基于 Django 框架的 Python Web 开发

黄玉书

（武汉软件工程职业学院，湖北 武汉 430205）

摘 要 随着大数据、人工智能的发展，Python 语言也迅速地流行。Python 的免费、开源和具有大量的第三方库使得 Python 被广泛应用于各个领域，Python 在 Web 开发领域也引起了人们的关注。Django 作为最常用的 Python Web 框架，可以使开发者能够专注于业务部分，便于快速搭建高性能、优雅的网站。本文首先对 Django 框架进行了详细介绍，然后使用 Django 框架开发了一个简易图书管理系统。实践表明，使用 Django 框架进行 Python Web 开发非常方便、快捷，也便于系统的维护和管理。

关键词 Django 框架 Python Web 图书管理系统

中图分类号：TP31 **文献标识码**：A **文章编号**：1007-0745(2022)11-0100-03

1 Django 框架

Django 是一个开源 Web 应用框架[1]，使用 Python 语言编写，Django 提供了许多网站后台开发经常用到的模块，可以使开发者能够专注于业务部分，便于快速搭建高性能、优雅的网站。

Django 借用了 MVC 的设计模式，使用 MTV 的框架模式[2]。该架构中的三部分组成分别为：模型（M）、模板（T）、视图（V）。各部分说明如表 1 所示。

在 Django 项目中，数据模型在 models.py 文件中定义；模板文件主要是在 Templates 目录下存储的，Templates 目录需手动创建与配置；业务逻辑是在视图文件 views.py 中实现。除此之外，还有一个用于实现路由分发功能的 urls.py 文件也在 Django 项目中有着非常重要的作用。

Django 工作流程示意图如图 1 所示。

启动项目后，打开浏览器并输入要访问的 URL 后回车，浏览器会向 Web 服务器发起请求，Web 服务器会将请求传递到要处理该请求的 Django 项目，项目中的 urls.py 文件根据 URL 地址将请求交给 views.py 中相应的视图进行处理（此时涉及数据存取，通过 models.py 文件与数据库交互），并将处理结果发送给模板进行渲染，最后将响应数据返回到 Web 服务器。

2 基于 Django 的图书管理系统的开发及实现

2.1 环境安装

安装 Python 3.7.4 版本、安装 PyCharm 工具并配置好 Python 解释器、安装 Django2.2.4 版本。

2.2 创建 Django 项目

打开 PyCharm，在终端（Terminal）中，输入如下命令创建 libManage 项目。

```
django-admin startproject libManage
```

2.3 创建 bookApp 应用

在终端（Terminal）中输入如下命令。

```
python manage.py startapp bookApp
```

找到 libManage 子文件夹中的 settings.py 文件中的 INSTALLED_APPS 字段，然后在该字段的末尾添加一行代码“‘bookApp’,”将应用添加到项目中。

2.4 创建“图书”模型

在 bookApp 应用中的 models.py 文件[3]中创建“图书”（Book）模型。编写如下代码：

```
class Book(models.Model):
    bookName = models.CharField(max_length=100,verbose_name='书名')
    bookNo = models.CharField(max_length=100,verbose_name='书号')
    bookAuthor = models.CharField(max_length=100,verbose_name='作者')
    bookPrice = models.DecimalField(max_digits=6,decimal_places=2, verbose_name='价格')
```

模型创建完成后需要将创建的模型同步到数据库系统中，在终端（Terminal）中输入命令“python manage.py makemigrations”，然后在终端（Terminal）中输入命令“python manage.py migrate”完成数据库模型的同步操作。

2.5 编写视图处理函数

在 bookApp 应用下的 views.py 文件中编写视图处理函数，代码如下所示。

表 1 MTV 框架模式说明

部分	职责
模型（M）	数据操作层。主要用来定义数据模型，封装对数据库层的访问。
模板（T）	表现层。主要是负责将页面呈现给用户。
视图（V）	业务逻辑层。主要是调用模型和模板，实现业务逻辑，是模型与模板之间的桥梁。

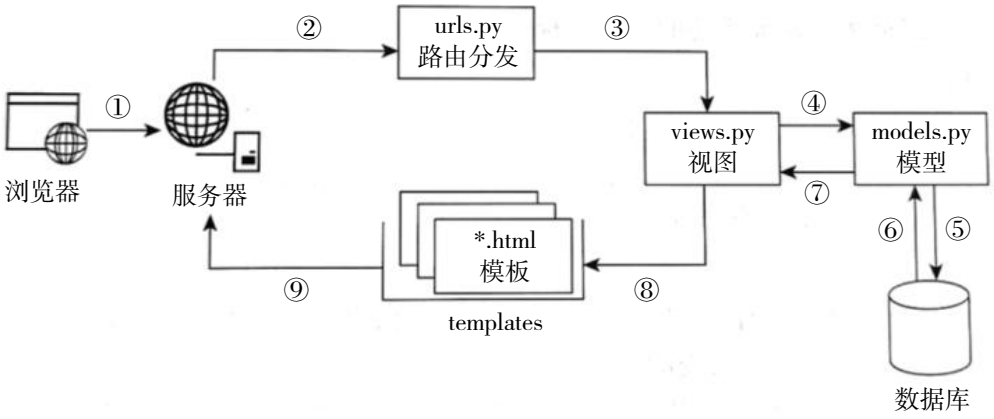

图 1 Django 工作流程示意图

```
from django.shortcuts import render
from book.models import Book
def home(request):
    books = Book.objects.all()
    return render(request,’index.html’,{
        ‘books’:books
    })
```

