开源软件基础课程报告

姓名	学号	Github 账号	分工	成绩
黎碧怡	201592057	bibiwannabe	组长	
叶俊宇	201592036	mengshi23	成员	
胡智勇	201592029	MidAutumn10100101	成员	
史久琛	201592449	shijiuchen	成员	
齐保坤	201592336	BOOMKUN	成员	

报 告 题 目 基于 Django 框架及 mysql 数据库的博客网站

项 目 网 址 https://github.com/bibiwannabe/Django-blog-with-basic-function

完 成 日 期 2018 年 1 月 6 日

大连理工大学软件学院

目录

1 项目概述

1.1 项目背景

BLOG，把它翻译成我们所熟悉的意思就是“网络文章”，现在一般叫做“博客”。具体说来，使用浏览器或者某些程序，在互联网上去创作、书写、发贴和刊登自己的文章、照片等信息的人就是博客。

随着互联网的发展，知识交流正越来越频繁与普及，在 IT 行业，开源精神正被越来越多的人所推崇，其中开源社区更是知识交流的聚集地。在这种趋势下，拥有属于自己的技术博客很有必要，越来越多的网络用户希望能够在网络平台上更多地展现自己的个性，更方便地与人互动交流。随着计算机网络的飞速发展，博客已经成为写网络日志必不可少的一种工具，也是一种简单有效的提供网络用户之间进行在线交流的网络平台，通过其可以结交更多的朋友，表达更多的想法，它随时可以发布日志，方便快捷。访客可以直接在个人 Blog 上留言，如提出问题或意见等。个人博客的发展，也已经成为广告商业务拓展的重要领域。总之，Blog 本着开源精神，是未来信息化教育和个人知识管理的强大而简单易用的工具。

1.1 项目简介

本系统利用 Python 动态网络开发技术，以 MySQL 作为后台数据库，使用 Apache 配置 Web 服务器，结合 HTML/CSS/JavaScript 等脚本语言，以及配合多种网页开发工具，实现了个人日志发表和交流的开源社区平台——个人博客系统。

本个人博客系统设计的目的旨在建立功能简洁、结构灵活且精致、轻巧的个人博客网站。其中博客文章的管理作为本系统的主要目标，用户注册账号并登录后，可创建、修改、删除文章，可修改个人信息（头像、昵称、邮箱），并可按文章标题、及内容搜索关键字，查找相应文章分页显示。其中主页广告位根据点击量选取最热四篇文章连接，所有文章列表可按照热度（点击量）、最新（最新修改时间）进行排序并分页展示。

2 需求分析

2.1 可行性分析

2.1.1 技术可行性

技术可行性是本设计最关键的部分，也是其他可行性的基础。需要根据用户所提出的各种功能要求和限制条件，在技术的角度上研究本系统实现的可行性。系统的开发涉及多个方面的技术。包括了系统的软件和硬件，网络的环境，人员的技术水平，系统开发的各种相关理论。

在技术可行性方面需要考虑的有很多，首先是技术人员，无论任何工作，都是以人为本的，由人来完成，最后面向的用户也是人。凭借小组成员以前所学的软件开发方法和编程的知识，并且查阅相关资料和书籍可以为网站的开发提供足够的技术保障。

博客作为一款网络应用，需要注意的还有网络环境的问题，如何降低网络环境的波动对博客系统性能产生的影响。比如博客在只有少数用户的时候能够流畅的运行，但是在面对大量的用户时原有的资源就可能捉襟见肘，如何在低谷和高峰之间进行切换，在面对大量用户时不会导致明显的性能能下降，在面对少量用户时避免资源的浪费。

2.1.2 经济可行性

本系统基于 MySQL 开发，完全可以实现免费、开源，并且我们所开发的这套博客网站系统正式基于提高用户使用效率，节省工作时间，简化操作管理的理念来设计的。并且本系统是个人独立设计开发的，并不需要投入大量经费，系统今后的运行和维护也相当简便，无需投入额外的资金，其成本不会出现超于预期的可能。并且在其过程中也可以提高个人的开发水平，因此在经济上是完全可行的。在系统运行的过程中将定期进行系统备份，在遇到突发事件时只需将备份的系统在另外一台具有所需环境的计算机上即可还原，并且重新运行，系统的可靠性高，风险小。

