**课程名称：Android移动开发技术**

项目名称：AgentDoo（代取快递的应用）

**指导教师： 杨卫明**

**学 院： 计算机与信息科学学院**

**专 业： 软件工程**

**学 号： 2016051604221**

**姓 名： 苏雪莲**

**成 绩：**

# 

# 功能需求

**1.1 背景**

当今社会中，科技发展迅速，网络购物已经在生活中普遍存在，然而取快递并不是特别方便，针对没有时间取快递的学生，设计了学生代取快递应用《AgentDoo》。这是一款独立开发的软件，该软件操作便捷，设计人性化，当有需要取快递的用户使用该软件时，只需将相关数据输入，由系统生成订单，等待接单。在订单被确认接收前发起人可以对订单进行修改或者删除订单，订单确认接收后用户可以追踪订单。系统会保存所有订单信息，方便用户在需要时查询历史订单。

**1.2 定义**

|  |  |
| --- | --- |
| 专业术语 | 定义 |
| 订单 | 一个有唯一编号的信息组，记录快递相关信息 |
| 发布者 | 发布订单的人 |
| 接收者 | 接收订单的人 |
| 订单池 | 订单发布后显示的地方 |
| 取货点 | 快递所在地 |
| 送货点 | 快递将被送往的地点 |

**1.3 用户特点**

该软件针对的是在校大学生，主要是因为某些原因不能及时取快递的学生和想赚取零用钱的学生，操作该软件的用户不限教育水平。当因为某种原因需要别人帮忙代取快递的时候就可以使用该软件。

**1.4 功能需求**

**1.4.1 账户管理**

实现用户帐号的常规管理，用户只有在账户登录的状态下使用该系统，用户成功登录后进入主界面，在用户管理中心可以对帐号进行操作。用户输入自己的账号和密码进行登录，请求登录，系统处理请求并返回结果。如果密码错误就返回重新登录，直到密码正确，成功登录系统后，用户可以对自己的账号进行管理，比如修改密码和修改昵称等，系统保存用户对帐号信息做的变更。

**1.4.2 订单管理**

实现客户端两个用户对订单的处理。用户1发布订单后，在订单被接收之前可以对订单进行相关操作，如修改订单或者删除订单，用户2可以进行关键字目的地和发送地的搜索，进行接收订单。

**1.4.3 信息查询**

该模块主要为订单信息查询和用户信息查询。订单信息查询指不论订单是否被人接收，发布者都可随时查询订单信息，以便追踪订单情况。用户信息查询是指不同用户之间可互相查看信用积分等信息，以便用户自行判断订单的真实性和接收者的可靠性。

**1.5非功能性需求**

需为学生提供一个安全的系统。首先，所有使用该系统的用户必须经过身份认证后方可使用系统。其次，为确保用户信息和财产安全，只有在订单被接收后才会将快递提取码、手机号等隐私单针对性的显示给接收者，在最大程度保证用户权益。并且在一个订单的开始到结束，用户都可追溯快递相关信息，保证发布者和接收者之间的联系。

**1.6 性能需求**

**1.6.1 时间性要求**

对于创建订单到发布订单成功的过程，需进行数据收集存储以及请求第三方平台等操作，所以该过程解题时间占总时间的最大部分，其次就是接收者接收订单的过程。具体时间特性要求如下

