背包问题知识社区系统项目设计说明书

团队名称	WGYG
指导教师	代祖华
班　　级	19 卓越
组　　员	魏瑾川 郭清华 姚恪 古丽妮尕尔

目录

一、 引言

1.1 编写目的

本报告的目的是对背包问题知识社区系统进行详细设计说明，以便用户及项目开发人员了解产品详细的设计与实现。为开发人员提供开发参考书。本报告的预期读者有客户、项目经理、开发人员以及跟该项目相关的其他人员。

1.2 项目介绍

本系统是建立在 B/S 系统架构下基于 MySql 数据库，采用 Django 框架、Bootstrap 前端开发框架、jQuery 进行开发。该系统基本满足了背包问题知识社区系统的需求，用户界面友好。

平台允许用户自由的注册账号，登录平台。在平台内发表博客，为其他用户发表的博客，评论。用户可便捷的在平台内发表博客以分享自己有关背包问题的理解。当用户有任何问题时也可以在平台自由的发表问题等待其他专业的用户来回答，当用户觉得他人的回答解决了自己的问题即可以采纳该回答。平台还可让用户分享背包问题相关数据集及相关算法代码。平台为用户提供相关文献资源供用户阅览。为了让用户交流进一步方便，平台还提供了私信功能，以用户交流进一步方便。最后为了良好的用户体验，平台还有消息提醒和搜索的功能。

二、 总体设计

2.1 需求概述

2.1.1 功能需求

（1） 用户模块

该平台的服务对象是互联网用户，用户在该平台注册后可使用该平台的功能。用户使用邮箱进行注册，然后在邮箱中收到邮件，用户经过验证后即可完成注册。如果用户忘记密码，还可以使用忘记密码功能来重置账号密码。用户还可进行个人信息修改。

（2） 首页问答模块

用户登录成功后可直接进入该页面。该页面展示该平台用户发起的提问，热门帖子以及用户可能感兴趣的问题。用户可在该页面进行回答，也可发起提问，可以点赞，评论他人的回答。

