软件工程实验报告

项目名称：Android 记账 APP

项目组成员：

姓名	学号	角色	工作内容	评分
付广庆	2018081309005	组长/项目经理	编码、集成、调试、报告撰写	A
卢鹏宇	2018081309013	技术经理	编码、报告撰写	B
李泽勤	2018081309012	产品经理	UI、报告撰写	C
黄戈里	2018081309001	组员	需求分析、报告撰写	D
解博文	2018081309008	组员	报告撰写、测试	E
宋依航	2018081309015	组员	报告撰写、测试	E

注：评分共分为 5 档：A、B、C、D、E，每组中评分须拉开差距，要求每个档次都有。

项 目 编 号	2018081309
文 档 编 号	1
密　　级	内部

Android 记账 APP 需求规格

V1.0

第 5 组

评　　审　　日　期： 2020 年 10 月 17 日

目录

错误!未指定书签。

简介

为了实现用户的随时随地的记账等需求，开发人员开发一款名为“贴身会计”的记账 APP，该软件适用于安卓手机，为广大用户提供可视化记账、财产管理的服务。编写此软件需求规约用以对该软件的开发及使用做出详细阐述及说明。

目的

贴身会计是一个独立的记账系统，区别于简单的账簿，该软件可以在本地注册登录后详细记录每一笔收入与支出，并具有输出消费报表、显示每一笔收入与支出和阅读财经新闻等功能，同时还具有计算器、扫一扫这样的实用小工具。编写此软件需求规约是为了给用户及开发人员阐明软件的详细情况，起参考及指导作用。

范围

此软件需求规约适用于贴身会计记账 APP 及其各项功能的使用与开发。

定义、首字母缩写词和缩略语

无。

参考资料

[1]《需求规格报告格式标准》 V1.1

概述

此软件需求规约由简介、整体说明、具体需求、支持信息四个部分构成。其中具体需求部分为主体部分，用以阐明贴身会计记账系统的具体功能及其可用性、可靠性、性能、可支持性、设计约束等方面的信息。

整体说明

随着各种线上支付应用越来越广泛，人们消费越来越快捷、方便，这极大刺激了人们的消费欲望。在这样快消费的时代，人们对于财产的管理需要进一步完善，这直接导致了对相比于原始记账方式更为便捷的线上记账方式的需求大大增加。贴身会计记账系统的开发，既是为了方便消费者的随时随地的记账需求，也是为了帮助消费者更好地进行财产管理以及消费结构的优化。替身会计记账 APP 适用于广大对于快捷记账方式有需求的用户，也适用于希望更好地管理收入与支出的学生以及上班族这样的用户类型。使用该软件只需一部安卓系统的手机即可，使得可视化财产管理从此如影随形，顺心自如。

用户的特点：

本项目主要面向的用户为：

1. 对生活消费有记账习惯的用户

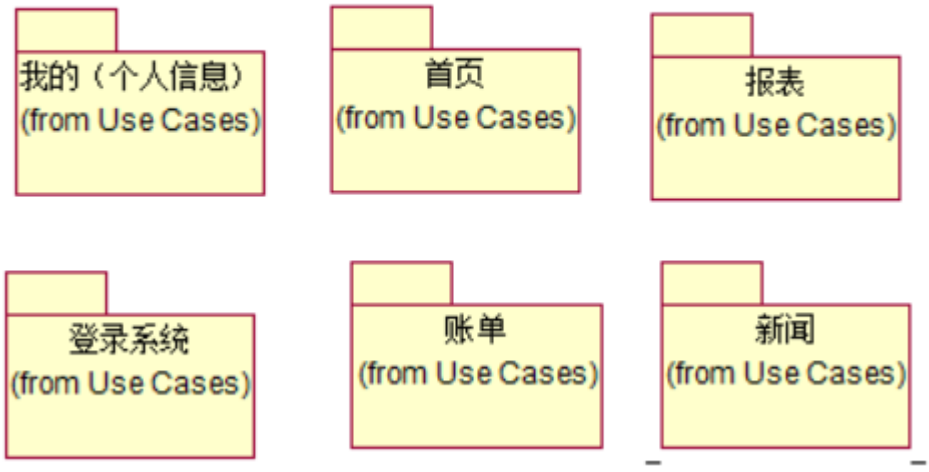
2. 想要改变自身消费习惯的用户

“贴身会计”的设计简洁明了，项目齐全，功能完备，容易上手，特别适合有记账习惯的用户长期使用。为了关照一些不习惯手机使用的中老年用户，本产品在小程序的设计上也别出心裁：精简的图标、加大的字体、简单的操作，使得即使是不常用手机的中老年人使用起来也得心应手。另外，显示账单、输出报表的功能，也能让用户对自己最近的开支了然于胸，从而对未来的消费进行自己的规划。当然，此软件附带的计算器功能也是一大特色，它能帮助用户更好地计算收入与开支，让用户在记账时更加准确严谨。除此之外，此软件提供的阅读财经新闻的功能，让用户能随时随地收到最新的财经消息报导，对于不管是对财经感兴趣的用户或者想要获取财经知识的用户来说，都是一个十分方便的功能。当然，此产品对于那些想要改变自身消费习惯的用户来说，“贴身会计”也是不二的选择。由于具有显示账单和输出报表的功能，用户能够清晰的看到自己最近的消费记录、自己在各个方面的消费金额占总消费的比例等，方便直白地了解自己近期的消费状况，从而直观地了解自己在哪个方面的消费投入过多，进而做出相应的调整，改变自己在某些方面花钱大手大脚的习惯。当然，对财经新闻功能的合理使用，也能了解到国内和国际市场的变化，提前预知一些产品的涨价或是降价，从而方便用户做出更好的消费选择。