a、响应时间：50ms左右

b、更新处理时间；1~2s

c、数据的转换和传送时间；1～2s

**1.6.2 灵活性**

A.操作上的不同：可提供给用户一定UI的操作软件，并支持在后台独立运行。

B.主要运行环境是安卓手机，时间允许将开发适用于pc端的相应平台。

**1.6.3数据管理能力要求**

需要将用户信息以及订单存入数据库中，对数据库要求较高。

**1.7 支持**

Android操作系统，android studio软件上开发并测试

# 设计与实现

# **2.1 程序系统的结构**

订单信息

账号设置

用户注册

用户登录

账号管理

信息查询

修改订单

确认订单

发布订单

接收订单

订单管理

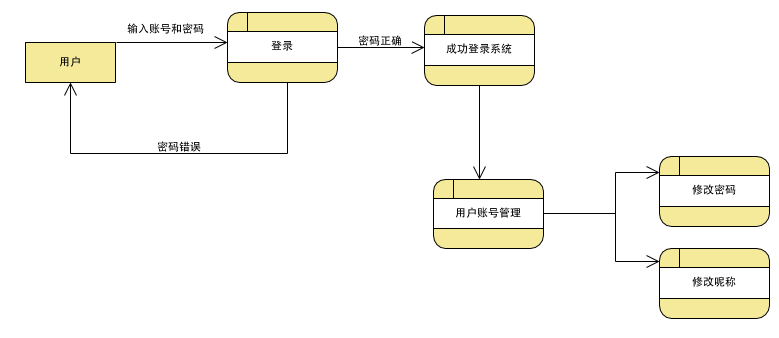
快递代取系统

接收人信息

**2.2 模块1（帐号管理）设计说明**

**2.2.1 模块描述**

实现用户帐号的常规管理，用户只有在账户登录的状态下使用该系统，用户成功登录后进入主界面，在用户个人界面可以对帐号进行更改操作。

**2.2.2 流程逻辑**

**2.2.3 输入项**

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 标识 | 数据类型 | 安全保密性 |
| 用户进入系统使用的用户名 | UserName | String | 无 |
| 用户登录密码 | Passworrd | String | 默认不可见,以\*的形式显示,用户可选择其可见性 |

**2.2.4 输出项**

系统首先检测用户是否存在，如果不存在，则提醒用户注册；若存在，系统然后检测帐号与密码是否匹配，如若不匹配，系统提示用户重新输入密码，匹配成功,则进入系统。

**2.2.5 接口**

帐号注册

Bomb数据库服务器

创建新用户

修改并保存

帐号设置

用户登录

查询访问

与本模块直接关系的数据：

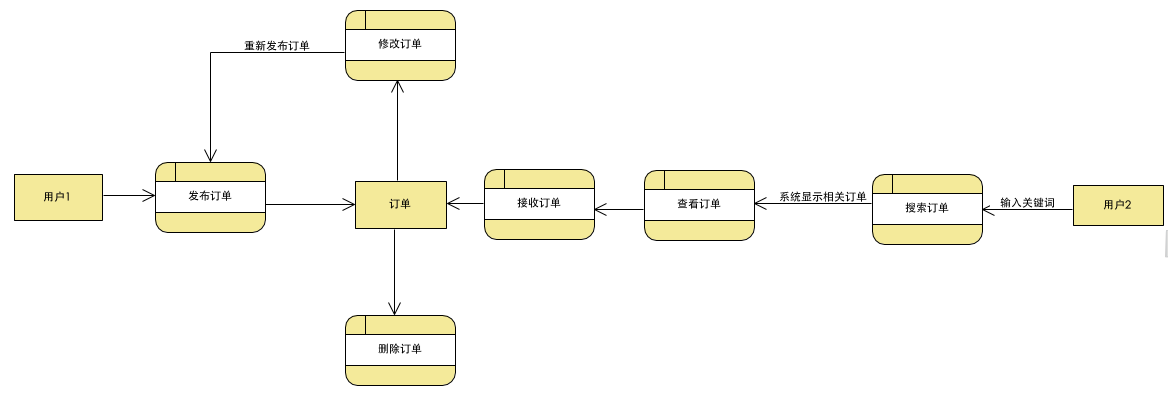
|  |  |  |  |
| --- | --- | --- | --- |
| 列名 | 数据类型 | 是否可为空 | 说明 |
| ObjectId | String | 否 | 用户id |
| UserName | String | 是 | 用户姓名 |
| Password | String | 否 | 用户密码 |
| Tel | String | 是 | 用户联系电话 |

**2.3 模块2（订单管理）设计说明**

**2.3.1模块描述**

实现客户端两个用户对订单的处理。用户1发布订单后，在订单被接收之前可以对订单进行相关操作，如修改订单或者删除订单，用户2可以进行关键字目的地和发送地的搜索，进行接收订单。

**2.3.2 流程逻辑**



**2.3.3 功能**

下单

修改订单

接单

显示该订单发布者姓名，电话和提货码

接收者

发布者

填写资料（取货地址，送达地址，发布人姓名，电话，取货码）

系统（存入数据库），并隐藏私密信息

订单池

**2.3.4 输入输出项**

* 用户1想要发布订单，则需要输入以下信息：

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 标识 | 数据类型 | 安全保密性 |
| 提货码 | take\_code | String | 保密，在订单被接收后私发给接收者 |
| 发布人姓名 | Name | String | 无 |
| 联系电话 | Tel | String | 保密，在订单被接收后私发给接收者 |
| 取货点 | addressBefore | String | 无 |
| 送货点 | addressAfter | String | 无 |
| 发布者帐号 | ObjectId | String | 无 |