（3） 资源模块

该模块是为了帮助用户分享背包问题相关的代码，数据集以及文献，从而让用户可以在该平台上获取跟多有关背包问题的知识。

（4） 文章模块

该模块主要是让平台用户可以将自己的知识分享出来，给其他用户参考，帮助用户进行学习。用户可以编写自己的文章。

（5） 私信模块

该平台具有社交属性，因此用户之间可以对感兴趣的话题进行私聊，进行站内交流。

（6） 消息通知模块

当平台内的用户与其他用户之间进行互动时，该用户会收到系统的通知：

- 有人对文章进行了点赞
- 有人对文章进行了评论
- 有人回复了我的评论

（7） 全局检索模块

为了方便用户快速找到自己想要看到的内容，该模块可以为用户提供文章，文献，文件，用户，算法的检索。

2.1.2 非功能需求

（1） 良好的用户界面

用户界面应该简单易懂，让用户可以 0 学习成本使用该平台，平台的字体配色等应该统一，方便用户浏览。

（2） 高速响应时间

用户在使用登录，文章阅读，点赞，评论，文件上传等功能时的加载时间要尽可能短，提高用户体验。

（3） 稳定性

当平台流量变大时，平台的一切功能应正常工作，并且在发生意外情况时，用户的数据不会丢失。

（4） 安全性

平台应有抵御 DDos 攻击等黑客攻击行为的防御措施。

（5） 拓展性

平台在随着版本的迭代，用户需求的变化时可以增加相应功能，以满足用户需求。

2.2 软件结构

根据前述需求分析，该平台可分为以下七个功能模块，如下图 3-1 所示。

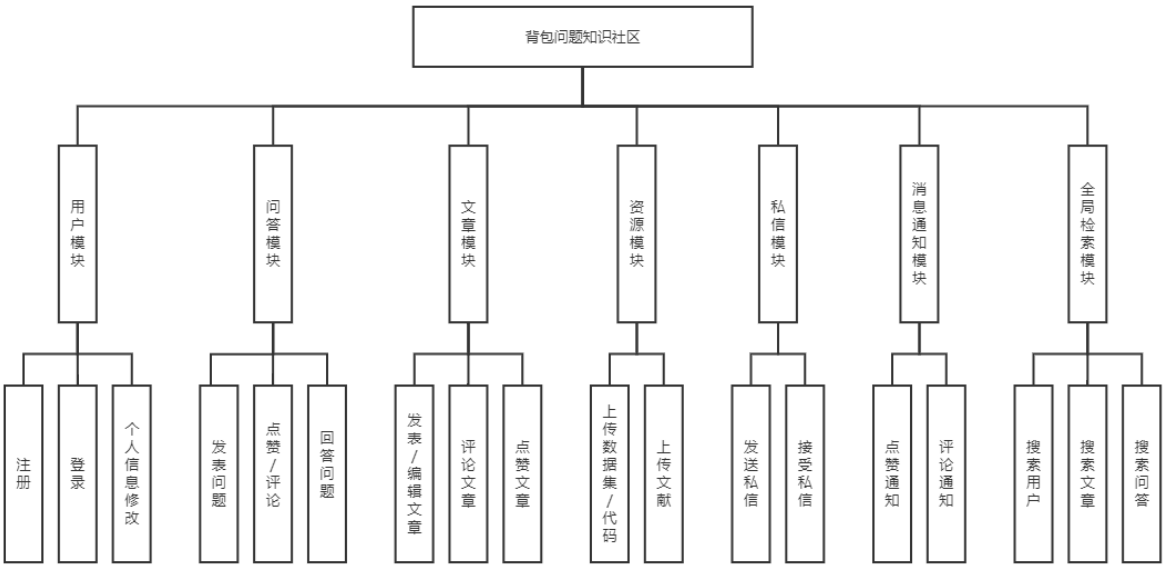

图 2-1 系统层次图

以下依次介绍每个功能模块的设计：

2.2.1 用户模块

用户模块是改平台的基础。平台没有游客模式，所以用户首先需要注册和登录平台。在注册部分，用户使用邮箱注册，填写邮箱，用户名，密码和确认密码之后，点击注册。

然后后端会给用户注册使用的邮箱发送验证邮件，用户点击邮件中的认证链接之后即完成注册。用户登录，输入注册时填写的用户名和密码，然后服务端对用户名和密码进行验证，验证通过即登录成功，进入首页。用户的个人信息的更新页面，不仅需要展示用户已经保存的个人信息，还要允许用户编辑已有的信息然后保存,所以需要在用户模块对应的 app 中的 view 中编写相应代码来分别实现上述的需求，同时这两个操作需要用户登录才有权限，所以还需要进行权限的验证。用户模块如图 2-2 所示。

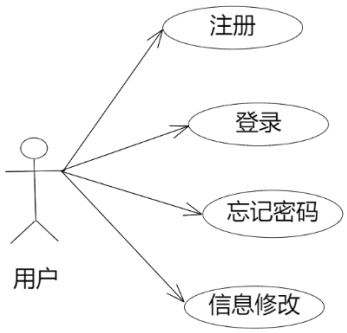

图 2- 2 用户模块

2. 2. 2 问答模块

首页问答模块是用户登录完成之后进入的平台首页，在该页面用户可以看到其他用户发表的问题，用户登录首页即可浏览问题列表。用户发表问题,对问题的评论和对问题回复的点赞和取消点赞，将评论或者点赞存入数据库，并且渲染到前端模板显示给用户。两个功能唯一的区别在于用户对所有回答都可以评论，但是对已经点过赞的回答只能取消点赞，所以对回答点赞的处理要先去查询用户是否点过赞，如果用户没有赞过，则添加点赞，如果用户已经点过赞，则取消点赞。最后就是用户可以删除自己已经发表的动态。问答模块如图 2-3 所示。

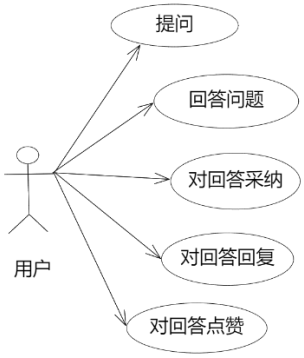

图 2- 3 问答模块

2.2.3 文章模块

文章模块，用户浏览平台所有已经发表的文章，用户点击文章 title 可以进入文章详情页，阅读完整的文章,。用户对文章的评论与动态模块的评论相同。同时用户也可以作为内容的创作者新建编辑一篇文章，编辑后可以选择直接发表或者，存入草稿箱以供下次继续编写完成之后发表。用户点击草稿箱中的文章即可重新编辑。文章模块如图 2-4 所示。

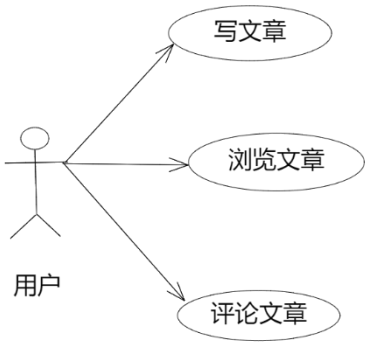

图 2-4 文章模块

2.2.4 资源模块

资源模块，用户可以在此浏览所有通过审核的资源，包括数据集，代码以及相关文献，同时用户也可以上传自己的相关资源。资源模块如图 2-5 所示。

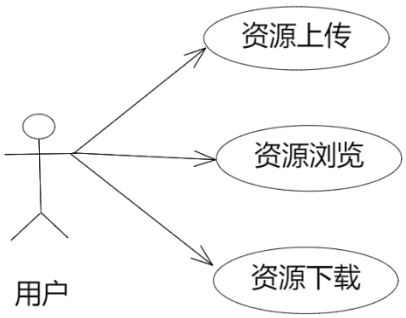

图 2-5 资源模块

2.2.5 私信模块

平台的私信功能让用户可以选择平台的其他用户自由交流。为了让用户有良好的聊天体验，用户间聊天消息的展示应是实时的。私信模块如图 2-6 所示。

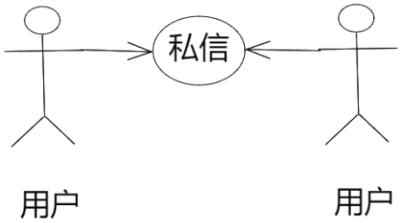

图 2- 6 私信模块

2. 2. 6 消息通知模块

当平台的用户对其他用户的发表的 object 产生行为时，该 object 的所有者则会实时的收到消息通知。例如当用户 A 对用户 B 发表的动态点赞、评论时，用户 B 则会实时的收到来自系统的通知“A 用户点赞、评论了我的动态”。因为平台涉及的功能模块较多，为了更好拓展性和代码的复用性，该部分需要设计一个通用的消息通知器接口，然后在其他诸如点赞，评论的功能模版调用该接口来实现内容操作的消息通知。消息模块如图 2-7 所示。