具体需求

软件需满足本地注册登录、记账、输出账单、生成报表、可阅读财经新闻、修改个人信息的功能。

本地注册登陆后显示主界面：



如上图所示，主界面分有五个子模块，分别是：首页、账单、报表、新闻、我的

首页：首页可看到今日账单，有“记一笔”按钮,记一笔后会实时更新今日账单；

账单：账单界面可以查看每月所有账单，并可选择月份；

报表：报表界面可以饼状图显示每月支出各类型占比，帮助用户更好规划消费；

新闻：该界面连接网络服务，显示当日最新财经新闻；

我的：该界面含有用户个人信息及实用小工具：简易计算器、扫一扫、房贷计算器，其中个人信息包含用户的用户名及用户各个账户的余额（可以在此页面进行修改），账户包括：微信、支付宝、银行卡、校园卡。

功能

1、登录&注册功能：

进入该界面，默认是让用户进行登录操作；

若用户之前没有注册过，则会提示让用户进行注册；

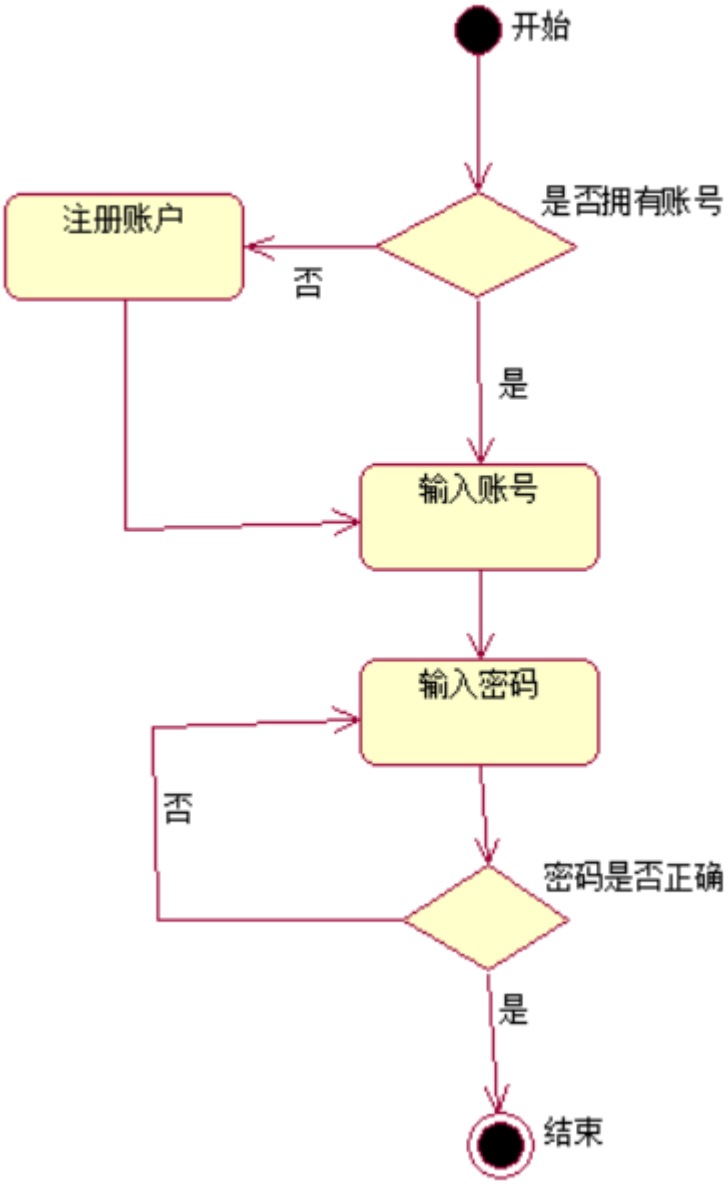
若用户忘记登录密码，也有重置密码的功能；

若用户少填或漏填某项，系统会给出相应提示；

首次注册及登录之后再次打开 APP 不需要重新登陆。