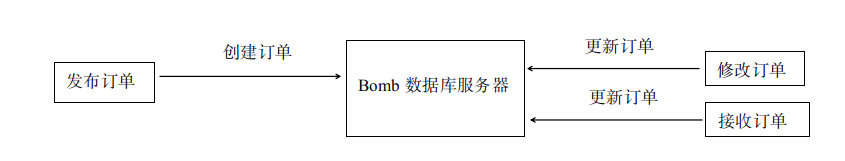
系统根据发布者输入的信息自动生成带有唯一订单号的订单，发布到订单池中。在订单被接收之前，发布者可以修改订单信息，保存后由系统更新并保存到数据库。

* 用户2想要接收订单，则需要输入以下信息：

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 标识 | 数据类型 | 安全保密性 |
| 姓名 | Name | String | 无 |
| 联系电话 | Tel | String | 保密，在订单被接收后私发给订单发布者 |

系统根据接收者输入的信息补充完善当前订单的接收者部分信息，并更新订单池当前订单信息。

**2.3.5 接口**



创建订单

更新订单

发布订单

Bomb数据库服务器

修改订单

接收订单

更新订单

创建订单

更新订单

发布订单

Bomb数据库服务器

修改订单

接收订单

更新订单

本模块主要涉及到的函数:

addOrder(Order order): 发布订单，将新订单添加到数据库

takeOrder():接收订单

editOrder():修改订单

deleteOrder():修改订单

getOrders(): 获得数据库中的订单记录

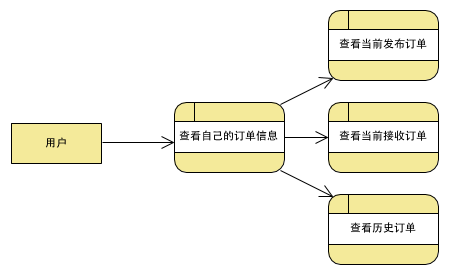
getOrder(ObjectId orderId): 根据唯一标识符订单号显示订单的具体信息

**2.4 模块3（信息查询）设计说明**

**2.4.1 模块描述**

该模块主要分为订单信息查询和用户信息查询。订单信息查询指不论订单是否被人接收，发布者都可随时查询订单信息，以便追踪订单情况。用户信息查询是指不同用户之间可互相查看信用积分等信息，以便用户自行判断订单的真实性和接收者的可靠性。

**2.4.2流程逻辑**



用户

用户信息查询

**2.4.3 功能**

通过订单追踪查询订单处理情况，能够较大层面的方便用户跟进并查看订单。

**2.4.4 输入输出项**

用户如果想要查询订单信息，则输入以下信息：

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 标识 | 数据类型 | 安全保密性 |
| 订单号 | Order\_num | String | 无 |