2.1.3 操作可行性

在当今这个信息技术迅猛进步的时代，信息技术早已深入到社会中的每个地方，它为人们带来来便利的工作方式，优越的工作环境，因此也对人们的工作效率和操作水平提出了更高要求。在这种发展趋势下，减少费时费力的人工操作，通过各种智能化软件来提高工作效率和工作精度就成为了一个很重要的方向。

一个成功的博客应该是很友好的，对于任何一个熟悉上网方式的人都应该能够无障碍的使用它。同时它的功能又应该是全面的，用户可以用符合日常习惯的方式来实现所要达到的目的。对于一些新颖的功能，应该通过给予用户一些奖励的方式使用户对其产生兴趣，并且试用它。若是使用起来有一定的难度则应该给予一定的说明和引导。

本博客网站系统的用户平台面向普通注册用户，其界面简洁，采用 web 的可视化界面，普通注册用户只需要轻点鼠标和键盘就可以使用博客，并且文章的输入和维护均由博客的用户来完成。正是由于这些特性，本博客系统十分适于新手使用，无须学习复杂的教程便可以获得好的体验。因此本博客网站系统在操作上是可行的。

2.2 功能需求

2.2.1 用户注册并登陆

1）实现新用户的注册功能

2）实现已注册用户的登陆功能

3）实现图片验证码的功能保证登陆认证。

注：应判断用户是否已经存在，并且保证注册信息的真实性和注册密码的安全性。

2.2.2 文章管理

1）实现文章的创建

2）实现文章的查看

3）实现文章的修改

4）实现文章的删除

5）显示文章的创建时间和更新时间

注：查看文章应显示其点击量

2.2.3 个人信息管理

1）显示个人信息，包括用户手机号、用户名、邮箱、头像等

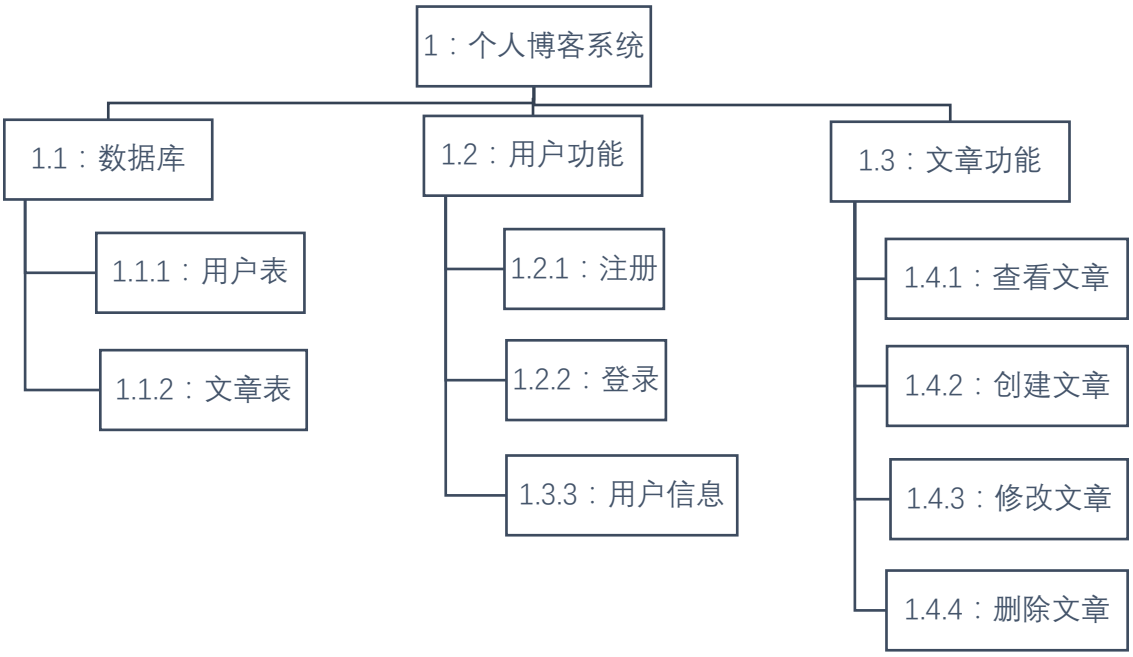
2）实现修改用户名、邮箱、上传头像的功能。

2.2.4 个性显示

1）主页文章显示，以两种方式实现：热门（点击量）最新（最新修改时间）
2）主页面广告位根据点击量选取最热四篇文章连接.