总体流程为:首次登录注册->填写注册信息->注册验证->验证通过->系统跳转回登陆界面->登录

已登陆账号->系统直接跳转到首页



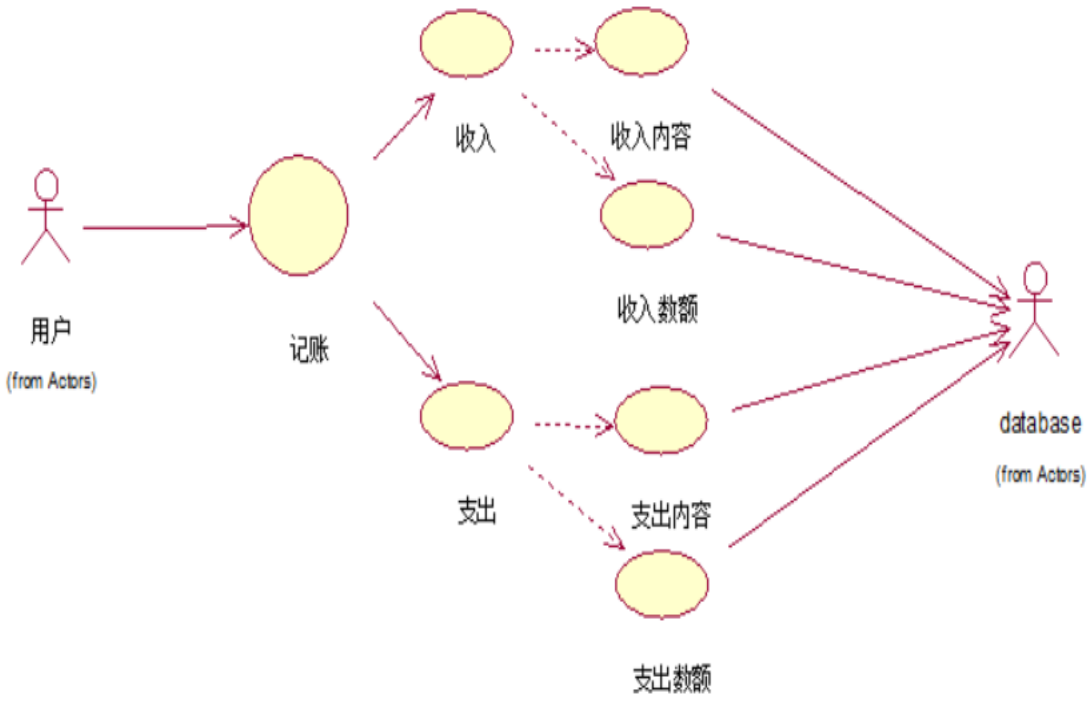
2、记账功能：

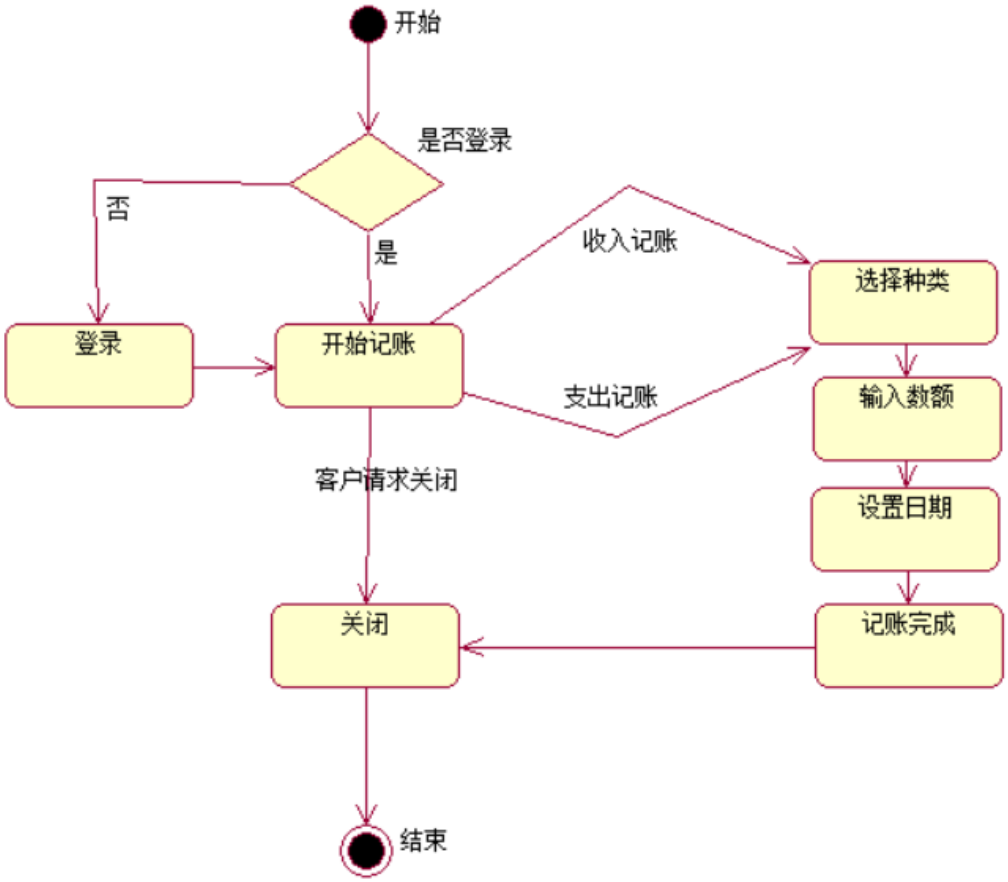
在首页页签上会显示【记一笔】按钮，点击该按钮，会让用户选择记录的类型，是收入或支出，金额是多少，该笔消费或收入是来自或收入用户的哪个账户（银行卡、支付宝、微信、校园卡），消费或收入的类型，其中支出包括：餐饮、购物、交通、日用、娱乐、通讯、人情、旅行、医疗、其它共八个模块，收入包括：薪资、奖金、借款、投资所得、投资回收、意外所得、人情、其他共八个模块；

根据用户选择的消费或收入类型，系统呈现并储存不同的数据内容；

当用户录入好收支明细，会在该页面显示今日账单。

总体流程：点击【记一笔】按钮->选择该笔账单为收入或支出->选择该笔账单消费或收入账户->输入该笔账单消费或收入金额->选择日期并输入备注信息->点击确定->系统跳转回首页并在首页显示今日所有账单





