KEY = 'django-insecure-bijf50)cez%324m01ld+an+-q9$dfd^zkn9w6kqnr940l4s1^#'

# SECURITY WARNING: don't run with debug turned on in production!

DEBUG = True

ALLOWED\_HOSTS = []

# Application definition

INSTALLED\_APPS = [

'django.contrib.admin',

'django.contrib.auth',

'django.contrib.contenttypes',

'django.contrib.sessions',

'django.contrib.messages',

'django.contrib.staticfiles',

'blog',

'webdirApp',

'workApp'

]

MIDDLEWARE = [

'django.middleware.security.SecurityMiddleware',

'django.contrib.sessions.middleware.SessionMiddleware',

'django.middleware.common.CommonMiddleware',

'django.middleware.csrf.CsrfViewMiddleware',

'django.contrib.auth.middleware.AuthenticationMiddleware',

'django.contrib.messages.middleware.MessageMiddleware',

'django.middleware.clickjacking.XFrameOptionsMiddleware',

]

ROOT\_URLCONF = 'myProj.urls'

TEMPLATES = [

{

'BACKEND': 'django.template.backends.django.DjangoTemplates',

'DIRS': [

os.path.join(BASE\_DIR, 'blog/templates/blog'),

os.path.join(BASE\_DIR, 'webdirApp/templates/webdirApp'),

os.path.join(BASE\_DIR, 'workApp/templates/workApp')

],

'APP\_DIRS': True,

'OPTIONS': {

'context\_processors': [

'django.template.context\_processors.debug',

'django.template.context\_processors.request',

'django.contrib.auth.context\_processors.auth',

'django.contrib.messages.context\_processors.messages',

],

},

},

]

WSGI\_APPLICATION = 'myProj.wsgi.application'

# Database

# https://docs.djangoproject.com/en/5.1/ref/settings/#databases

DATABASES = {

'default': {

'ENGINE': 'django.db.backends.sqlite3',

'NAME': BASE\_DIR / 'db.sqlite3',

}

}

# Password validation

# https://docs.djangoproject.com/en/5.1/ref/settings/#auth-password-validators

AUTH\_PASSWORD\_VALIDATORS = [

{

'NAME': 'django.contrib.auth.password\_validation.UserAttributeSimilarityValidator',

},

{

'NAME': 'django.contrib.auth.password\_validation.MinimumLengthValidator',

},

{

'NAME': 'django.contrib.auth.password\_validation.CommonPasswordValidator',

},

{

'NAME': 'django.contrib.auth.password\_validation.NumericPasswordValidator',

},

]

# Internationalization

# https://docs.djangoproject.com/en/5.1/topics/i18n/

LANGUAGE\_CODE = 'en-us'

TIME\_ZONE = 'UTC'

USE\_I18N = True

USE\_TZ = True

# Static files (CSS, JavaScript, Images)

# https://docs.djangoproject.com/en/5.1/howto/static-files/

STATIC\_URL = 'static/'

# Default primary key field type

# https://docs.djangoproject.com/en/5.1/ref/settings/#default-auto-field

DEFAULT\_AUTO\_FIELD = 'django.db.models.BigAutoField'

**urls.py**

from django.contrib import admin

from django.urls import path, include

urlpatterns = [

path('admin/', admin.site.urls),

path('webdirApp/', include('webdirApp.urls')),

path('workApp/', include('workApp.urls')),

]

**Question – 1**

**urls.py**

from django.urls import path

from . import views

urlpatterns = [

path('', views.directory\_view, name='webdirApp'),

path('add\_category/', views.add\_category, name='add\_category'),

path('add\_page/', views.add\_page, name='add\_page'),

]

**models.py**

from django.db import models

class Category(models.Model):

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

number\_of\_visits = models.PositiveIntegerField(default=0)

number\_of\_likes = models.PositiveIntegerField(default=0)

def \_\_str\_\_(self):

return self.name

class Page(models.Model):

category = models.ForeignKey(Category, related\_name='pages', on\_delete=models.CASCADE)

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

url = models.URLField()

views = models.PositiveIntegerField(default=0)

def \_\_str\_\_(self):

return self.title

**forms.py**

from django import forms

from .models import Category, Page

class CategoryForm(forms.ModelForm):

class Meta:

model = Category

fields = ['name', 'number\_of\_visits', 'number\_of\_likes']

class PageForm(forms.ModelForm):

class Meta:

model = Page

fields = ['category', 'title', 'url', 'views']

**views.py**

from django.shortcuts import render, redirect

from .models import Category, Page

from .forms import CategoryForm, PageForm

# View to display the directory

def directory\_view(request):

categories = Category.objects.all()

return render(request, 'webdirApp/directory.html', {'categories': categories})

# View to display and submit the Category form

def add\_category(request):

if request.method == 'POST':

form = CategoryForm(request.POST)

if form.is\_valid():

form.save()

return redirect('webdirApp')

else:

form = CategoryForm()

return render(request, 'webdirApp/add\_category.html', {'form': form})

# View to display and submit the Page form

def add\_page(request):

if request.method == 'POST':

form = PageForm(request.POST)

if form.is\_valid():

form.save()

return redirect('webdirApp')

else:

form = PageForm()

return render(request, 'webdirApp/add\_page.html', {'form': form})