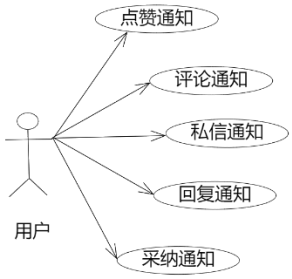

图 2- 7 消息通知模块

2. 2. 7 全局检索模块

搜索功能允许用户通过关键字搜索平台内的各种内容，包括文章，用户，动态，问答，标签等。给用户提供一个搜索框，当用户输入某一个字符串 query 时，服务端可以获取到该用户的 query，然后可以在动态，文章，问答，用户四个功能模块里面对应的数据表中去搜索这个 query。全局检索模块如图 2-8 所示。

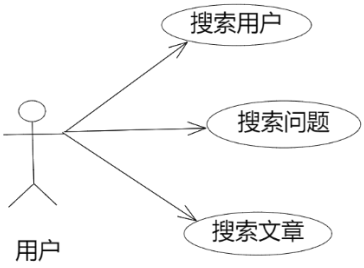

图 2- 8 全局检索模块

2.3 设计模式

由于本项目采用 Django 框架开发，因此采用 MTV 模式。MTV 模式本质上和 MVC 是一样的，都是为了各个组件之间保持松耦合关系，只是定义上有些不同。MTV 分别是指：

M（Model）代表模型：负责着业务对象和数据库的关系映射（ORM）；

T（Template）代表模板：负责以何种方式将页面展示给用户；

V（View）代表视图：负责业务逻辑，并在适当时刻调用 Model 和 Template。

除此之外，还包含一个路由分发器 Urls，它将各种 url 的页面请求分发给不同的 View 处理，View 再调用相应的 Template 和 Model。

下图为 MTV 的运作流程：

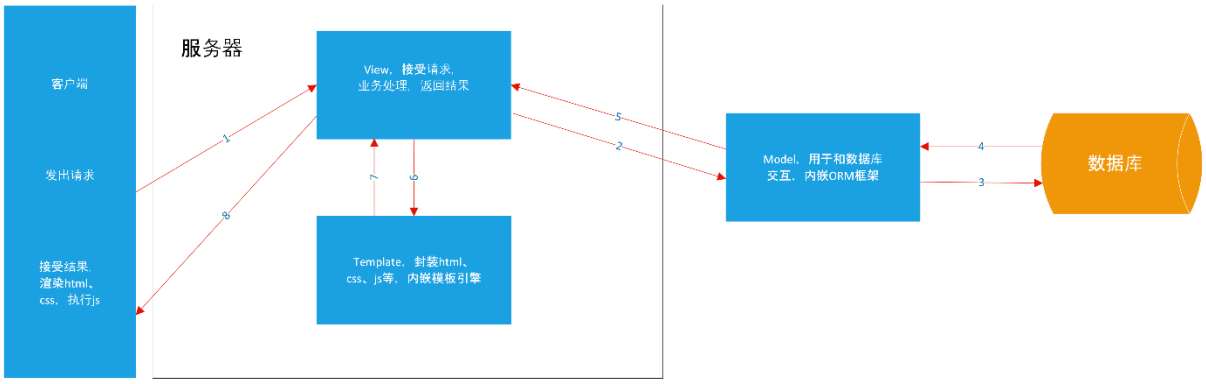

图 2-9 MTV 流程

三、 数据库设计

根据知识问答平台的业务需求，本小节给出了几个主要功能模块的数据库中表的设计。

（1） 用户表，主要存储用户的基本信息。具体结构见表 3-1。

表 3- 1 用户表

字段名	类型	描述
id	bigint	主键
account	varchar(32)	账号
password	varchar(50)	密码
email	varchar(50)	邮箱
phone	varchar(20)	手机号
avator	varchar(100)	头像
introduction	varchar(200)	简介
created_at	date	创建时间
update_at	date	更新时间

（2） 文章表，具体表结构见表 3-2。

表 3- 2 文章表

字段名	类型	描述
id	bigint	主键
title	varchar(50)	标题
content	longtext	内容
image	varchar(100)	文章图片
user	bigint	创建者 id
edited	Boolean	是否可编辑
tags	varchar(30)	标签
created_at	date	创建时间
update_at	date	更新时间

（3） 评论表，具体表结构见表 3-3。

表 3- 3 评论表

字段名	类型	描述
id	bigint	主键
user	bigint	用户
content	text	内容
liked	bigint	点赞用户
created_at	date	创建时间
update_at	date	更新时间

（4） 问题表，具体表结构见表 3-4。

表 3- 4 问题表

字段名	类型	描述
id	bigint	主键
user	bigint	提问用户
title	varcahr(50)	标题
content	text	内容
has_anser	boolean	是否有采纳的回答
tags	varchar(30)	标签
created_at	date	创建时间
update_at	date	更新时间