3、显示账单及输出报表功能

该功能分为两个页签：明细、类别报表；

这两个页签内容，都根据月份来统计呈现；

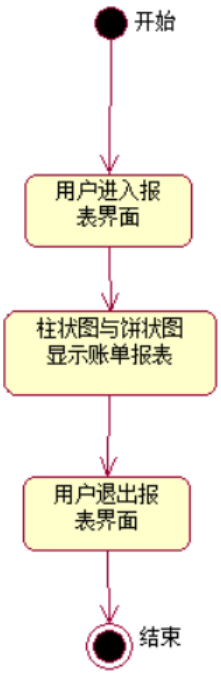
并根据月份，显示当月的收入与支出。

类别报表页签：

该页签会根据用户的收支明细的类别来进行统计，呈现方式是圆形饼图。

输出账单总体流程：点击【账单】按钮跳转到账单界面->选择要查询的年份->选择要查询的月份->系统显示当月所有支出与收入的明细

生成报表总体流程：点击【报表】按钮跳转到报表界面->选择要生成报表的年份->选择月份->系统以圆饼图的形式生成当月支出与收入的占比情况



4、阅读财经新闻功能：

打开该页面可看到当日最新的财经新闻，帮助用户更好地进行财产管理。

总体流程：点击【新闻】按钮->跳转到新闻界面->系统显示财经网新闻

5、管理账户余额功能：

在【我的】界面系统会显示用户的账户余额并随着支出与收入变化，用户可以随时查看每个账户余额并在需要时进行修改，账户包括：支付宝、微信、银行卡、校园卡。

总体流程：点击【我的】按钮->系统跳转到我的信息界面->系统显示各个账户余额->选择并点击要修改金额的账户->输入金额->确定->系统储存信息并显示新的账户余额

6、退出账户功能：

退出账户后需重新登陆或登录其他账户。

总体流程：点击【我的】按钮->点击【退出登录按钮】->系统跳转至登陆界面

〈功能性需求一〉

无。

可用性

贴身会计记账 APP 适用于安卓系统，对于无论是普通用户还是高级用户来说，都易于操作，并不需要额外培训，阅读软件功能说明即可。

一般用户按照软件提示或帮助即可完成注册、记账、输出账单、生成报表等业务；

用户可随时通过“记一笔”更新自己的账单及报表。

〈可用性需求一〉

对于贴身会计记账 APP 的使用只需要一个成熟的安卓系统环境，能兼容其他的安卓系统，具有很明显的普适性。

可靠性

贴身会计记账 APP 可靠性较强；

支持 7*24 小时的服务；

平均故障时间为 1 个月；

平均修复时间为 1 天；

需定期进行系统维护。

〈可靠性需求一〉

无。

性能

页面响应时间应该在 3 秒以内，最长不超过 6 秒；

可以至少容纳 6000 条数据。

<可靠性需求一>

无。

可支持性

对于贴身会计记账 APP 的维护及支持服务由开发人员提供。

<可靠性需求一>

无。

设计约束

对于贴身会计记账 APP 的开发，运用的软件及语言支持为：

Java 运行环境：JDK

Android 开发及运行工具：Android Studio：java 语言，用以对软件界面及功能的开发；

Android 运行模拟器：天天模拟器

SQLite：sql 语言，用以对用户数据得储存及管理；

Photoshop 用以制作界面图片及图标等。

<可靠性需求一>

无。

联机用户文档和帮助系统需求

无。

购买的构件

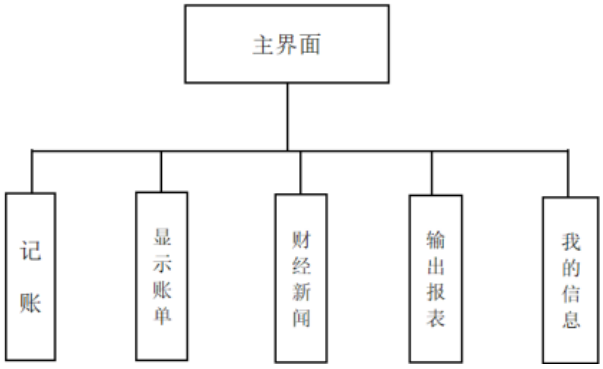
无。

接口

用户界面

贴身会计记账 APP 用户界面具有登陆注册、记账（日期、账户、支出或收入、类型...），显示全部账单，生成报表的功能。同时具有实用小工具，包括：计算机、扫一扫、房贷计算器。

界面结构图：



硬件接口

无。

软件接口

无。

通信接口

无。

许可需求

无。

法律、版权及其他声明

无。

适用的标准

无。

支持信息

无。

项 目 编 号	**200602006**
文 档 编 号	**11**
密 级	内部

Android 记账 APP 概要设计

V1.0

第五组

评 审 日 期： 2020 年 10 月 26 日

目 录

1.导言

1.1 目的

该文档的目的是描述网上招聘系统项目的概要设计，其主要内容包括：

- 系统功能简介
- 系统结构设计
- 系统接口设计
- 数据设计
- 模块设计
- 界面设计

本文档的预期的读者是：

- 开发人员
- 项目管理人员
- 测试人员

1.2 范围

该文档定义了系统的结构和单元接口，但未确定单元的实现方法，这部分内容将在详细设计/实现中确定。

1.3 缩写说明

UML

Unified Modeling Language（统一建模语言）的缩写，是一个标准的建模语言。

AS

Android Studio（Android 开发软件）的缩写。

JSP

Java Server Page（Java 服务器页面）的缩写，一个脚本化的语言。

MVC

Ｍｏｄｅｌ－Ｖｉｅｗ－Ｃｏｎｔｒｏｌ（模式－视图－控制）的缩写，表示一个三层的结构体系。

1.4 术语定义

无

1.5 引用标准

[1]《企业文档格式标准》
北京长江软件有限公司
[2]《软件概要设计报告格式标准》
北京长江软件有限公司软件工程过程化组织

1.6 参考资料

[1]《Android Studio 开发实战——从零基础到 APP 开发》〔中〕欧阳燊
清华大学出版社

1.7 版本更新信息

本文档的更新记录如表 B － 1 所示。

表 B-1 版本更新记录

修改编号	修改日期	修改后版本	修改位置	修改内容概述
000	2020.10.26	0. 1	全部	初始发布版本
001	2020.10.30	1.0	全部	完善具体内容

