**背包问题知识社区系统**

详细设计说明书

**目录**

[1 引言 2](#_Toc105944290)

[1.1 编写目的 2](#_Toc105944291)

[1.2 背景 2](#_Toc105944293)

[2 设计概述 2](#_Toc105944294)

[2.1 任务和目标 2](#_Toc105944295)

[2.1.1 需求概述 2](#_Toc105944297)

[2.1.2 运行环境概述 2](#_Toc105944300)

[2.1.3 条件与限制 2](#_Toc105944303)

[2.1.4 详细设计方法和工具 2](#_Toc105944306)

[3 系统详细需求分析 3](#_Toc105944309)

[3.1 详细需求分析 3](#_Toc105944310)

[功能需求： 3](#_Toc105944311)

[3.2 总体结构设计 3](#_Toc105944316)

[4 系统详细设计 4](#_Toc105944319)

[4.1 设计软件总体设计结构 4](#_Toc105944320)

[4.1.1 UI(User Interface)用户界面表示层 4](#_Toc105944321)

[4.1.2 BLL(Business Logic Layer)业务逻辑层 4](#_Toc105944322)

[4.1.3 DAL(Data Access Layer)数据访问层 5](#_Toc105944323)

[4.1.4 Common类库 5](#_Toc105944324)

[4.1.5 Entity Class实体类 5](#_Toc105944325)

[4.2 系统结构设计及子系统划分 5](#_Toc105944326)

[4.3 系统功能模块详细设计 5](#_Toc105944327)

[4.4 软件重用方案 7](#_Toc105944329)

[4.5 设计关键类的重点服务 8](#_Toc105944330)

[5 数据库系统设计 8](#_Toc105944332)

[5.1 数据库设计 8](#_Toc105944333)

[5.1.1 顶层数据流图 8](#_Toc105944334)

[5.1.2 一层数据流图 8](#_Toc105944335)

[5.1.3 二层数据流图 9](#_Toc105944337)

[5.1.4 ER图 9](#_Toc105944339)

[5.1.5 数据字典 10](#_Toc105944341)

# 引言

## 编写目的

## 为了了解以及学习背包问题以及相关的算法，设计实现一款由注册用户共享的背包知识社区系统。为便于学习和了解，本此知识社区系统应具备知识资源的上传、下载、检索以及审核等相关的功能，同时此次背包知识社区系统满足相关算法的演练和计算。在知识社区系统设计中，也设计了相关知识的主题论坛，便于用户更便捷的处理一些问题。