3）所有文章列表应按照热度（点击量）、最新（最新修改时间）进行排序并分页展示
4）实现快速返回主页功能
5）实现根据标题或关键字搜索文章功能

3 项目设计

本个人博客系统主要分为 3 个模块，分别是数据库模块、用户功能模块、文章功能模块，设计框架如下图



3.1 数据库设计

一个设计良好的数据库，可以使系统的实现变得非常的简单。同时，也可以使系统的执行速度变得很快。反之，一个设计混乱的数据库，不仅增加了吸引的管理实现过程，同时在系统的执行过程中，使得检索变得很慢。所以数据库的设计是一个系统设计很重要的步骤。本系统采用的数据库是 MySQL, 共有 2 张数据表，具体结构如下：

3.1.1 用户表

用户表的信息包括 用户名、密码、邮箱、电话、头像，如图

字段	属性	注释
id	int	用户编号
username	varchar	用户名
userpassword	varchar	密码
useremail	varchar	邮箱
userphone	varchar	电话
userpic	image	头像

3.1.2 文章表

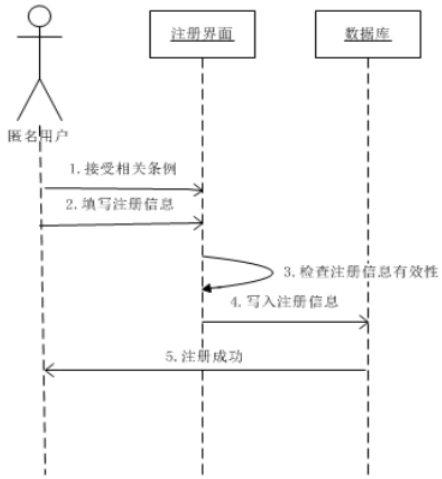
文章表的信息包括 文章编号、文章标题、文章内容、点击量、创建时间、最近修改时间、作者，如图

字段	属性	注释
id	int	文章编号
title	varchar	文章标题
content	HTML	文章内容
gclick	int	点击量
createdate	datatime	创建时间
modefidate	datatime	最近修改时间
uid	int	作者(外键)

3.2 用户功能设计

当用户输入博客网址进入博客时，首先需要进入登录页面。若为新用户，应先注册，而后登录。无论是新老用户登录，都应保证注册信息的真实性和注册密码的安全性。登录成功后，用户可以查看自己的个人资料并修改保存。以下是实现用户功能各个模块的顺序图。