2.软件分析

本软件可以实现手动记账的功能，用户通过手动输入资金的流动方式、账户、用途、时间等信息，记录账单。同时，用户可以逐月查看账单，也可以查看每个月或每年的报表饼状图。此外，用户还可以在软件内实时访问财经网的内容，获取最新财经信息。

用户还能在我的信息界面查看各个账户的余额。

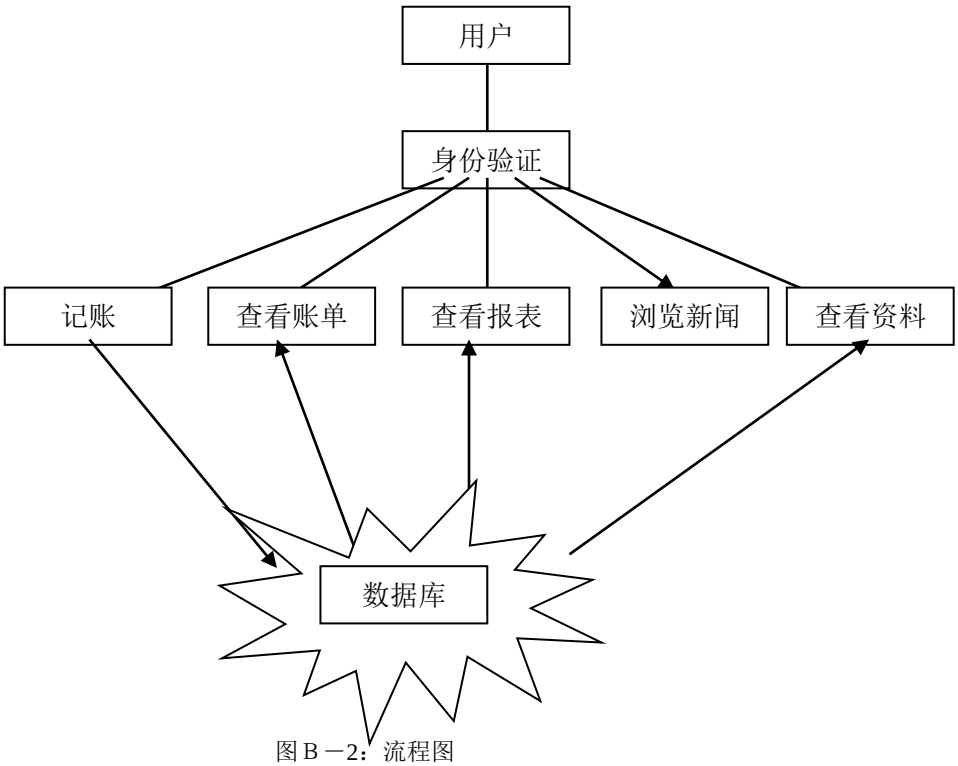


图 B －2：流程图

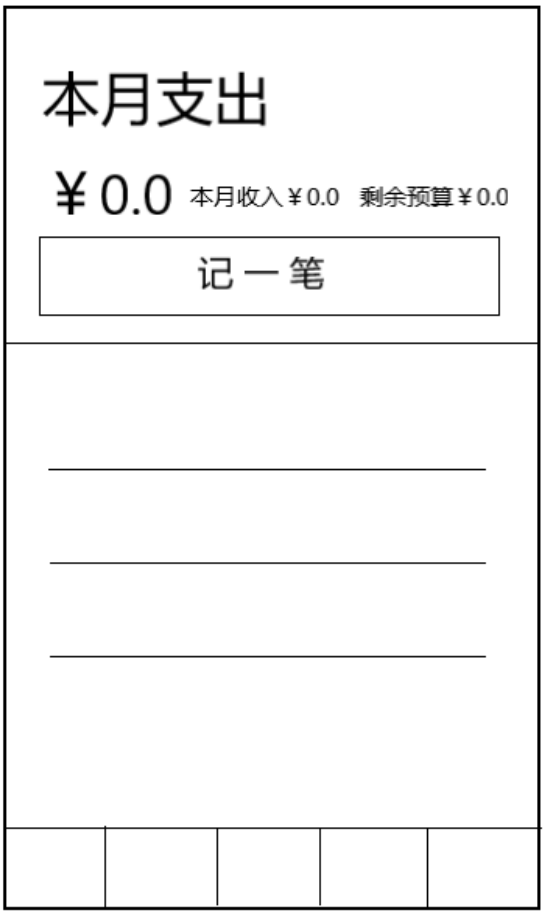
3.界面设计

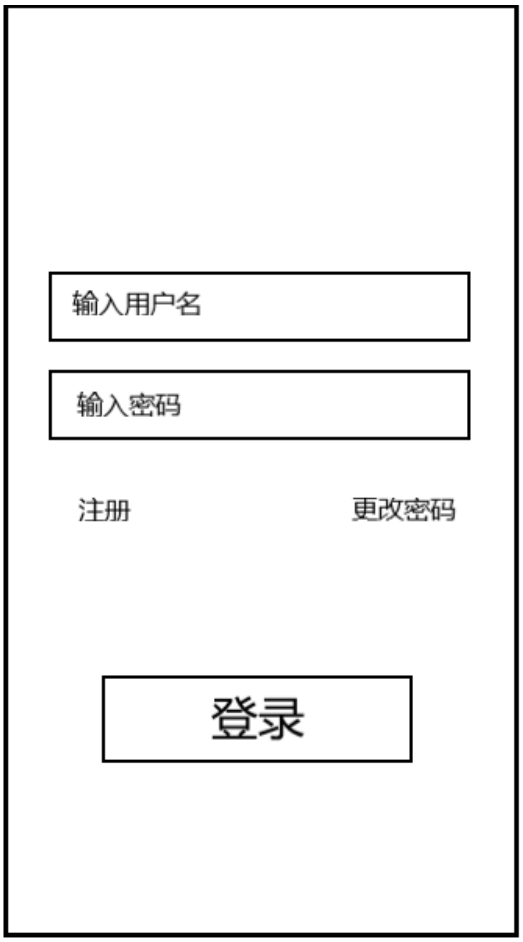
本软件为单机软件，只显示客户端界面，实现本地登录注册、记录账目、查看报表、查看账单、浏览新闻、查看资产等功能主要界面设计如下：

