Chương 1: Giới thiệu & Cài đặt Django

Chương 1: Giới thiệu & Cài đặt Django

Django là một web framework mã nguồn mở được viết bằng Python, dùng để xây dựng các ứng dụng web nhanh chóng, bảo mật và có thể mở rộng.

1. Vì sao chọn Django?

- Được sử dụng bởi các công ty lớn như Instagram, Mozilla, Pinterest...
- Hỗ trợ sẵn các tính năng: ORM, quản trị, bảo mật, phân quyền, xử lý form...
- Tốc độ phát triển nhanh nhờ cơ chế “DRY – Don’t Repeat Yourself”

2. Cài đặt môi trường Django

Yêu cầu: Đã cài sẵn Python >= 3.8

Bước 1: Tạo môi trường ảo (virtual environment)

```
$ python -m venv venv
$ source venv/bin/activate  (Linux/Mac)
$ venv\Scripts\activate   (Windows)
```

Bước 2: Cài đặt Django

```
(venv)$ pip install django
```

Bước 3: Kiểm tra cài đặt

```
(venv)$ django-admin --version
```

3. Tạo project đầu tiên

```
(venv)$ django-admin startproject mysite
```

Cấu trúc thư mục:

```
mysite/
├── manage.py
├── mysite/
│   ├── __init__.py
```

```
│   ├── settings.py
│   ├── urls.py
│   ├── asgi.py
│   └── wsgi.py
```

4. Chạy thử project

```
(venv)$ cd mysite
(venv)$ python manage.py runserver
```

Truy cập trình duyệt: http://127.0.0.1:8000/

Chương 2: Cấu trúc Project & App

Khi bạn tạo một project Django bằng lệnh:

$ django-admin startproject myproject

Sẽ tạo ra cấu trúc như sau:

```
myproject/
├── manage.py
├── myproject/
│   ├── __init__.py
│   ├── asgi.py
│   ├── settings.py
│   ├── urls.py
│   └── wsgi.py
```

Giải thích:

- manage.py: file dòng lệnh quản lý project
- settings.py: cấu hình toàn bộ project
- urls.py: định nghĩa các đường dẫn chính
- wsgi.py/asgi.py: giao tiếp với server thật

Tạo 1 ứng dụng (app) riêng:

$ python manage.py startapp myapp

Cấu trúc một app:

```
myapp/
├── admin.py
├── apps.py
├── models.py
├── views.py
├── tests.py
```

Chương 2: Cấu trúc Project & App

├── migrations/

├── templates/myapp/

└── static/myapp/

Khai báo app trong settings.py:

INSTALLED_APPS += ['myapp']

Chương 3: URL Dispatcher và View

URL Dispatcher điều hướng yêu cầu từ người dùng đến các view tương ứng.

myproject/urls.py:

```
from django.urls import path, include
urlpatterns = [
    path('', include('myapp.urls')),
]
```

myapp/urls.py:

```
from django.urls import path
from . import views
urlpatterns = [
    path('', views.trang_chu, name='trang_chu'),
]
```

myapp/views.py:

```
from django.http import HttpResponse

def trang_chu(request):
    return HttpResponse("Chào mừng đến với Django!")
```

Chương 4: Templates (HTML + Template Tags)

Templates chứa HTML và biến động.

Tạo file: myapp/templates/myapp/trangchu.html

Nội dung:

```
<h1>Chào {{ ten_nguoi_dung }}</h1>
```

views.py:

```
from django.shortcuts import render

def trang_chu(request):
    return render(request, 'myapp/trangchu.html', {'ten_nguoi_dung': 'Trúc'})
```

Các tag phổ biến:

- {{ ten }}: Biến
- {% if %}, {% for %}: Điều kiện, vòng lặp
- {% include %}, {% url %}: Import file, gọi URL

Chương 5: Models và ORM

models.py:

```
from django.db import models


class BaiViet(models.Model):
    tieu_de = models.CharField(max_length=200)
    noi_dung = models.TextField()
    ngay_dang = models.DateField(auto_now_add=True)
```

Tạo bảng:

```
$ python manage.py makemigrations
$ python manage.py migrate
```

Thao tác dữ liệu:

```
bv = BaiViet(tieu_de='A', noi_dung='...')
bv.save()
BaiViet.objects.all()
```

Chương 6: Django Admin

Tạo superuser:

$ python manage.py createsuperuser

Khai báo model:

admin.py:

```
from django.contrib import admin
from .models import BaiViet

admin.site.register(BaiViet)
```

Truy cập:

http://127.0.0.1:8000/admin/

Chương 7: Forms và xử lý dữ liệu người dùng

forms.py:

```
from django import forms


class LienHeForm(forms.Form):
    ho_ten = forms.CharField()
    email = forms.EmailField()
    noi_dung = forms.CharField(widget=forms.Textarea)
```

views.py:

```
def lien_he(request):
    if request.method == 'POST':
        form = LienHeForm(request.POST)
        if form.is_valid():
            # xử lý dữ liệu
            pass
    else:
        form = LienHeForm()
    return render(request, 'form.html', {'form': form})
```

Chương 8: CRUD hoàn chỉnh

URL:

path('baiviet/', views.ds_baiviet),

path('baiviet/them/', views.them_baiviet),

path('baiviet/<int:id>/sua/', views.sua_baiviet),

path('baiviet/<int:id>/xoa/', views.xoa_baiviet),

Chương 9: Tạo Blog mini

Tạo blog gồm:

- Danh sách bài viết
- Xem chi tiết
- Thêm mới
- Sửa, xoá

Sử dụng model BaiViet, views, template và admin.

Chương 10: REST API với Django Rest Framework

Cài đặt:

```
$ pip install djangorestframework
```

Cấu hình:

```
INSTALLED_APPS += ['rest_framework']
```

serializer.py:

```
from rest_framework import serializers
from .models import BaiViet

class BaiVietSerializer(serializers.ModelSerializer):
    class Meta:
        model = BaiViet
        fields = '__all__'
```

views.py:

```
from rest_framework import viewsets

class BaiVietViewSet(viewsets.ModelViewSet):
    queryset = BaiViet.objects.all()
    serializer_class = BaiVietSerializer
```

urls.py:

```
from rest_framework.routers import DefaultRouter

router = DefaultRouter()
router.register(r'baiviet', BaiVietViewSet)
urlpatterns += router.urls
```