（5） 回答表，具体表结构见表 3-5。

表 3- 5 回答表

字段名	类型	描述
id	bigint	主键
user	bigint	回答用户
question	bigint	回答问题
is_answer	Boolean	回答是否被接受
created_at	date	创建时间
update_at	date	更新时间

（6） 私信表，具体表结构见表 3-6。

表 3- 6 私信表

字段名	类型	描述
id	bigint	主键
sender	bigint	发送者
receiver	bigint	接收者
message	text	内容
created_at	date	创建时间
update_at	date	更新时间

（7） 通知表，具体表结构见表 3-7。

表 3- 7 通知表

字段名	类型	描述
id	bigint	主键
actor	bigint	触发者
receiver	bigint	接收者
verb	varchar(50)	通知类别
created_at	date	创建时间
update_at	date	更新时间

（8） 资源表，具体表结构见表 3-8。

表 3- 8 资源表

字段名	类型	描述
id	bigint	主键
url	vatchar(200)	资源地址
type	varchar(2)	资源类别(0:代码 1:数据集 2:文献)
created_at	date	创建时间
update_at	date	更新时间

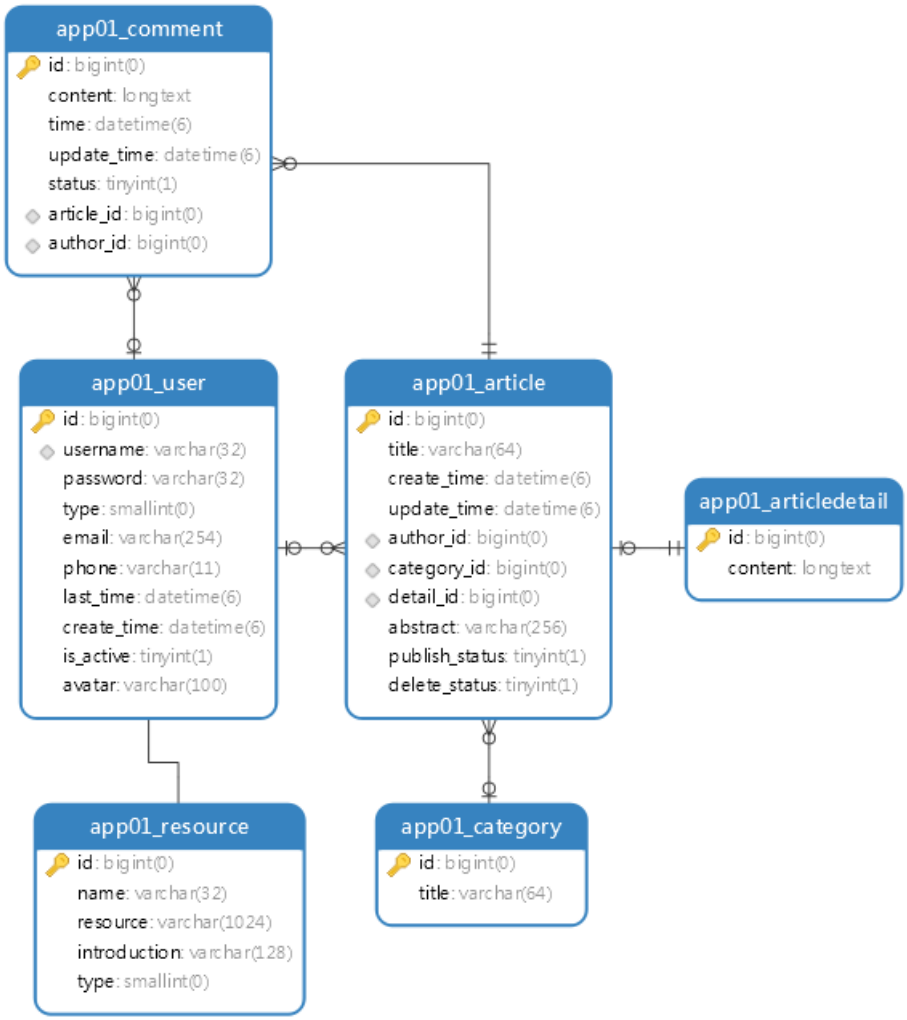
app01_comment
id: bigint(0)
content: longtext
time: datetime(6)
update_time: datetime(6)
status: tinyint(1)
article_id: bigint(0)
author_id: bigint(0)
app01_user
id: bigint(0)
username: varchar(32)
password: varchar(32)
type: smallint(0)
email: varchar(254)
phone: varchar(11)
last_time: datetime(6)
create_time: datetime(6)
is_active: tinyint(1)
avatar: varchar(100)
app01_article
id: bigint(0)
title: varchar(64)
create_time: datetime(6)
update_time: datetime(6)
author_id: bigint(0)
category_id: bigint(0)
detail_id: bigint(0)
abstract: varchar(256)
publish_status: tinyint(1)
delete_status: tinyint(1)
app01_articledetail
id: bigint(0)
content: longtext
app01_resource
id: bigint(0)
name: varchar(32)
resource: varchar(1024)
introduction: varchar(128)
type: smallint(0)
app01_category
id: bigint(0)
title: varchar(64)

图 3-1 数据库模型图

四、 软件重用方案

4.1 Django 模板继承

在开发前端页面的过程中，存在多个页面用共同的部分，这是便可将这部分单独抽离出来。写成一个模板，然后在需要的时候进行继承，这样可以避免代码冗余，方便维护，仅仅修改父模板便可以在多个引入改模板的页面生效。