- 登录界面
  - ◆ 通过用户名和密码实现用户登录。
- 注册界面
  - ◆ 用户输入用户名、密码和手机号进行记录。
- 改密界面
  - ◆ 用户通过输入用户名和手机号来更改密码。
- 软件主页
  - ◆ 包括五个导航栏，分别对应记账界面、账单界面、报表界面、新闻界面和个人信息界面。
- 记账
  - ◆ 包括“今日账单”、“本月支出”、“本月收入”、“本月预算”、“编辑账目信息”等页面。
- 查看账单
  - ◆ 账单界面，按照选择的月份与年份显示对应的账目。
- 查看报表
  - ◆ 报表界面，按照选择的月份与年份显示对应的饼图。。
- 浏览新闻
  - ◆ 展示新闻页面。

- 浏览新闻
  - ◆ 展示个人信息页面。

界面设计图如下：





具体页面流如下图Ｂ－３所示：

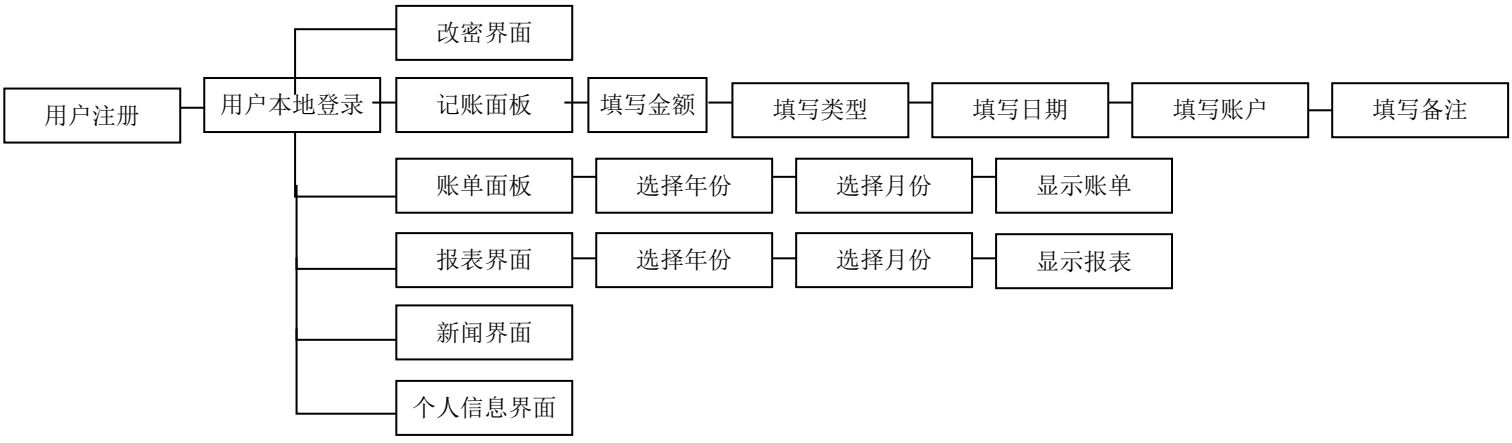


图 B －3：客户端的页面流程

4.体系结构

系统的总体结构设计遵循如下原则：

1 ）系统应具有良好的适应性：能适应用户对系统的软件环境、管理内容、模式和界面的要求；

2 ）系统应具有可靠性：采用成熟的技术方法和软件开发平台，以保证在以后的实际应用中安全、可靠；

3 ）系统应具有较好的安全性：应提高完善的安全机制和用户权限限制机制，确保数据的受限访问；

4 ）系统应具有良好的可维护性：系统应易于维护、安装；

5 ）系统应具有良好的可扩展性：系统应适应未来信息化建设的要求，能方便得进行功能扩展，以建立完善的信息集成管理体系。

本软件采用 Android 体系结构，基于模型 (Model) －视图 (View) －控制器 (Controller)(MVC) 模式。

4.1 体系结构

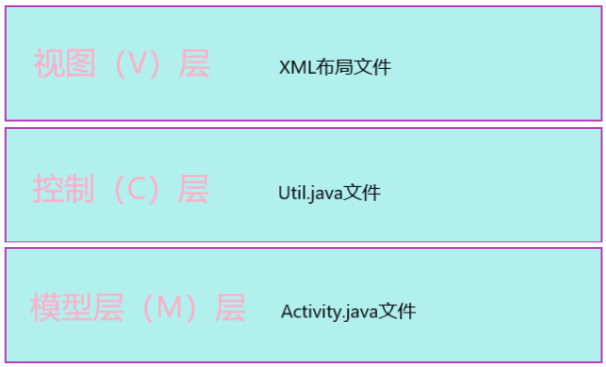
在 Android 里 MVC 模式体现在:

