

Django Tham Khảo Nhanh

Model, view, template, ORM, form, admin, xác thực

Thiết Lập Dự Án

Tạo Dự Án & App

```
pip install django
django-admin startproject mysite
cd mysite
python manage.py startapp blog
```

Các Lệnh Thường Dùng

runserver	Khởi động dev server trên cổng 8000
makemigrations	Tạo file migration từ thay đổi model
migrate	Áp dụng migration vào database
createsuperuser	Tạo superuser cho admin
shell	Shell Python tương tác với Django
test	Chạy bộ test

Cấu Trúc Dự Án

manage.py	Điểm vào CLI
settings.py	Cấu hình dự án
urls.py	Cấu hình URL gốc
wsgi.py / asgi.py	Điểm vào server
apps/models.py	Database model
apps/views.py	Xử lý request

Models

Định Nghĩa Model

```
from django.db import models

class Post(models.Model):
    title = models.CharField(max_length=200)
    body = models.TextField()
    created = models.DateTimeField(auto_now_add=True)
    published = models.BooleanField(default=False)
```

Các Loại Field

CharField(max_length=N)	Văn bản ngắn (bắt buộc max_length)
TextField()	Văn bản dài (không giới hạn)
IntegerField()	Giá trị số nguyên
FloatField()	Số thực
BooleanField()	True / False
DateTimeField()	Ngày và giờ
EmailField()	Email có xác thực
FileField(upload_to='')	Upload file

Quan Hệ

```
author = models.ForeignKey(
    User, on_delete=models.CASCADE
)
tags = models.ManyToManyField(Tag, blank=True)
profile = models.OneToOneField(User, on_delete=models.CASCADE)
```

Meta & Methods

```
class Meta:
    ordering = ['-created']
    verbose_name_plural = 'posts'

def __str__(self):
    return self.title
```

Views

Function-Based View

```
from django.shortcuts import render, get_object_or_404

def post_list(request):
    posts = Post.objects.filter(published=True)
    return render(request, 'blog/list.html', {'posts': posts})
```

Class-Based Views

```
from django.views.generic import ListView, DetailView

class PostListView(ListView):
    model = Post
    template_name = 'blog/list.html'
    context_object_name = 'posts'
    paginate_by = 10
```

CBV Thường Dùng

ListView	Hiển thị danh sách object
DetailView	Hiển thị một object
CreateView	Form tạo object
UpdateView	Form chỉnh sửa object
DeleteView	Xác nhận và xóa object
TemplateView	Render template (không có model)

JSON Response

```
from django.http import JsonResponse

def api_posts(request):
    data = list(Post.objects.values('id', 'title'))
    return JsonResponse(data, safe=False)
```

Templates

Cú Pháp Template

```
{{ variable }}
{{ post.title|truncatewords:30 }}
{% if user.is_authenticated %}
    <p>Xin chào, {{ user.username }}!</p>
{% endif %}
```

Vòng Lặp & Điều Kiện

```
{% for post in posts %}
    <h2>{{ post.title }}</h2>
    {% if forloop.last %}<hr>{% endif %}
{% empty %}
    <p>Chưa có bài viết nào.</p>
{% endfor %}
```

Kế Thừa Template

```
{# base.html #}
<html>
<body>{% block content %}{% endblock %}</body>
</html>

{# child.html #}
{% extends "base.html" %}
{% block content %}<h1>Xin chào</h1>{% endblock %}
```

Filter Thường Dùng

 date:"Y-m-d"	Định dạng ngày
 default:"N/A"	Dự phòng khi giá trị rỗng
 length	Đếm phần tử trong list
 truncatewords:N	Giới hạn N từ
 safe	Đánh dấu HTML an toàn (không escape)
 slugify	Chuỗi URL-safe chữ thường

URLs

URL Patterns

```
from django.urls import path, include

urlpatterns = [
    path('', views.index, name='index'),
    path('post/<int:pk>', views.detail, name='detail'),
    path('blog/', include('blog.urls')),
]
```

Path Converters

<int:pk>	Số nguyên (vd: 42)
<str:slug>	Chuỗi không có dấu gạch chéo
<slug:slug>	Slug (chữ, số, gạch ngang)
<uuid:id>	Định dạng UUID
<path:rest>	Đường dẫn đầy đủ kể cả dấu gạch chéo

Reverse URLs

```
from django.urls import reverse
url = reverse('detail', kwargs={'pk': 1})
# Trong template: {% url 'detail' pk=post.pk %}
```