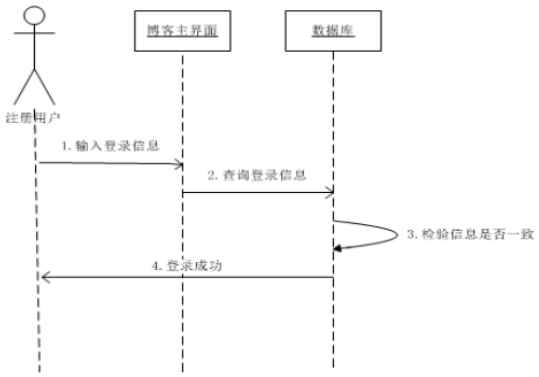
3.2.1 新用户注册



注册流程图

具体界面展示

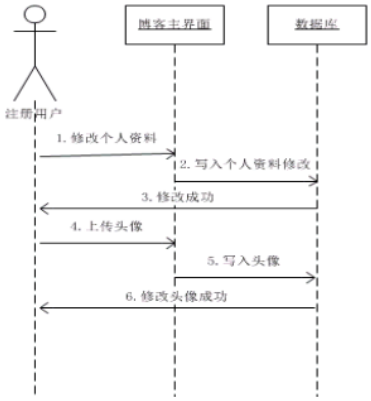
3.2.2 用户登录



登录流程图

具体界面展示

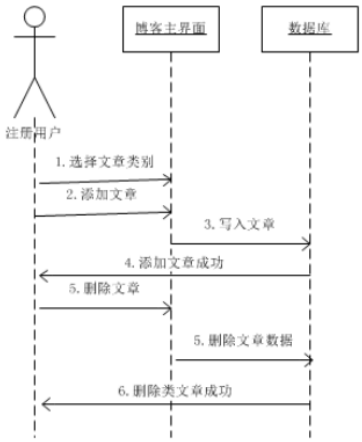
3.2.3 用户修改个人信息



修改个人信息流程图

修改用户个人信息界面展示

3.2.4 用户浏览、修改、创建、删除自己的文章



用户个人文章操作流程图

用户个人文章列表页面展示

用户个人文章内容可编辑界面展示

3.3 文章功能设计

当用户进入博客主页面时，页面上方广告位根据点击量选取最热四篇文章连接，页面下方显示所有文章列表，文章列表应按照热度（点击量）、最新（最新修改时间）进行排序并分页展示。此外，页面右侧搜索栏处，用户可以根据标题或关键字搜索文章。用户可以通过以上方式查看文章。用户也可以新建文章，编辑完成后发表。若用户修改或删除文章，可以先查看文章，决定下一步操作。

博客主页面展示

搜索界面展示

浏览他人文章界面展示

4 编码实现

4.1 数据库实现

本项目使用 mysql 数据库进行实现，数据库中主要由两张表构成，分别是 userinfo 表和 Article 表，它们分别用于存储用户基本信息和用户所写的文章的信息内容，Article 表中存有外键 user_id，作为参照 userinfo 表的外键，使得两张表发生关联。表分别定义在 user_articles 包的 model.py 和 user 包的 model.py 中，使用 python 内置的数据库框架进行定义。

4.2 HTML 的实现

网站界面的编写主要使用 html5 和 CSS 渲染相结合，其中再加入一些 javaScript 代码实现接受服务器端的简单逻辑的判断。主要包含网站主界面（主要实现文章的展示、分页以及按点击量或者日期进行排序）、注册界面（用于新用户的注册）、登录界面（用于已注册用户的登录）、我的信息界面（用于展示并且修改个人信息，包括图片、用户名、密码等）、新建文章界面（用于写新的博客）、查看具体文章界面（用于查看已经存在的文章的界面）。通过这些界面的相互配合，我们实现了一个小型博客网站的基本功能。

4.3 主要逻辑类 user/views.py 的实现

此逻辑类中主要是关于用户相关的动态后端逻辑。其中包括登录、注册、退出登录、用户信息修改、展示文章列表、展示用户自己的文章内容并且修改、删除文章、创建文章等操作。

登录功能在后端逻辑类中，根据 url 地址先跳转到对应登录界面，在界面中用户输入信息，在后端通过 session 会话获取登录信息，并在数据库中进行查询，是否存在该用户以及输入密码是否正确，如果匹配成功，则跳转到主界面，并且将登录标志位置为 true，若登录失败，跳转回登录界面，并且进行错误提示。

注册功能和登录功能类似，根据 url 地址先跳转到对应登录界面，在界面中用户输入信息，在后端通过 session 会话获取登录信息，并在数据库中进行查询，是否存在已经注册过的用户名，以及用户名和密码是否格式正确，如果注册成功，则跳入登录界面，数据库中存入一条新的用户信息，如果注册失败，则提示失败原因。

用户信息修改主要是使用用户在界面中修改的信息，进行获取，并在数据库中找到相应记录，并重新存入。

展示文章列表主要有两种展示方法，一种是按照日期进行排序，另一种是通过点击量进行排序，通过前端用户的选择，我们通过 python 内置的排序算法进行排序，并且将排序结果传回到前端。

