**注意事项：**

如果在开发过程中安装了新的组件，记得pip freeze

**数据库的设置**

**前提：mblog为项目名。Mainsite为其中的一个应用**

>修改mainsite/models.py：定义数据模型（即数据表及其字段）。Mainsite为该项目的一个app应用。本例增加了数据表Post

>使模型生效

(VENV) ~myDjango/mblog $ python manage.py makemigrations mainsite

(VENV) ~myDjango/mblog $ python manage.py migrate

>修改mainsite/admin.py将模型在Admin管理页面中注册：将Post纳入管理，

先导入（import）Post类（即Model），然后通过admin.site.register(Post)进行注册

>通过Admin管理界面增加Post数据表的记录

前提：创建Admin管理界面的管理员账号和密码——

Python manage.py createsuperuser

>修改admin.py，创建“表名Admin”类并注册：

创建PostAdmin类并注册，使Admin管理界面能够显示除第一个字段外的其他字段

**读取数据库的内容并展示到页面中**

>修改mainsite/view.py

导入模型： from .models import Post

定义内容输出函数：首页函数homepage和展示具体内容函数showpost

>修改mblog/urls.py

从views.py中导入内容输出函数：from mainsite.views import homepage

在urlpatterns[ ]中，将网址和内容输出函数对应（批注A）

**模板的使用**

>创建templates文件夹。与mblog、mainsite同级

>修改mblog/settings.py

设置模板文件夹的位置：修改TMPLATES->DIRS

>在url.py中创建网址和views.py中函数的对应关系。（参考批注A）

>在templates文件夹中创建.html文件

>运行程序，以objects.all()在views.py中取得数据

>以render函数（渲染函数）把数据送到指定的模板文件（如index.html）中

>将得到的结果交给HttpResponse输出给浏览器

**共享模板的使用**

>在template文件夹中新建base.html、header.html、footer.html。

>base.html

① include引入header.html和footer.html

② 创建block。此案例创建了3个block：title、headmessage、content

>其它文件，如index.html

① 继承base.html ：{% extends ‘base.html’ %}

② 不再需要<html></html>等标签，因为已经在base.html中定义过了

③ 提供base.html 中定义的区块（block）内容

**CSS、JS和静态文件的使用**

> Bootstrap的引用

将Bootstrap的CDN链接复制到base.html的</head>前面。这样继承base.html的文件中都可以使用Bootstrap的功能

> 图像文件的使用

新建static文件夹（和template、APP应用文件夹）

在static文件夹下，新建images、css、js文件夹

修改settings.py，指定静态文件要放置的位置

修改.html，加入 {% load staticfiles %}

> HTML内容处理，如在文章内容中加入图片外链 <img src=http://xxxxxx.jgp/>

在.html输出时，需要加safe过滤器：{{ post.body | safe }}

>Markdown语句解析与应用

在虚环境VENV中安装相关组件：

$ pip install django-markdown-deux

$ pip freeze > requirements.txt

修改settings.py，在INSTALLED\_APPS中，加入markdown\_deux

在.html文件中加载MarkDown语句标记：{% load markdown\_deux\_tags %}

在.html输出时，加markdown过滤器：{{ post.body | markdown }}

**数据库初步介绍**

1. 注意随时makemigrations和migrate
2. 可以通过python manage.py shell来在Python的交互界面中进行数据表的操作
3. 通过 Product.objects.all()函数取得的数据，数据类型是QuerySet
4. 可通过def \_unicode\_(self)来定义实例打印的结果

**网址的对应与委派**

在URL正则中加入要传送的参数名称 P147

验证RE设计URL的正确性：<http://pythex.org/>