**tempates: directory.html**

<!DOCTYPE html>

<html>

<head>

<title>Website Directory</title>

</head>

<body>

<h1>Website Directory</h1>

<a href="{% url 'add\_category' %}">Add Category</a><br>

<a href="{% url 'add\_page' %}">Add Page</a><br><br>

{% for category in categories %}

<h2>{{ category.name }}</h2>

<p>Visits: {{ category.number\_of\_visits }}</p>

<p>Likes: {{ category.number\_of\_likes }}</p>

<ul>

{% for page in category.pages.all %}

<li><a href="{{ page.url }}" target="\_blank">{{ page.title }}</a> - Views: {{ page.views }}</li>

{% endfor %}

</ul>

{% endfor %}

</body>

</html>

**templates: add\_category.html**

<!DOCTYPE html>

<html>

<head>

<title>Add Category</title>

</head>

<body>

<h1>Add Category</h1>

<form method="POST">

{% csrf\_token %}

{{ form.as\_p }}

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

</form>

<a href="{% url 'webdirApp' %}">Back to Directory</a>

</body>

</html>

**templates: add\_page.html**

<!DOCTYPE html>

<html>

<head>

<title>Add Page</title>

</head>

<body>

<h1>Add Page</h1>

<form method="POST">

{% csrf\_token %}

{{ form.as\_p }}

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

</form>

<a href="{% url 'webdirApp' %}">Back to Directory</a>

</body>

</html>

**OUTPUT**







G

**Question - 2**

**urls.py**

from django.urls import path

from . import views

urlpatterns = [

path('', views.insert\_works, name='insert\_works'),

path('retrieve', views.retrieve\_people, name='retrieve\_people'),

]

**models.py**

from django.db import models

class Works(models.Model):

person\_name = models.CharField(max\_length=100)

company\_name = models.CharField(max\_length=100)

salary = models.DecimalField(max\_digits=10, decimal\_places=2)

def \_\_str\_\_(self):

return f"{self.person\_name} works at {self.company\_name}"

class Lives(models.Model):

person\_name = models.CharField(max\_length=100)

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

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

def \_\_str\_\_(self):

return f"{self.person\_name} lives at {self.street}, {self.city}"

**forms.py**

from django import forms

from .models import Works, Lives

# Form for Works and Lives data

class WorksAndLivesForm(forms.Form):

person\_name = forms.CharField(max\_length=100)

company\_name = forms.CharField(max\_length=100)

salary = forms.DecimalField(max\_digits=10, decimal\_places=2)

street = forms.CharField(max\_length=200)

city = forms.CharField(max\_length=100)

class CompanyForm(forms.Form):

company\_name = forms.CharField(max\_length=100)

**views.py**

from django.shortcuts import render, redirect

from .forms import WorksAndLivesForm, CompanyForm

from .models import Works, Lives

# View to insert data into the WORKS and LIVES tables

def insert\_works(request):

if request.method == 'POST':

form = WorksAndLivesForm(request.POST)

if form.is\_valid():

# Create and save the Works record

person\_name = form.cleaned\_data['person\_name']

company\_name = form.cleaned\_data['company\_name']

salary = form.cleaned\_data['salary']

works\_instance = Works.objects.create(

person\_name=person\_name,

company\_name=company\_name,

salary=salary

)

# Create and save the Lives record

street = form.cleaned\_data['street']

city = form.cleaned\_data['city']

Lives.objects.create(

person\_name=person\_name,

street=street,

city=city

)

return redirect('retrieve\_people')

else:

form = WorksAndLivesForm()

return render(request, 'workApp/insert\_works.html', {'form': form})

# View to retrieve names of people working for a specific company

def retrieve\_people(request):

people = None

if request.method == 'POST':

form = CompanyForm(request.POST)

if form.is\_valid():

company\_name = form.cleaned\_data['company\_name']

people\_in\_company = Works.objects.filter(company\_name=company\_name)

people = []

#people\_city\_pairs = [[(work.name, life.city) for life in Lives.objects.get(person\_name=work.person\_name)] for work in people\_in\_company]

#people\_in\_company = [(work.person\_name, Lives.objects.get(person\_name=work.person\_name).city)

# for work in people\_in\_company]

for work in people\_in\_company:

for life in Lives.objects.filter(person\_name=work.person\_name):

people.append((work.person\_name, life.city))

else:

form = CompanyForm()

return render(request, 'workApp/retrieve\_people.html', {'form': form, 'people\_in\_company': people})

**templates: insert\_works.html**

<!DOCTYPE html>

<html>

<head>

<title>Insert Data into WORKS</title>

</head>

<body>

<h1>Insert Data into WORKS Table</h1>

<form method="POST">

{% csrf\_token %}

{{ form.as\_p }}

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

</form>

<a href="{% url 'retrieve\_people' %}">Go to Retrieve People</a>

</body>

</html>

**templates: retrieve\_people.html**

<!DOCTYPE html>

<html>

<head>

<title>Retrieve People Working for a Company</title>

</head>

<body>

<h1>Retrieve People Working for a Specific Company</h1>

<form method="POST">

{% csrf\_token %}

{{ form.as\_p }}

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

</form>

{% if people\_in\_company %}

<h2>People working for the company:</h2>

<ul>

{% for person, city in people\_in\_company %}

<li>{{ person }} - City: {{ city }}</li>

{% endfor %}

</ul>

{% endif %}

<a href="{% url 'insert\_works' %}">Go to Insert Works</a>

</body>

</html>

**OUTPUT**