展示用户自己的文章内容并且修改，可以根据用户的修改，查询数据库的相应记录，并作出相应的修改。

删除文章要求只能是用户删除自己所写的文章，对应删除数据库中的一条记录。创建文章则是在数据库中新增一条此用户的文章记录。

4.4 主要逻辑类 user_articles/view.py 的实现

此逻辑类中主要为文章相关静态操作功能，其中包括主页面具体的排序和分页的实现，用户浏览文章的实现，以及用户查询文章的实现。

排序算法主要使用 python 内置的算法，分别根据日期和点击量进行排序。分页的实现为 10 篇文章为一页额，通过向上取整得到文章的总页数。

用户浏览文章，主要是执行一次数据库的查询操作，并将结果显示到界面中，在用户退出时，将点击量+1，并写入数据库。

用户查询文章，通过用户的输入，在数据库中进行查询，如果查询成功，则返回查到的文章列表，如果查询失败，则作出相应提示。

4.5 路径选择与跳转的实现

路径的选择和跳转的实现主要定义在了 urls.py 这个文件中。通过相应的格式的定义，用户输入不同格式的 url 可以跳转到不同的页面并且有对应不同的操作。例如：

url(r'^index_(\d+)_(\d+)/$',views.index)，第一个数字是选择主页面文章显示的排序方式，第二个参数为显示第几页。

5 项目测试

1.

问题描述：在前端进行表单提交的时候报错：403 Forbidden. CSRF token missing or incorrect

原因：Django 提供的 CSRF 防护机制，在处理 POST 请求之前，django 会验证这个请求的 cookie 里的 csrftoken 字段的值和提交的表单里的 csrfmiddlewaretoken 字段的值是否一样。如果一样，则表明这是一个合法的请求，否则，这个请求可能是来自于别人的 csrf 攻击，返回 403 Forbidden.

解决方法：前端 form 表单语句下加入{% csrf_token %}

2.

问题描述：在已经添加{% csrf_token %}前提下，前端使用 ajax Post 的时候报错：403 Forbidden. CSRF token missing or incorrect

原因：Django 提供的 CSRF 防护机制。

解决方法：使用 Djago 自带装饰器@csrf_token，使用方法：

From Django.views.decorators.csrf import csrf_exempt

在相应相应函数前加上@csrf_exempt

3.

问题描述：文章 modles 中的日期（DateTimeField）属性显示不正确，比当前时间早 8 个小时。

原因：setting 中默认时区设置不对。

解决方法：在 setting 中修改以下参数

TIME_ZONE = 'Asia/Shanghai'

USE_TZ = False

4.

问题描述：在用户上传个人头像图片时，上传成功，Django 后台获取头像图片路径错误。

原因：没有在配置中加入 MEDIA_ROOT 和 MEDIA_URL 设置。

解决方法：setting 中加入如下配置：

MEDIA_ROOT= os.path.join(BASE_DIR, 'static').replace('\\', '/')

MEDIA_URL = '/static/images/'

并在 USER 应用下的 urls.py 中加入

urlpatterns = [ ]+ static(settings.MEDIA_URL,document_root=settings.MEDIA_ROOT)

5.

问题描述：在用户未曾登录的状态下没有禁止用户访问创建文章、修改文章的界面。

原因：没有在用户试图访问以上页面时验证用户登录情况。

解决方法：

（1） 在用户登录时，session 中存入用户 id 字段用于验证：

```
request.session['user_id'] = users[0].id
```

（2） 应用下创建 login_decorator.py，用于验证用户 session 中是否有[‘user_id’]字段，如没有则返回登录页面，提示用户登录。

```
def login(func):
    def login_fun(request,*args,**kwargs):
        if request.session.has_key('user_id'):
            return func(request,*args,**kwargs)
        else:
            red = HttpResponseRedirect('/user/login/')
            red.set_cookie('url',request.get_full_path())
            return red
    return login_fun
```

（3）在需要验证用户是否登录的响应函数前都加上装饰器@login_decorator.login