## 背景

待开发软件系统名称：背包问题知识社区系统

系统的基本概念：此次设计知识社区系统主要是为了解决背包问题以及相关的

算法而设计的

# 设计概述

## 任务和目标

## 此次详细设计的任务主要是实现一款由注册用户共享的背包问题知识社区系统

### 需求概述

### 系统至少具有资源上传、检索、资源审核与管理、背包问题主题知识论坛、资源

### 少具有资源上传、检索、资源审核与管理、背包问题主题知识论坛、资源类型至少包括：典型算法源代码、开源数据集、背包问题相关文献资源，系统设置管理员，用于系统用户管理、资源有效性审核。

### 运行环境概述

### 此次背包知识社区系统交付最终是基于APP的用户界面，背包知识社区系统是用

### Java进行编写。基于Android stduio的APP的设计与实现。

### 条件与限制

### 因此次设计所用软件的相关的限制，需要在安卓版本10以上的真机可以进行测

### 试运行。以及需要在最初的管理方面设计了相关的用户管理

### 详细设计方法和工具

### 在详细设计的过程之中，需要进行的是程序化的结构设计，其次之后是人机

### 界面的设计。在此次设计过程之中主要采用了HIPO图方法、IDEF（I2DEF）方法、E－R图，数据流程图、业务流程图、选用的CASE工具

# 系统详细需求分析

## 详细需求分析

## 功能需求：

## 1、此次背包问题知识社区平台具有用户登录以及管理员用户

## 2、系统至少具有资源上传、检索、资源审核与管理

## 3、此次社区平台具有对于背包问题讨论的论坛

## 4、能够在该平台进行相关的算法实践和应用。

## 总体结构设计

## 模块设计

## 

各模块功能：

1、注册用户模块：

功能有注册、登录。

其中注册需要手机号、密码和短信验证码；登录需要手机号和密码，用户上传需要识别的图片之后，系统执行图片处理模块。

2、首页模块：

功能查看学习有关背包问题的相关算法、资源的上传、审查、算法的编辑以及运行结果导出。

3、管理模块：

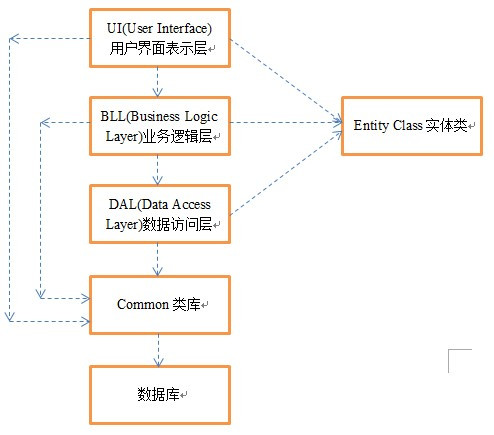
功能有管理用户信息、检查信息提取准确率。

其中管理用户信息模块可执行对用户信息的添加、修改、查询、删除功能。

# 系统详细设计

## 设计软件总体设计结构

系统采用三层架构模型，将应用系统划分为用户界面表示层、业务逻辑层、数据访问层，以及Entity Class实体类、Common类库组成，各层的关系如下图所示：



### UI(User Interface)用户界面表示层

负责与用户进行交互，显示、接受数据，与此同时，做一些简单逻辑处理，如：输入数据有效性判断、显示各种异常、处理Dataset记录集数据。它只与BLL(Business Logic Layer)业务逻辑层、Entity Class实体类两个项目发生关联，可能与Common类库发生关联。

### BLL(Business Logic Layer)业务逻辑层

是整个系统的核心，它承担了所有的逻辑判断，实现了程序的功能，它是灵活的。BLL层既是调用者，又是被调用者，因此，要适当的进行设计达到解耦的效果。BLL层只关联DAL层和实体类，可能关联Common类库。虽然BLL层被U层调用，但是BLL层无需关心UI层的情况。数据库中每个表都对应一个BLL类，为了达到解耦效果，BLL类不能直接调用其他表的DAL类，可以BLL类之间相互调用。

### DAL(Data Access Layer)数据访问层

提供数据访问的接口，没有任何逻辑。在接口中对数据库操作语句进行组合装配。DAL层一般关联Common类库中的最底层，最基础的数据库类（比如：链接数据库），必须关联Entity Class实体类项目。DAL层只是数据库的管理者，但不是访问者，不直接与数据库发生关联。数据库中每个表都对应一个DAL层的接口(访问控制)类。

### Common类库

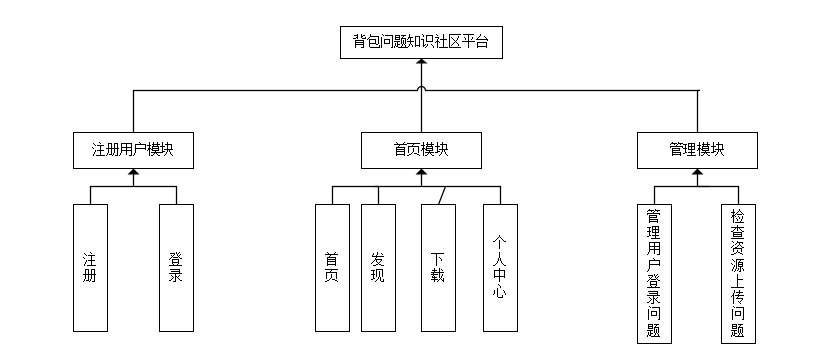
用于存放公用的类。最常用的就是数据库访问类，比如：链接字符串，数据库引擎类。它直接与数据库进行机械式的交换，无任何逻辑。

### Entity Class实体类

相当于加强的数据结构，实现了对数据的封装。数据库中每个表都对应一个实体类，表的字段就是实体类的属性，类型一一对应。UI、BLL、DAL这三层的交互主要就是通过实体类作为参数，并Return回信息。

## 系统结构设计及子系统划分

根据业务和功能，将系统的逻辑结构划分为用户登录、管理、以及主页内容模块，如下图所示：



各个子系统按照功能角度分解，划分出若干不同的功能模块，

## 系统功能模块详细设计

#### 用户管理模块

**模块描述：**管理系统用户，包括添加用户并赋予角色、修改用户资料和角色、删除用户。

**主要功能：**添加用户、修改用户、删除用户、列表和分页。

添加用户：

1. **流程图**

输入用户资料

选择角色

加密密码

填写合规

用户信息保存至表System\_admin\_info

验证必填项

验证用户名

是否存在

是否成功

分拆角色ID字符串

循环字符串数组

信息保存至表Dict\_admin\_vs\_roles

返回成功信息

是

否

是

返回失败信息

否

是

否

操作记录日志

1. **输入项**

用户ID string 必填

姓名 string 必填

用户密码 string 必填

确认密码 string 必填

联系电话 string

E-mail string

状态 char

用户角色ID string

1. **输出项**
2. 成功，UI提示添加用户完成；
3. 失败，UI提示具体信息。
4. **算法描述**
5. 前端js判断必填项的输入和二次密码是否相同，角色可以不选；
6. BLL须验证必填项，用户名是唯一的不能重复；
7. 加密密码使用MD5.Encrypt(string String, string Key)，Key = "XXXXXXXX"；
8. 用户信息保存至表System\_admin\_info，ExamSys.DAL.System\_admin\_info Add(ExamSys.Model.System\_admin\_info model) ；
9. 分拆角色ID字符串并循环字符串数组，信息保存至表Dict\_admin\_vs\_roles，ExamSys.BLL.Dict\_admin\_vs\_roles Add(ExamSys.Model.Dict\_admin\_vs\_roles model)。

