|  |
| --- |
|  |
|  |  |
|  | 2) models.py |
|  |  |
|  | from django.db import models |
|  |  |
|  | class Author(models.Model): |
|  | name = models.CharField(max\_length=100) |
|  | book = models.CharField(max\_length=100) |
|  | bookid = models.IntegerField() |
|  |  |
|  | def \_\_str\_\_(self): |
|  | return self.name |
|  |  |
|  | forms.py |
|  |  |
|  | from django import forms |
|  | from .models import Author |
|  |  |
|  | class AuthorForm(forms.ModelForm): |
|  | class Meta: |
|  | model = Author |
|  | fields = ['name', 'book', 'bookid'] |
|  |  |
|  | settings.py |
|  | INSTALLED\_APPS = [ |
|  | ... |
|  | 'myapp', |
|  | ... |
|  | ] |
|  |  |
|  | Terminal |
|  | python manage.py makemigrations myapp |
|  | python manage.py migrate |
|  |  |
|  | views.py |
|  |  |
|  | from django.shortcuts import render, redirect |
|  | from .forms import AuthorForm |
|  |  |
|  | def create\_author(request): |
|  | if request.method == 'POST': |
|  | form = AuthorForm(request.POST) |
|  | if form.is\_valid(): |
|  | form.save() |
|  | return redirect('author-list') |
|  | else: |
|  | form = AuthorForm() |
|  | return render(request, 'create\_author.html', {'form': form}) |
|  |  |
|  | 3)Terminal |
|  |  |
|  | python manage.py startapp myapp |
|  |  |
|  | view.py |
|  |  |
|  | from django.shortcuts import render |
|  |  |
|  | def get\_name(request): |
|  | return render(request, 'get\_name.html') |
|  |  |
|  | def post\_name(request): |
|  | if request.method == 'POST': |
|  | first\_name = request.POST.get('first\_name') |
|  | last\_name = request.POST.get('last\_name') |
|  | return render(request, 'post\_name.html', {'first\_name': first\_name, 'last\_name': last\_name}) |
|  | else: |
|  | return render(request, 'get\_name.html') |
|  |  |
|  | get\_name.html |
|  |  |
|  | <form method="GET" action="{% url 'post-name' %}"> |
|  | {% csrf\_token %} |
|  | <label for="first\_name">First name:</label> |
|  | <input type="text" id="first\_name" name="first\_name" required> |
|  | <br> |
|  | <label for="last\_name">Last name:</label> |
|  | <input type="text" id="last\_name" name="last\_name" required> |
|  | <br> |
|  | <input type="submit" value="Submit"> |
|  | </form> |
|  |  |
|  | post\_name.py |
|  | <p>First name: {{ first\_name }}</p> |
|  | <p>Last name: {{ last\_name }}</p> |
|  |  |
|  | urls.py/app |
|  | from django.urls import path |
|  | from . import views |
|  |  |
|  | urlpatterns = [ |
|  | path('', views.get\_name, name='get-name'), |
|  | path('post/', views.post\_name, name='post-name'), |
|  | ] |
|  |  |
|  | settings.py |
|  | INSTALLED\_APPS = [ |
|  | ... |
|  | 'myapp', |
|  | ... |
|  | ] |
|  |  |
|  | 4)myapp/forms.py |
|  | from django import forms |
|  | from django.contrib.auth.forms import AuthenticationForm |
|  |  |
|  | class LoginForm(AuthenticationForm): |
|  | def \_\_init\_\_(self, \*args, \*\*kwargs): |
|  | super().\_\_init\_\_(\*args, \*\*kwargs) |
|  | self.fields['username'].widget.attrs.update({'class': 'form-control'}) |
|  | self.fields['password'].widget.attrs.update({'class': 'form-control'}) |
|  | views.py |
|  | from django.contrib.auth.views import LoginView |
|  | from .forms import LoginForm |
|  |  |
|  | class MyLoginView(LoginView): |
|  | form\_class = LoginForm |
|  | template\_name = 'login.html' |
|  |  |
|  | templates/login.html |
|  | {% extends 'base.html' %} |
|  |  |
|  | {% block content %} |
|  | <h2>Login</h2> |
|  | <form method="post"> |
|  | {% csrf\_token %} |
|  | {{ form.as\_p }} |
|  | <button type="submit">Login</button> |
|  | </form> |
|  | {% endblock %} |
|  |  |
|  | urls.py |
|  | from django.urls import path |
|  | from .views import MyLoginView |
|  |  |
|  | urlpatterns = [ |
|  | path('login/', MyLoginView.as\_view(), name='login'), |
|  | ] |
|  |  |
|  | settings.py |
|  | INSTALLED\_APPS = [ |
|  | ... |
|  | 'myapp', |
|  | ... |
|  | ] |
|  |  |