例如：该系统的大多数页面都包含一个导航栏，此时我们便可将导航栏的代码抽离出来，单独编写为一个文件，同时预留相应位置，供子模板修改。

代码 4-1 父模板 layout.html

```
<!DOCTYPE html>
<html lang="zh-CN">
<head>
</head>
<body>
{% block container %}

{% endblock %}
</body>
</html>
```

代码 4-2 子模板 index.html

```
{% extends ‘layout.html’%}
{% block container %}
这里编写子模板相应的代码
{% endblock %}
```

4.2 Form 类重用

Django Form 组件用于对页面进行初始化，生成 HTML 标签，此外还可以对用户提交对数据进行校验。在编写代码的过程中会有多个页面要实现类似的功能，他们所用到的代码逻辑相同，于是我们便可以创建一个定制的 Form 类，所有要实现该功能的模块只需继承该类，并进行特定代码的编写即可。这样一方面使得系统的可维护性大大提高，同时也减少了代码的冗余程度。

五、 类的设计

5.1 关键类的重点服务

本系统要实现的关键类为 view 相关的类，这些类为多个视图函类，简称视图，是一个简单的 Python 类，它接受 Web 请求并且返回 Web 响应。响应可以是一张网页的 HTML 内容，一个重定向，一个 404 错误，一个 XML 文档，或者一张图片。无论视图本身包含什么逻辑，都要返回响应。因此该系统的关键类为视图类。

对于该系统来说，关键的类有如下几个：

（1）Login：负责用户的登录。

（2）Logout：负责用户的注销

（3）Register：负责用户的注册。

（4）Article：负责博客的增删查改

（5）Comment：负责评论的增删查改

（6）Resource：负责资源的增删查改

（7）User：负责用户信息的增删查改

（8）Admin：负责管理员的权限管理以及增删查改

（9）Backend：负责后端系统的管理，如资源审核、文章审核

（10）Forms：form 类的父类，负责将页面初始化，动态呈现 HTML 标签

5.2 Model 类

模型准确且唯一的描述了数据。它包含您储存的数据的重要字段和行为。一般来说，每一个模型都映射一张数据库表。

- 每个模型都是一个 Python 的类，这些类继承 django.db.models.Model
- 模型类的每个属性都相当于一个数据库的字段。
- 利用这些，Django 提供了一个自动生成访问数据库的 API