模型层（Model）：对数据库的操作、对网络等的操作都应该在 Model 里面处理，当然对业务计算等操作也是必须放在的该层的。

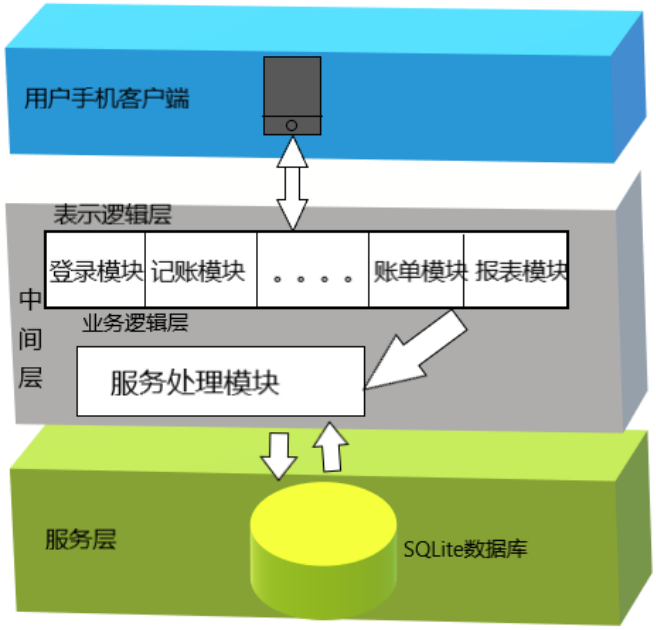
视图层（View）：一般采用 XML 文件进行界面的描述，使用的时候可以非常方便的引入. 当然，在 Android 中也可以使用 JavaScript+HTML 等的方式作为 View 层，当然这里需要进行 Java 和 JavaScript 之间的通信， Android 提供了它们之间非常方便的通信实现。

控制层（Controller）：Android 的控制层的任务主要是由众多的 Acitvity 的实现。

系统体系结构如图：



体系结构的具体拓扑图示如图：



4.2 系统运行环境

Android4.0 及以上版本。

4.2.1 硬件环境

本软件的硬件环境如下：

- 客户机：Android 手机
  - CPU：P4 1.8GHz 以上
  - 内存：256MB 以上
  - 能够运行 Android 4.0 以上版本的机器
  - 分辨率：推荐最低使用 540*960 像素

4.2.2 软件环境

本软件的的软件环境如下：

- 操作系统：Android 4.0 或以上版本
- 数据库：SQLite
- 开发工具包：JDK Version 11.4.2
- 开发环境：AS

（1）数据库及操作系统:

对于核心数据库来说，选择一个合适的数据库系统对我们的系统运行是很重要的，选择数据库的关键因素是要考虑预计会有多少人同时访问数据库；正常工作时间的级别；用来访问数据库的应用程序的类型；运行数据库的服务器的硬件和操作系统类型；以及管理人员的专业技术水平。目前市场上适用于中小型企业的数据库产品有 IBM DB2 、Microsoft SQL Server 系列、 Oracle 系列。所有这些产品都基于 SQL 语言。同时，它们还拥有精密复杂的安全控制以适应不同的商业需要。服务器操作系统使用 Windows 2000 Server 系统。

考虑到价格因素、易用性，我们使用 SQL SERVER 2000 作为系统后台数据库系统，服务器操作系统采用 Windows 2000 Server。

（2）WEB 服务软件：

目前的 WEB 服务器软件有很多种，成熟而且稳定有 Apache、Ｔｏｍｃａｔ和 Microsoft 的 IIS ，它们也是占据着 Web 服务器市场最大的份额。Tomcat 是 Sun 和 Apache 合作做出来的 JSPServer,支持 Servlet2.2 及 JSP1.1 等版本。而且 Tomcat 未来将会取代 Jserv，成为 Apache 主要的 Servlet&JSPEngine。Tomcat 在设计上是以独立的 Server 执行,而不像 Jserv 是附在 Apache 中,这样就更可以发挥在 servlet 中，非 HttpServlet 的能力。Tomcat 是 Java 程序，所以只要有 JDK 就可以使用,不需要考虑操作系统平台。因此选择Ｔｏｍｃａｔ作为 WEB 服务器。

5.数据模型

本系统的数据模型主要是进行数据库的设计。

5.1 数据库的概念结构模型设计

概念设计以反映现实世界中的实体、属性和它们之间的关系等的原始数据形式，建立数据库的每一幅用户视图。图Ｂ－5 是系统 E-R 图。其中软件中的数据库包含账目信息，账户信息。账目信息包括日期、金额、账户、类型、备注、支出/收入等信息；账户信息包括用户名、密码、手机号、支付宝余额、银行卡余额、校园卡余额、微信钱包余额。