列表中

点击删除按钮

是否成功

返回成功信息

是

返回失败信息

否

根据用户ID删除表Dict\_admin\_vs\_roles记录

是否成功

是

否

根据用户ID删除表System\_admin\_info记录

操作记录日志

## 软件重用方案

在本次软件的设计与实现的过程之中，主要从以下三个层次进行了重用，知识重用，方法和标准的重用以及软件成分的重用。

（1）知识重用是应用相同的知识设计规划、建设不同的软件产品。方法和标准的重用是比如本产品是利用面向对象方法设计又或者是采用国家标准局制定的软件开发文档规范。

（2）软件成分重用一般将划分为三个级别，即代码重用，设计结果重用和分析结果重用。而本次设计的软件产品的重用主要在源代码级，此次开发和设计软件产品采用了spring mvc框架，利用此框架可以快速搭建出一个程序的框架。

（3）此次我们的重用设计是建立在MVC框架之上。主要是继承类库中的构件要用到的基本功能的类。主要是一些界面元素，如菜单活框、列表框，这些构件在很多模块中都要用到，且处理逻辑大致相同，并进行扩充其处理逻辑。其次是构件在类库找不到相似的类，我们将从头创建自己的类，但为了将自己创建的类，如用户权限处理排序打印预览等，也纳入MVC的框架，我们自己创建的类都继承框架的一个抽象类，这样做的好处是把自己创建的类纳入Controller类的层次化管理，且这也是为实现虚拟函数和动态联编方便。我们也为每个件的功能和接口建立了文档，供应用开发中使用。这样我们就在MVC的框架下，建立了我们自己的一个重用构件库。

（4）使用重用明显会将开发效率提高一个层次，它不仅提高了开发效率，还保证了应用的风格和质量。特别是对新手较多的团队开发，采用重用的方法，让有经验的成员负责整个应用的框架，让新手使用重用来创建应用，这非常有利于提高效率，在保证质量的同时也为新手提供一个循序渐进的学习机会，有利于新手的成长。当然使用重用容易，但自己建立重用构件库也是需要代价的，是一个需时间积累的过程，也有一定风险。

## 4.5 设计关键类的重点服务

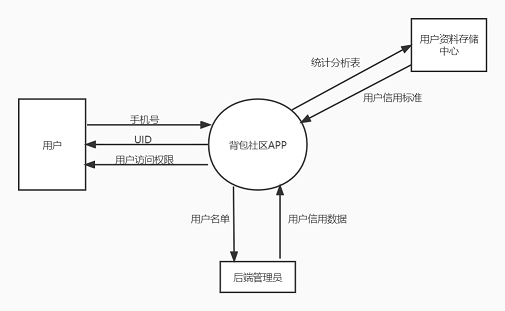
## （1）用户登录：因此次是背包知识社区平台，需要在每次访问该平台之间需要进行用户登录验证，以便更好的管理和查找