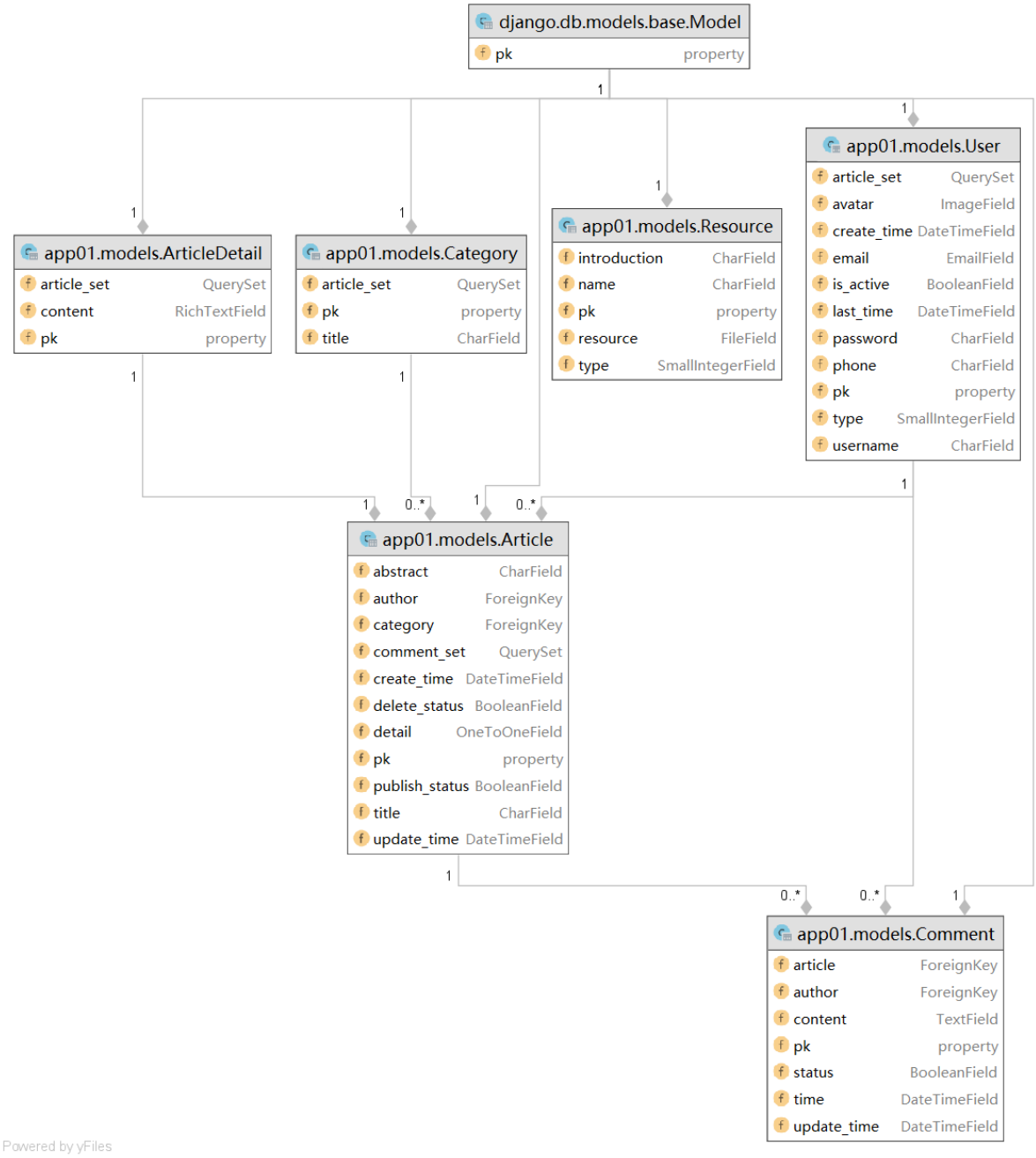

图 5- 1 Models 类依赖图

本次项目的 Models 类有以下：

Model：所有模型类的基类

User：用户类

Article：文章类

ArticleDetail：文章详情类

Category：文章类别类

Resource：资源类

Comment：文章评论类

这五个类分别对应数据库中的 5 张表，每个类中的各个属性分别对应相应数据库表的相应字段。

5.3 Form 类

Django Form 组件用于对页面进行初始化，生成 HTML 标签，此外还可以对用户提交对数据进行校验。因此根据需求分析的结果本项目有以下 5 个 Form 类，他们都继承了 django.forms.models.ModelForm 类。

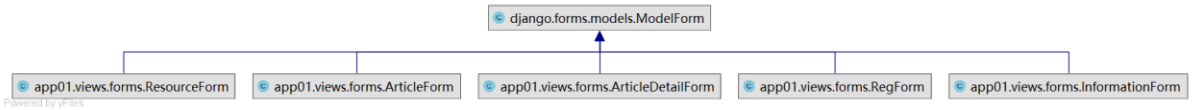

图 5-2 Form 类

5.4 Views

本项目包括 7 个 view，分别为 account，article，backend，comment，index，paper 和 resource，接下来为以上 7 个 view 的详细介绍。