系统返回订单的详细信息

**2.4.5 接口**

Bomb数据库服务器

用户

查询访问

返回结果

# 用户界面设计

**3.1用户登录、注册**

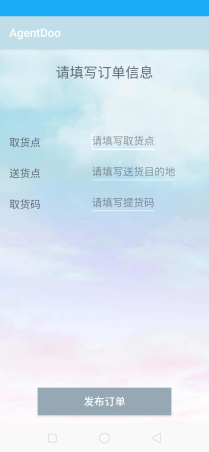
 说明：使用SharedPreferences将帐号保存到本机，方便下次使用

**3.2个人用户界面、修改账户信息**

首次登录需要完善信息 完整信息 修改信息

说明：自定义组件类RoundImageView extends ImageView 实现圆形头像

**3.3 订单**

列表信息 发布订单 修改订单 成功接收订单

说明：

1）OrderLab使用了单列模式，一旦创建，就会一直存在，直至整个应用进程被销毁。

2）使用工具栏菜单，提供应用导航（快捷发布订单按钮，个人主页按钮，显示/隐藏总订单数）

3）使用SwipeRefreshLayout实现下拉刷新

4）使用RecyclerView显示列表

5）使用RecyclerView.addItemDecoration(new DividerItemDecoration)对RecyclerView添加分割线

6）悬浮搜索按钮FloatingActionButton

7）ViewPager对详细信息进行显示

# 系统测试

**4.1测试内容及编号**

|  |  |
| --- | --- |
| 功能模块 | 测试内容 |
| A帐号管理 | 1在注册部分系统能否准确保存新增的用户信息；  2以及注册后系统对用户帐号的更改设置数据的更新；  3在登录部分系统对用户请求的处理。 |
| B订单管理 | 1系统的业务逻辑：   * 发布者必须填写订单全部信息才可以发布 * 用户不可以接收自己的订单 * 用户不可以更改别人的订单   2系统能否准确保存新订单；  3系统对修改的订单数据信息的更新。 |
| C信息查询 | 1通过订单号能否准确查找到订单详细信息；  2通过用户名能否准确查找对应的用户信息。 |

**4.2主要测试设计说明**

**4.2.1 A1帐号管理（注册）**

在注册部分系统能否准确保存新增的用户信息，并检测系统在注册部分能否做出合理的响应。

**1设计测试**

1）划分等价类

|  |  |  |  |
| --- | --- | --- | --- |
| 输入项 | 限制 | 有效等价类 | 无效等价类 |
| 用户名 | 由大于3位的字母组成且未被注册过 | a:大于3位字母且未被注册 | b:有非字母字符  c:小于等于3位字母  d：已被注册 |
| 密码 | 由大于6位的数字和字母组成 | m:大于6位纯数字  l:大于6位纯字母  n:大于6位字母和数字 | o：有非数字或者字母字符  p：小于6位字母和数字字符 |

1. 设计测试用例

|  |  |  |  |
| --- | --- | --- | --- |
|  | 测试数据 | 测试范围 | 期望 |
| 用户名 | ABCD（未被注册） | 等价类a | 有效 |
| qwer（已被注册） | 等价类d | 无效 |
| Re34 | 等价类b | 无效 |
| fek | 等价类c | 无效 |
| 密码 | 1234567 | 等价类m | 有效 |
| abcdefg | 等价类l | 有效 |
| 1234abc | 等价类n | 有效 |
| ！123456 | 等价类o | 无效 |
| 1234a | 等价类p | 无效 |

**2 输入输出**

|  |  |
| --- | --- |
| 输入 | 输出 |
| 用户名 | * 如果输入的用户名和密码格式不符合标准，系统提示用户重新输入； * 如果输入的用户名已注册，系统提示用户帐号已注册; * 如果输入的用户名和密码符合标准且未被注册且，则系统提示注册成功。 |
| 密码 |

**4.2.2 A2帐号管理（账户设置）**

注册登录后用户可以对自己的账号进行管理，比如修改密码和修改昵称等，系统对数据的更新。

**1 输入输出**

|  |  |
| --- | --- |
| 输入 | 输出 |
| 新的用户名或者密码 | 系统响应用户操作 |

**2 过程**

初始化：启动系统，用户登录后系统加载到个人信息界面

中间步骤：用户选择编辑信息，输入新的用户名或者密码，系统保存并更新数据库

运行结束方式：用户提交新的用户名或者密码，系统保存并返回提示信息

**4.2.3 A3帐号管理（登录）**

在登录部分系统对用户请求的处理。

**1输入输出**

|  |  |
| --- | --- |
| 输入 | 输出 |
| 用户名 | * 如果输入的用户名已注册且密码与之匹配，则进入系统，显示个人信息界； * 如果输入的用户名未注册，系统提示用户帐号未注册; * 如果输入的密码与注册的用户名不匹配，系统提示用户“密码错误，重新输入”。 |
| 密码 |

**2 过程**

初始化：启动系统，加载到登录界面

中间步骤：用户输入用户名及密码，系统根据用户输入数据判断并做出相应反应及提示

运行结束方式：登录成功，进入个人信息界面；登录失败，重新登录

**4.2.4 B1订单管理（业务逻辑）**

在业务逻辑部分实现客户端两个用户对订单的处理。用户1发布订单后，在订单被接收之前可以对订单进行相关操作，如修改订单或者删除订单，用户2可以进行关键字目的地和发送地的搜索，进行接收订单。

**1输入输出**

|  |  |  |
| --- | --- | --- |
|  | 输入 | 输出 |
| 用户1 | 发布相关信息（取货地址，送达地址，发布人姓名，电话，取货码等） | 带有唯一标识（订单号）的订单 |
| 用户2 | 接收者相关信息（姓名、联系电话等） | 含有接收者信息的完整订单 |

