1. Crear entorno de trabajo

>> Crear carpeta del proyecto

```
mkdir primerproyecto
```

```
cd primerproyecto
```

>> Crear y activar entorno virtual

```
python3 -m venv env
```

```
source env/bin/activate
```

Nota: para salir del entorno virtual:

```
deactivate
```

2. Instalación de Django

>> Crear archivo `requirements.txt`

Contenido:

```
Django==5.1.3
```

>> Instalar dependencias

```
pip install -r requirements.txt
```

>> Verificar instalación

```
django-admin --version
```

3. Crear proyecto y aplicación

>> Crear proyecto (IMPORTANTE: el punto al final)

```
django-admin startproject nombre_proyecto .
```

>> Crear una aplicación

```
python manage.py startapp nombre_aplicacion
```

4. Configuración inicial (settings.py)

Ajustar idioma, zona horaria, estáticos y registrar la app:

```
TIME_ZONE = 'Europe/Madrid'
LANGUAGE_CODE = 'es-es'
ALLOWED_HOSTS = ['127.0.0.1']

STATIC_URL = '/static/'
STATIC_ROOT = BASE_DIR / 'static'

INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',

    'nombre_aplicacion',
```

```
]
```

5. Configuración de URLs y Vistas

>> URLs del proyecto

```
from django.contrib import admin
from django.urls import path, include

urlpatterns = [
    path('admin/', admin.site.urls),
    path('nombre_app/', include('nombre_app.urls')),
]
```

>> URLs de la aplicación

```
from django.urls import path
from . import views

urlpatterns = [
    path('', views.principal, name='principal'),
    path('<int:pk>', views.detalle_post, name='detalle_post'),
]
```

6. Modelos (models.py)

```
from django.db import models

class Post(models.Model):
    titulo = models.CharField(max_length=200)
    autor = models.CharField(max_length=60)
    cuerpo = models.TextField()

    def __str__(self):
```

```
        return self.titulo
```

7. Vistas (views.py)

```
from django.shortcuts import render, get_object_or_404
from .models import Post

def principal(request):
    posts = Post.objects.all()
    return render(request, 'nombre_app/principal.html', {'posts':
posts})

def detalle_post(request, pk):
    post = get_object_or_404(Post, pk=pk)

    return render(request, 'nombre_app/detalle_post.html',
{'post': post})
```

8. Templates y static files

>> Estructura recomendada

```
nombre_app/
└── static/
    └── nombre_app/
        ├── css/
        │   └── style.css
        └── js/
            └── main.js

└── templates/
    └── nombre_app/
        ├── base.html
        ├── principal.html

        └── detalle_post.html
```

>> Template base (base.html)

```
<!-- templates/nombre_app/base.html -->
<!DOCTYPE html>
<html>
<head>
    <title>Mi wiki</title>
    <link rel="stylesheet" href="{% static
'nombre_app/css/style.css' %}">
</head>
<body>
    <h1>{% block titulo %}{% endblock %}</h1>

    {% block contenido %}{% endblock %}

    <script src="{% static 'nombre_app/js/main.js' %}"></script>
</body>

</html>
```

>> Template heredado (principal.html)

```
<!-- templates/nombre_app/principal.html -->
{% extends 'nombre_app/base.html' %}

{% block titulo %}Entradas del blog{% endblock %}

{% block contenido %}
<ul>
    {% for post in posts %}
        <li>
            <a href="{% url 'detalle_post' pk=post.pk %}">
                {{ post.titulo }} ({{ post.autor }})
            </a>
        </li>
    {% endfor %}
</ul>

{% endblock %}
```

>> Paso de datos a templates

```
def principal(request):
    context = {
        "titulo": "Mi primer post",
```

```
        "autor": "JI",
        "cuerpo": "Este es mi primer post"
    }

    return render(request, 'nombre_app/principal.html', context)
```

```
<h1>{{ titulo }}</h1>
<p>Autor: {{ autor }}</p>

<p>{{ cuerpo }}</p>
```

9. Migraciones

```
## Crear migraciones para la app
python manage.py makemigrations nombre_app

## Aplicar migraciones
python manage.py migrate nombre_app

## Migrar todo (todas las apps incluyendo admin y más)

python manage.py migrate
```

10. Panel de administración

>> Registrar modelo

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


admin.site.register(Post)
```

>> Crear superusuario

```
python manage.py createsuperuser
```

11. ORM y QuerySets

>> Acceder a la shell

```
python manage.py shell
```

>> Consultas básicas

```
from nombre_app.models import Post

Post.objects.all()
Post.objects.get(id=1)
Post.objects.filter(autor="Patricia")
Post.objects.create(titulo="Hola", autor="A", cuerpo="Texto")
Post.objects.order_by("autor")

Post.objects.order_by("-autor")
```

>> Filtros comunes

```
Post.objects.filter(titulo__contains="entrada")
Post.objects.filter(titulo__icontains="entrada")
Post.objects.filter(id__gt=5)
Post.objects.filter(autor__startswith="J")

Post.objects.filter(autor__endswith="z")
```

12. Template Tags y Filtros

>> Bucle for

```
<ul>
{% for post in posts %}
    <li>{{ forloop.counter }}. {{ post.titulo }}</li>
{% empty %}
```

```
    <li>No hay entradas</li>
{% endfor %}

</ul>
```

>> Variables útiles

- forloop.counter → n° de iteración (empezando en 1)
- forloop.first → True si es la primera vuelta
- forloop.last → True si es la última

>> Condicionales

```
{% if post.fpublicado %}
    <time>{{ post.fpublicado }}</time>
{% else %}
    <p>Aún no está publicado</p>

{% endif %}
```

>> Filtros útiles

```
{{ post.cuerpo|linebreaksbr }}
{{ post.titulo|upper }}
{{ post.titulo|lower }}

{{ valor|default:"N/A" }}
```

13. Comandos útiles de Django

```
python manage.py runserver
python manage.py makemigrations
python manage.py migrate
python manage.py createsuperuser

python manage.py shell
```