5.4.1 account

该 view 有 login、register、logout 和 information 四个方法，主要处理用户的登录、注册、注销以及个人信息的管理。

1.login：该方法的参数为 request，返回值为一个页面。具体逻辑如图 5-3 所示。

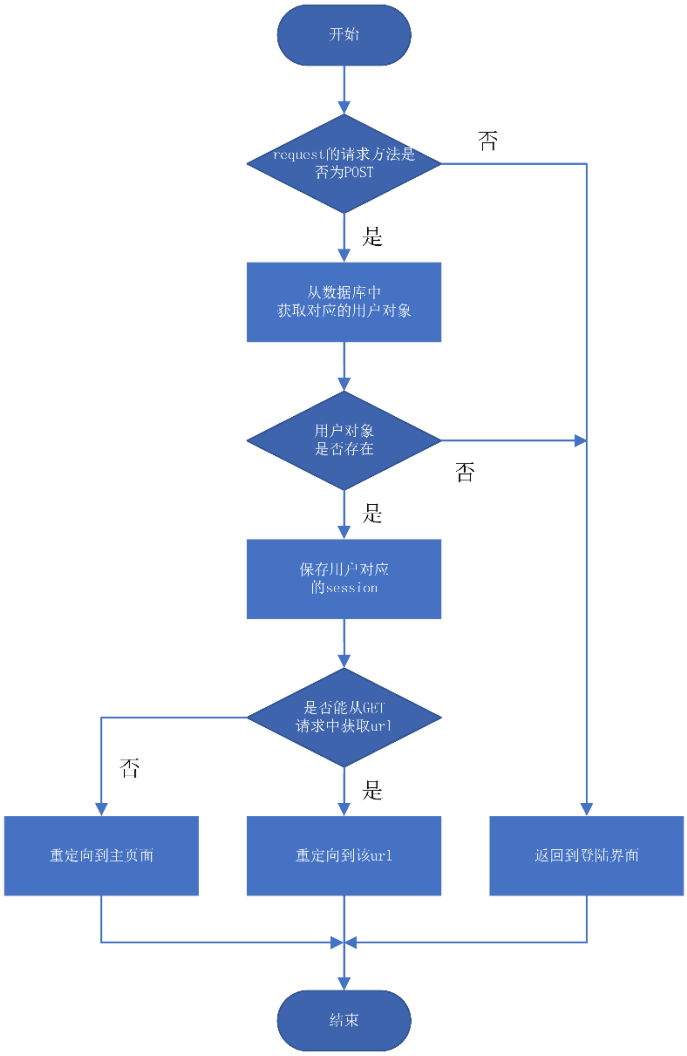

图 5-3 login 逻辑

2.register：该方法的参数为 request，返回值为一个页面。具体逻辑如图 5-4 所示。

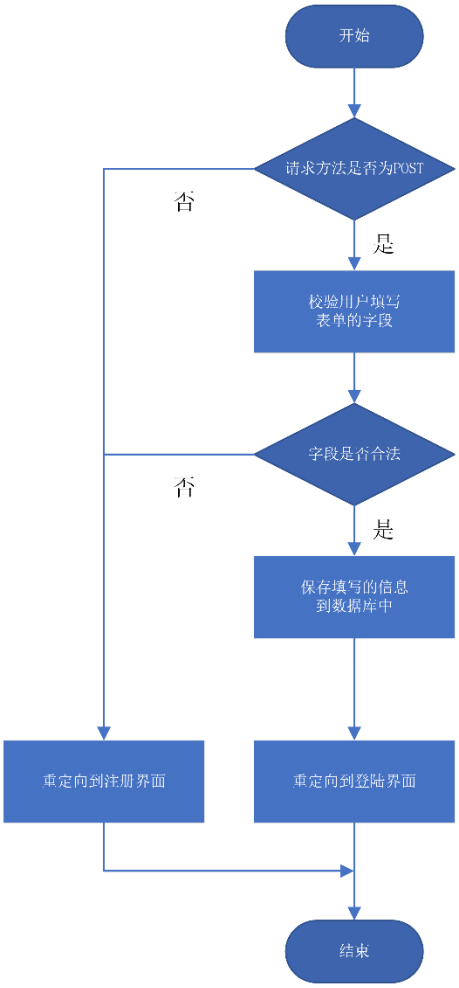

图 5-4 register 逻辑

3.logout：该方法的参数为 request，返回值为一个页面。具体逻辑如图 5-5 所示。

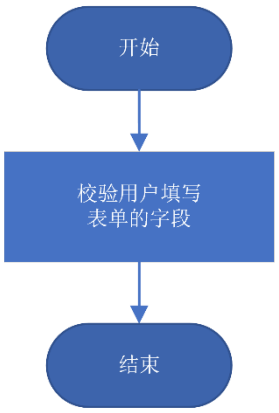

图 5-5 logout 逻辑

4.information：该方法的参数为 request，返回值为一个页面。具体逻辑如图 5-5 所示。

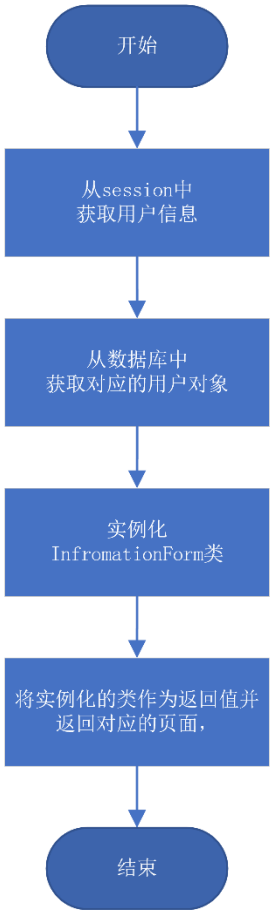

图 5-6 个人中心逻辑

5.4.2 index

该 view 有 index 方法，主要处理文章列表页的逻辑。

index 的逻辑如图 5-7 所示。

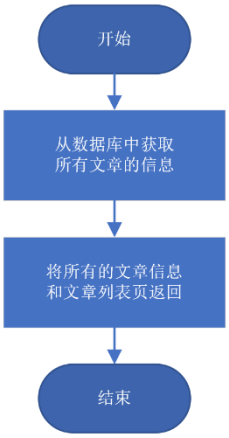

图 5-7 index 的逻辑

5.4.3 article

该 view 有 article 方法，主要处理文章详情页的逻辑。

article 的逻辑如图 5- 8 所示。

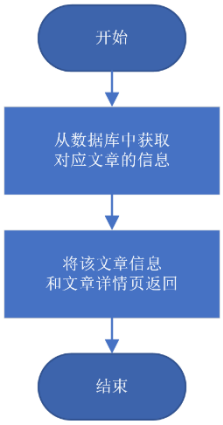

图 5- 8 article 的逻辑

5.4.4 backend

该 view 有 article_list、article_add、article_edit 和 article_delete 方法，主要处理文章管理页面的逻辑。