Forms

Model Form

```
from django import forms

class PostForm(forms.ModelForm):
    class Meta:
        model = Post
        fields = ['title', 'body', 'published']
```

Xử Lý Form trong View

```
def create_post(request):
    form = PostForm(request.POST or None)
    if form.is_valid():
        post = form.save(commit=False)
        post.author = request.user
        post.save()
        return redirect('detail', pk=post.pk)
    return render(request, 'blog/form.html', {'form': form})
```

Form trong Template

```
<form method="post">
    {% csrf_token %}
    {{ form.as_p }}
    <button type="submit">Luu</button>
</form>
```

Validation

```
def clean_title(self):
    title = self.cleaned_data['title']
    if len(title) < 5:
        raise forms.ValidationError("Tiêu đề quá ngắn.")
    return title
```

Admin

Đăng Ký Model

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

@admin.register(Post)
class PostAdmin(admin.ModelAdmin):
    list_display = ['title', 'author', 'created', 'published']
    list_filter = ['published', 'created']
    search_fields = ['title', 'body']
```

Django Tham Khảo Nhanh

Tùy Chọn Admin

list_display	Cột trong list view
list_filter	Tùy chọn lọc thanh bên
search_fields	Trường có thể tìm kiếm
prepopulated_fields	Tự điền (vd: slug từ title)
readonly_fields	Không chỉnh sửa được trong admin
ordering	Thứ tự sắp xếp mặc định

ORM Queries

Query Cơ Bản

<code>Post.objects.all()</code>	# tất cả bản ghi
<code>Post.objects.get(pk=1)</code>	# một bản ghi (lỗi nếu không tìm thấy)
<code>Post.objects.filter(published=True)</code>	# queryset
<code>Post.objects.exclude(draft=True)</code>	# loại trừ kết quả khớp
<code>Post.objects.count()</code>	# tổng số

Field Lookups

field_exact	Khớp chính xác (mặc định)
field_icontains	Chứa, không phân biệt hoa thường
field_gt / __lt	Lớn hơn / nhỏ hơn
field_gte / __lte	Lớn hơn hoặc bằng / nhỏ hơn hoặc bằng
field_in=[1,2,3]	Giá trị nằm trong danh sách
field_isnull=True	Là NULL
field_startswith	Bắt đầu bằng chuỗi
field_range=(a,b)	Trong khoảng a đến b (bao gồm)

Kết Hợp & Tổng Hợp

```
from django.db.models import Q, Count, Avg

Post.objects.filter(
    Q(title_icontains='django') | Q(body_icontains='django')
).order_by('-created')[:10]

Post.objects.aggregate(avg_views=Avg('views'))
```

Tạo, Cập Nhật, Xóa

```
post = Post.objects.create(title='Mới', body='...')
post.title = 'Đã cập nhật'
post.save()
Post.objects.filter(draft=True).update(published=False)
post.delete()
```

Xác Thực

Đăng Nhập / Đăng Xuất

```
from django.contrib.auth import authenticate, login, logout

user = authenticate(request, username='admin', password='pw')
if user is not None:
    login(request, user)
```

Bảo Vệ Views

```
from django.contrib.auth.decorators import login_required

@login_required
def dashboard(request):
    return render(request, 'dashboard.html')
```

Auth URLs

```
# urls.py
path('accounts/', include('django.contrib.auth.urls'))
# Cung cấp: login, logout, password_change, password_reset
```

Auth trong Template

```
{% if user.is_authenticated %}
<p>Xin chào, {{ user.username }}</p>
<a href="{% url 'logout' %}">Đăng xuất</a>
{% else %}
<a href="{% url 'login' %}">Đăng nhập</a>
{% endif %}
```

Settings

Cài Đặt Quan Trọng

DEBUG	True cho dev, False cho production
ALLOWED_HOSTS	Danh sách hostname hợp lệ
SECRET_KEY	Khóa ký mật mã (giữ bí mật)
DATABASES	Engine DB, tên, host, thông tin xác thực
INSTALLED_APPS	Danh sách app đã đăng ký
STATIC_URL	URL prefix cho static file
MEDIA_URL / MEDIA_ROOT	Đường dẫn file do người dùng tải lên

Cấu Hình Database

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql',
        'NAME': 'mydb',
        'USER': 'dbuser',
        'HOST': 'localhost',
        'PORT': '5432',
    }
}
```

Static Files

```
STATIC_URL = '/static/'
STATICFILES_DIRS = [BASE_DIR / 'static']
# Trong template: {% load static %}
# <link href="{% static 'css/style.css' %}">
```