（2）权限管理：管理员对角色和用户进行权限信息的配置。以及相关的用户资源的上传以及审查。

# 数据库系统设计

## 数据库设计

### 顶层数据流图



### 一层数据流图

### 

### 二层数据流图

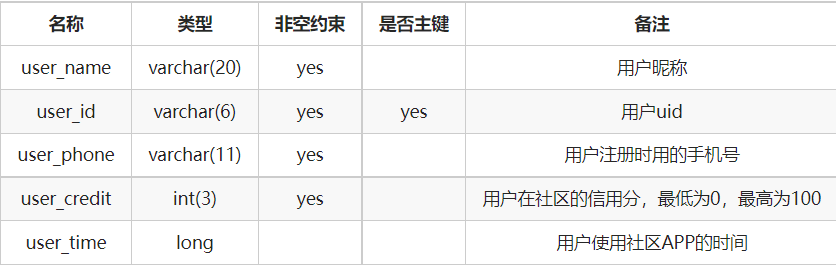
### 

### ER图

### 

### 数据字典

用户信息表



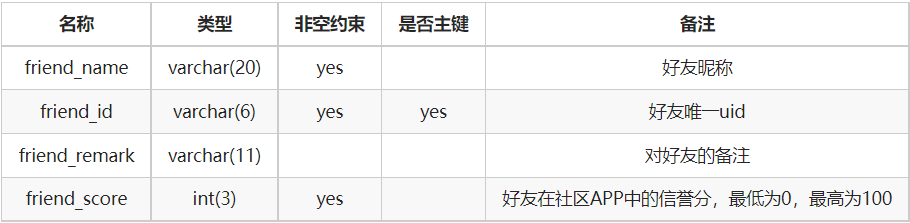
社区文章表



课程表

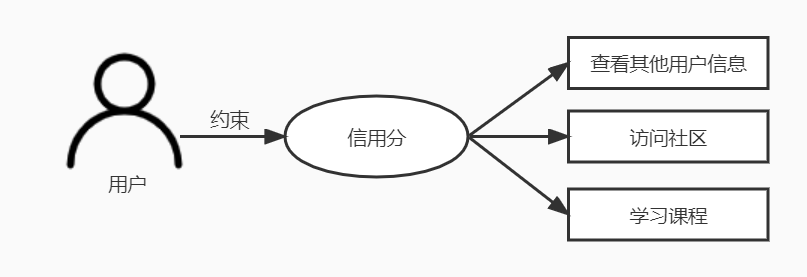


好友表

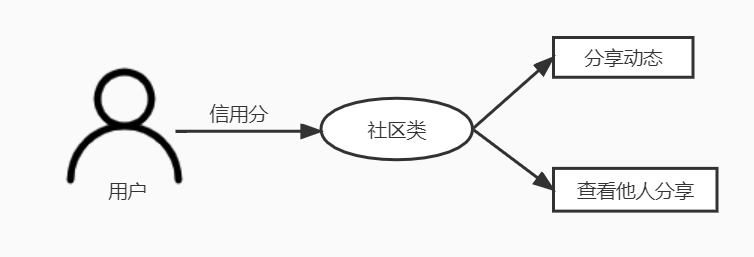


# 精化类的属性和操作，定义类中服务参数和具体实现逻辑

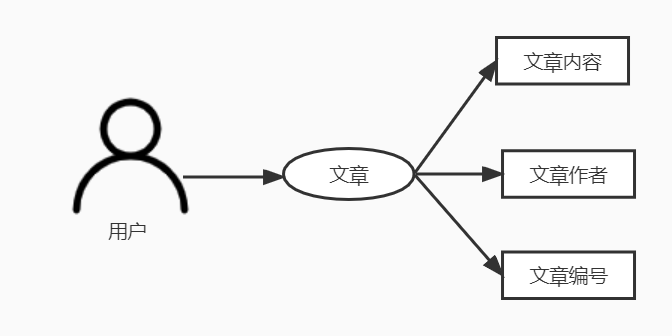
## 6.1 用户类：访问权限向较于管理员比较低，可查看其他用户的有限前端信息，根据社区信用分有限地访问社区文章的动态和访问课程。



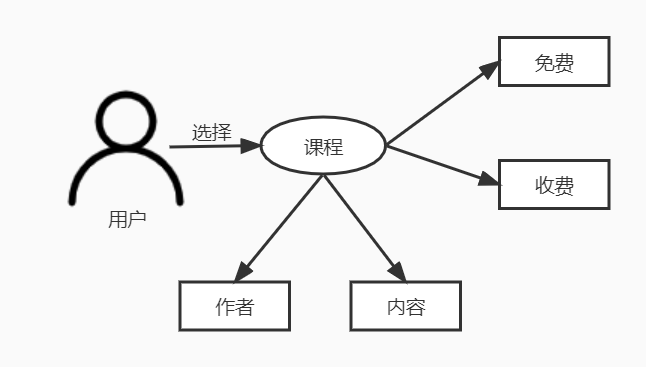
**6.2 社区类：**用户分享自己的动态，查看他人分享的文章。



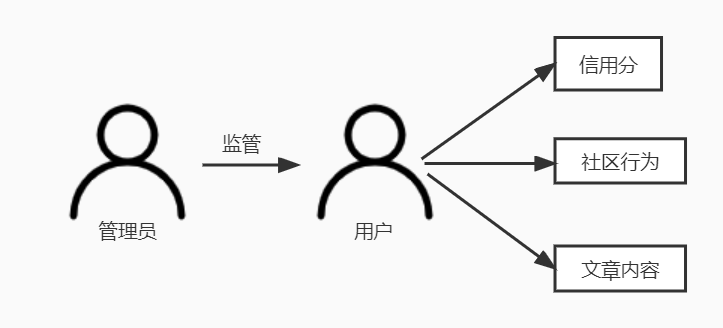
**6.3 文章类：**文章内容、文章编号、文章作者等。



**6.4 课程类：**用户选择自己习惯的课程进行他学习，其中有付费的课程，也有免费的课程。



**6.5 管理员类：**拥有app的全部权限，可管理所有已经注册在户的用户。



# 类的层次关系和关联关系