1. article_list 文章列表，其逻辑如图 5-9 所示。

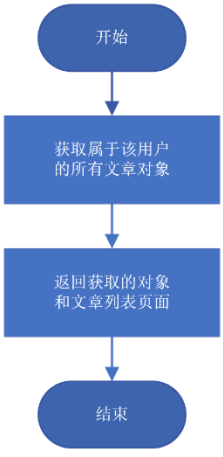

图 5-9 article_list 逻辑

2.article_add 新增文章，其逻辑如下图 5-10 所示。

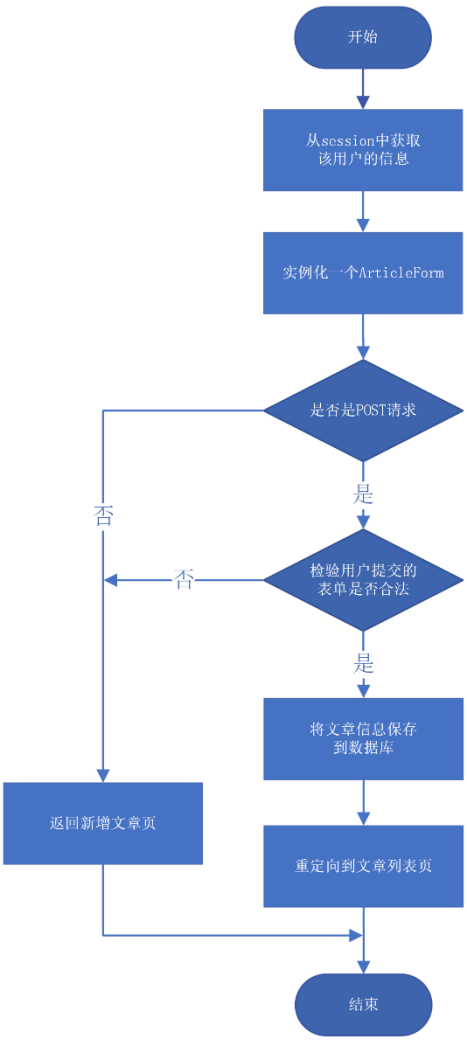

图 5-10 article_add

4.article_edit 文章便捷，其逻辑如下图 5-11 所示。

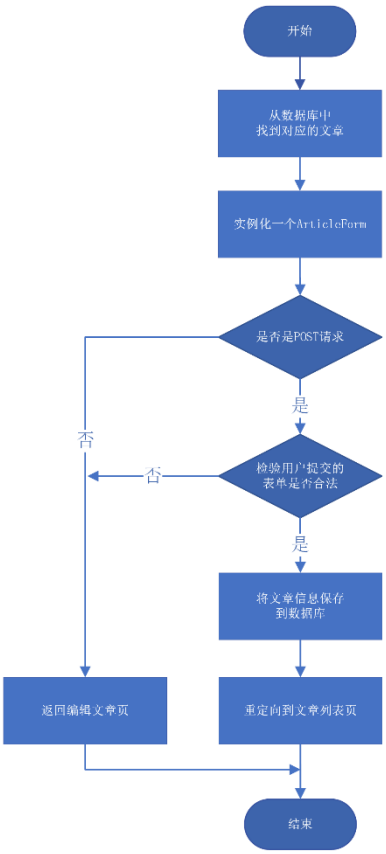

图 5-11 article_edit 逻辑

5.article_delete 删除文章，其逻辑如下图 5-12 所示。

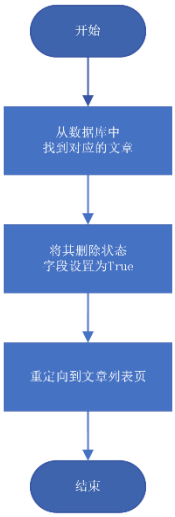

图 5-12 article_delete 逻辑

5.4.5 comment

该 view 有 comment 方法，主要处理评论功能的逻辑。

5.4.6 resource 和 paper

resource 有 resource_list 和_add 方法，主要处理代码资源页面的逻辑。

paper 有 paper_list 主要处理文献资源页面的逻辑

resource 处理资源上传

1.resource_list，逻辑如图 5-13 所示。

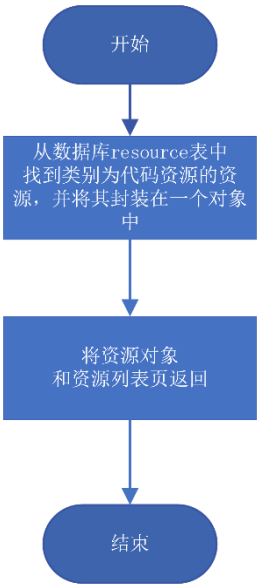

图 5-13 resource_list 逻辑

2.paper_list，逻辑如所图 5-14 示。

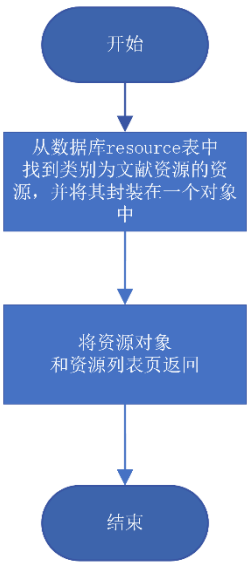

图 5-14 paper_list 逻辑

3.resource_add，逻辑如图 5-15 所示。

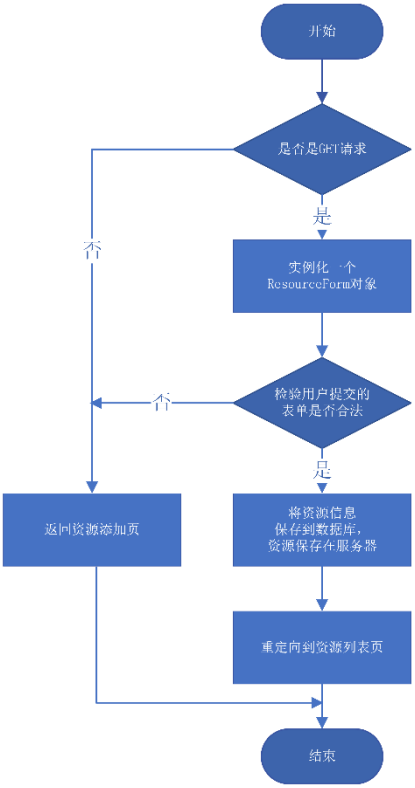

图 5-15 resource_add 逻辑