**2过程**

初始化：用户成功登录

中间步骤：用户1选择发布订单，输入生成订单的必要信息（取货地址，送达地址，发布人姓名，电话，取货码等），系统根据用户输入数据生成订单；用户2选择接收订单，系统在更新订单接收者的数据信息

运行结束方式：订单成功被处理

**4.2.5 B2订单管理（新订单）**

系统能否准确保存新增的订单信息。

**1 控制**

本部分的主要测试数据库是否存储了新的订单信息，人工输入，测试人员输入订单发布信息，然后记录每项输入数据库对新订单的保存情况。

**2 输入输出**

|  |  |
| --- | --- |
| 输入 | 输出 |
| 发布相关信息（取货地址，送达地址，发布人姓名，电话，取货码等） | 带有唯一标识（订单号）的订单 |

**3 过程**

初始化：用户成功登录

中间步骤：用户选择发布订单，输入生成订单的必要信息（取货地址，送达地址，发布人姓名，电话，取货码等），系统根据用户输入数据生成订单

运行结束方式：数据库成功保存新订单相关数据

**4.2.6 B3订单管理（更新订单）**

系统能否准确更新更改的订单信息。

**1 控制**

本部分的主要测试数据库是否更新了更改的订单信息，人工输入，测试人员接收订单输入订单接收信息或者发布者选择更改订单信息，记录操作后数据库对更改订单的更新情况。

**2 输入输出**

|  |  |
| --- | --- |
| 输入 | 输出 |
| 接收相关信息（接收者姓名，电话等） | 含有接收信息的完整订单 |

**3 过程**

初始化：用户成功登录

中间步骤：用户选择接收订单，输入接收相关信息（接收者姓名，电话等），系统根据用户输入数据更新当前订单信息

运行结束方式：数据库成功更新更改的订单的相关数据

**4.2.7 C信息查询**

在信息查询功能模块，用户输入的查询条件，系统对用户请求的处理。

**1 控制**

本部分的测试人工输入，测试人员输入想要查询订单的订单号，然后记录每项不同输入系统所做出的反应。

**2 输入输出**

|  |  |
| --- | --- |
| 输入 | 输出 |
| 订单号 | * 如果输入的订单号存在，系统显示当前订单的详细信息 * 如果输入的订单号不存在，系统提醒用户“订单号不存在，重新输入” |

**3 过程**

初始化：启动系统，用户成功登录

中间步骤：用户选择信息查询，用户输入订单号或者用户名，系统根据用户输入数据判断并做出相应反应及提示

运行结束方式：查询成功（存在则显示详细信息，不存在输出提示信息）

# 项目总结

**5.1试运行工作评价**

1. 订单需要更多的细节，如发布的时间，订单的佣金。
2. 运行后想让应用更人性化，提供通过订单的发货点、送货点、佣金等条目搜索符合条件的订单。
3. 有时初始化进入应用后订单池为空，加载订单失败。

**5.2技术积累总结与评价**

1. 软件的实现严格按照需求说明文档，先设计后实现来完成的。需求由于进行了比较严格的分析和策划，所以后期实现相对而言，改动较少，提高了开发效率。
2. 该项目使用AndroidStudio集成开发环境，AS使用方便快捷，代码的提示及补全功能给开发带来了便利；但是AS的错误不好找，错误提示无法快速找到具体错误原因。
3. 使用Bmob后端云对数据进行存储，Bmob后端云操作简单便捷。
4. 熟悉了Android的组件，如：

* 使用工具栏菜单，提供应用导航；
* 使用SwipeRefreshLayout实现下拉刷新；使用RecyclerView显示列表，并使用RecyclerView.addItemDecoration(new DividerItemDecoration)对RecyclerView添加分割线；
* 悬浮搜索按钮FloatingActionButton；
* ViewPager对详细信息进行显示。

1. 订单池采用单例模式，OrderLab一旦被创建，就一直会存在直至整个应用被销毁。
2. 采用MVC（模型-视图-控制器）设计模式，可以按层而非一个个类来考虑设计开发，有助于设计和理解应用；同时MVC设计模式还便于复用类，相比功能多而全的类，功能单一的专用类更有利于代码复用。