|  | 4) forms.py |
|  | from django import forms |
|  | from .models import sampleModel |
|  |  |
|  | class SampleModelForm(forms.ModelForm): |
|  | class Meta: |
|  | model = sampleModel |
|  | fields = ['title', 'description'] |
|  |  |
|  | admin.py |
|  | from django.contrib import admin |
|  | from .models import sampleModel |
|  | from .forms import SampleModelForm |
|  |  |
|  | class SampleModelAdmin(admin.ModelAdmin): |
|  | form = SampleModelForm |
|  |  |
|  | admin.site.register(sampleModel, SampleModelAdmin) |
|  |  |
|  | shell |
|  | from myapp.models import sampleModel |
|  |  |
|  | sample = sampleModel(title='My Sample Title', description='My Sample Description') |
|  | sample.save() |
|  |  |
|  | sample = sampleModel.objects.get(title='My Sample Title') |
|  | sample.description = 'Updated Sample Description' |
|  | sample.save() |
|  |  |
|  | sample = sampleModel.objects.get(title='My Sample Title') |
|  | sample.delete() |
|  |  |
|  | 2)models.py |
|  | from django.db import models |
|  |  |
|  | class TextFile(models.Model): |
|  | file = models.FileField(upload\_to='text\_files/') |
|  |  |
|  | views.py |
|  |  |
|  | from django.shortcuts import render, redirect |
|  | from .forms import TextFileForm |
|  |  |
|  | def upload\_file(request): |
|  | if request.method == 'POST': |
|  | form = TextFileForm(request.POST, request.FILES) |
|  | if form.is\_valid(): |
|  | form.save() |
|  | return redirect('admin:index') |
|  | else: |
|  | form = TextFileForm() |
|  | return render(request, 'upload\_file.html', {'form': form}) |
|  |  |
|  | template |
|  | <form method="post" enctype="multipart/form-data"> |
|  | {% csrf\_token %} |
|  | {{ form.as\_p }} |
|  | <button type="submit">Upload</button> |
|  | </form> |
|  |  |
|  | from django.contrib import admin |
|  | from .models import TextFile |
|  |  |
|  | admin.site.register(TextFile) |
|  |  |
|  | 3)pip install django-dynamic-csv |
|  |  |
|  | views.py |
|  | from django.shortcuts import render |
|  | from dynamic\_rawid.admin import DynamicRawIDMixin |
|  | from dynamic\_rawid.widgets import ManyToManyRawIdWidget |
|  | from dynamic\_rawid.fields import DynamicRawIDField |
|  | from django.views.generic import ListView |
|  | from dynamic\_csv.views import CSVView |
|  | from .models import Employee |
|  |  |
|  | class EmployeeListView(DynamicRawIDMixin, ListView): |
|  | model = Employee |
|  |  |
|  | class EmployeeCSVView(CSVView): |
|  | model = Employee |
|  | filename = 'employees.csv' |
|  | header = ['Ename', 'EID'] |
|  | delimiter = ',' |
|  |  |
|  | urls.py |
|  | from django.urls import path |
|  | from .views import EmployeeListView, EmployeeCSVView |
|  |  |
|  | urlpatterns = [ |
|  | path('employees/', EmployeeListView.as\_view(), name='employee\_list'), |
|  | path('employees/csv/', EmployeeCSVView.as\_view(), name='employee\_csv'), |
|  | ] |
|  |  |
|  | html |
|  | {% extends 'base.html' %} |
|  |  |
|  | {% block content %} |
|  | <h1>Employee List</h1> |
|  | <table> |
|  | <thead> |
|  | <tr> |
|  | <th>Name</th> |
|  | <th>ID</th> |
|  | </tr> |
|  | </thead> |
|  | <tbody> |
|  | {% for employee in object\_list %} |
|  | <tr> |
|  | <td>{{ employee.Ename }}</td> |
|  | <td>{{ employee.EID }}</td> |
|  | </tr> |
|  | {% endfor %} |
|  | </tbody> |
|  | </table> |
|  | <a href="{% url 'employee\_csv' %}">Export CSV</a> |
|  | {% endblock %} |
|  |  |
|  | 4)from reportlab.pdfgen import canvas |
|  |  |
|  | # Get string message from user |
|  | message = input("Enter a message: ") |
|  |  |
|  | # Create PDF file |
|  | pdf\_file = canvas.Canvas("message.pdf") |
|  |  |
|  | # Set font and font size |
|  | pdf\_file.setFont("Helvetica", 12) |
|  |  |
|  | # Write message into PDF file |
|  | pdf\_file.drawString(100, 750, message) |
|  |  |
|  | # Save and close the PDF file |
|  | pdf\_file.save() |
|  |  |
|  | print("Message saved in message.pdf") |