2.6 制作“图书信息”页面

在“图书信息”页面使用了 Bootstrap3 系列版本的前端框架。在 bookApp 应用下创建一个 static 文件夹，将官网下载的 Bootstrap3 源码中的 css、fonts 和 js 子文件夹拷贝到 static 文件夹中。

在 bookApp 应用下新建一个 templates 文件夹，并新建一个 index.html 文件，展示图书信息，代码如下所示。

```
{% load staticfiles %}
<html>
  <head>
    <meta charset=” utf-8” >
     <meta http-equiv=” X-UA-Compatible” cont
ent=” IE=edge” >
      <meta name=” viewport” content=” width=
device-width, initial-scale=1” >
    <title> 图书信息 </title>
     <link rel=” stylesheet” href=” {% static ‘css/
bootstrap.min.css’ %}” />
  </head>
  <body>
   <div class=” container” >
    <div class=” row” >
        <div class=” col-md-12” style=” margin-
top: 30px;” >
          <table class=” table table-bordered” >
            <thead>
              <tr>
                <th> 序号 </th>
                <th> 书名 </th>
                <th> 书号 </th>
                <th> 作者 </th>
                <th> 价格（元） </th>
              </tr>
            </thead>
            <tbody>
              {% for book in books %}
              <tr>
                <td>{{forloop.counter}}</td>
                <td>{{book.bookName}}</td>
                <td>{{book.bookNo}}</td>
                <td>{{book.bookAuthor}}</td>
                <td>{{book.bookPrice}}</td>
              </tr>
```

序号	书名	书号	作者	价格（元）
1	Python 数据可视化实战	978-7-115-12345-2	李四	59.80
2	MySQL 数据库技术与项目应用教程	978-7-115-12345-5	王小辉	49.80
3	Python 数据分析	978-7-115-12345-3	张小洋	39.80

图 2 页面效果图

```
            {% endfor %}
          </tbody>
        </table>
      </div>
    </div>
  </div>
 </body>
</html>
```

2.7 配置访问路由 URL

在项目子文件夹 libManage 中的 urls.py 文件中配置访问路由，编写如下代码。

```
from django.contrib import admin
from django.urls import path
from book.views import home
urlpatterns = [
   path('admin/', admin.site.urls),
   path('' ,home,name='home'),
]
```

2.8 Django 后台管理系统

Django 提供了一个现成高效的后台管理系统，在创建项目的过程中已经自动生成了一个编辑的后台。能够根据定义的模型自动生成管理模块，使用 Django 的功能的需要创建超级管理员和注册模型类 [4]。

（1）创建超级管理员。在终端（Terminal）中输入命令“python manage.py createsuperuser”根据弹出提示信息完成超级管理员的创建。

（2）注册模型类。在 bookApp 应用中的 admin.py 文件，编写如下代码。

```
from django.contrib import admin
from .models import Book
class BookAdmin(admin.ModelAdmin):
    list_display = ['bookName','bookNo','bookAuthor','bookPrice']
admin.site.register(Book, BookAdmin)
```

2.9 系统测试

在终端输入命令“python manage.py runserver”启动项目，在浏览器中输入 http://127.0.0.1:8000/admin 出现 Django 后台管理系统登录界面。输入超级管理员的用户名和密码登录。登录成功后，向 Books 模型中添加几条图书信息数据。

打开浏览器，输入网址 http://127.0.0.1:8000/，访问“图书信息”页面如图 2 所示。

通过 Django 后台管理系统对创建的图书模型进行增删改的操作，前端页面也会随之变化，实现了图书信息的管理功能。这样就搭建好了一个基于 Django 的简易图书管理系统。

3 结语

Django 框架自带大量常用工具，结合 Django 项目默认的 SQLite3 数据库完成了简易的图书管理系统开发。使用 Django 框架进行 Python Web 开发可以非常方便、快捷地完成 Web 开发，同时 Django 自带的后台管理系统也便于系统的维护和管理。

参考文献：

[1] 杨志庆 . 基于 Django 的 Blog 系统的开发与实现 [J]. 机电一体化 ,2013(09):69-72.
[2] 邱红丽 , 张舒雅 . 基于 Django 框架的 web 项目开发研究 [J]. 科学技术创新 ,2021(27):97-98.
[3] 白昌盛 . 基于 Django 的 Python Web 开发 [J]. 信息与电脑 ,2019(24):37-40.
[4] 钱彬 .Python Web 开发从入门到实战 (Django+Bootstrap)- 微课视频版 [M]. 北京 : 清华大学出版社 ,2020.