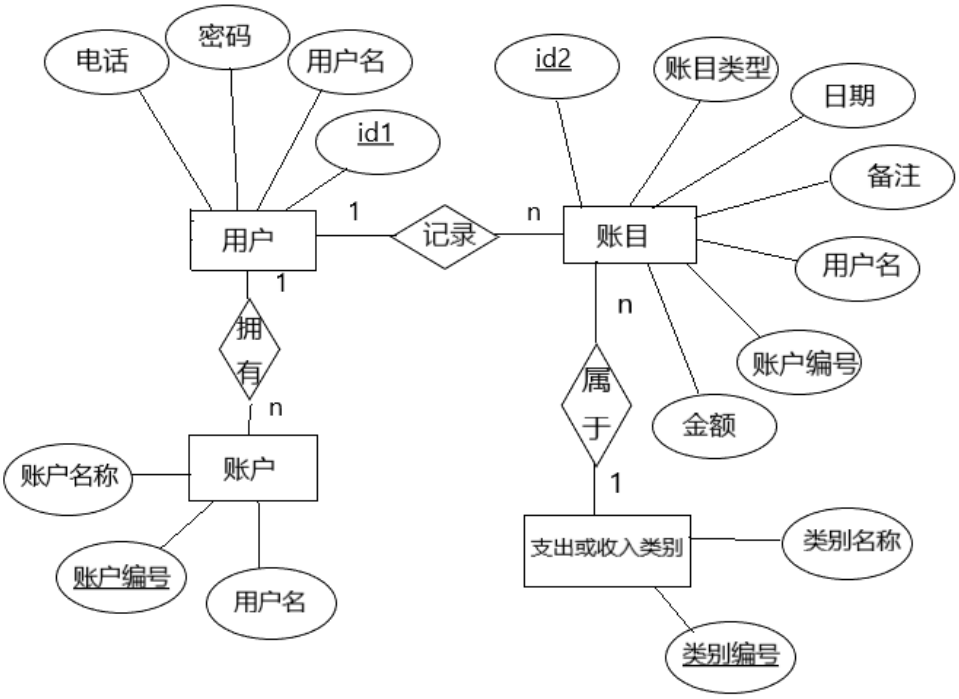


图 B －5：软件的 E － R 图

5.2 数据库的逻辑结构模型设计

数据库的逻辑设计是将各局部的 E-R 图进行分解、合并后重新组织起来形成数据库全局逻辑结构，包括所确定的关键字和属性、重新确定的记录结构、所建立的各个数据之间的相互关系。根据本系统需求分析,系统的数据库包括了用户管理、账目管理以及需要的基本数据字典等部分。

用户管理包括的库表：

- user-已经注册的用户信息

账目管理包括的库表：

- data－账目信息

表 B － 2 是对表 data 的设计，其它库表的设计详细见数据表设计文件。

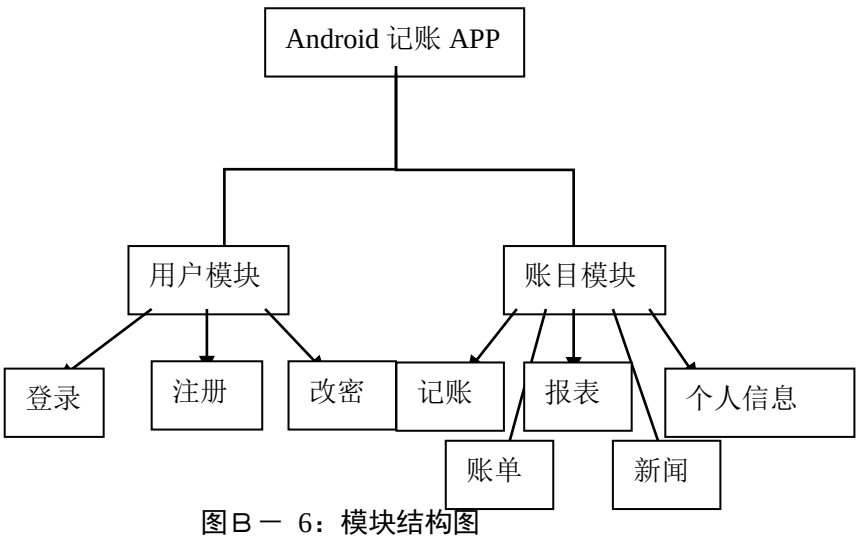
表 B － 2：data（账目信息）表

字段名	字段代码	字段类型	关键字	可否为空	代码字典表

类别	Category	TEXT		N	
账目编号	Id_1	INT	Y	N	
记账用户名	userID	TEXT		N	
记账日期	makeDate	Date		N	
支出/收入	Type	INT		N	
账户	way	TEXT		Y	
备注信息	note	TEXT		Y	
金额	cost	Money		N	

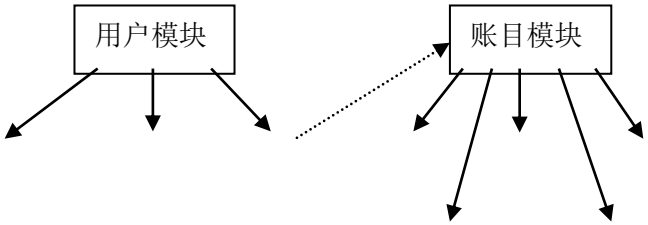
6.模块设计

按照数据分解，本系统分为用户模块和记账模块。根据页面流的设计，记账模块又分为记账、账目、报表、新闻、个人信息五个界面。用户模块分为注册、登录和改密三个界面。如图B－6。



图B－ 6：模块结构图

各个部分的关系如图B－ 7 如下：



以下将分小节对各个部分进行设计

6.1 用户模块设计

软件在初次使用时需要用户进行账号注册，用以区分不同的记账者。需输入用户名、密码以及手机号码。登录过后可以手动退出，若未手动推出则下次启动默认为登录状态。在登录界面可以选择注册新账号或更改密码。

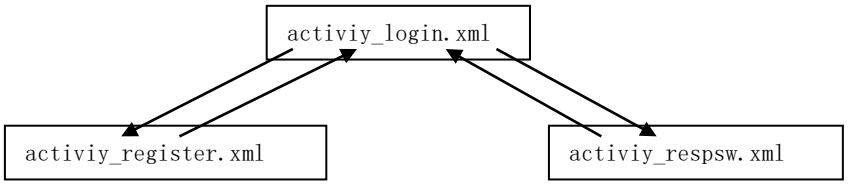
6.1.1 表示层设计

根据上述的功能介绍，总结出客户端的页面设计如表Ｂ－３。

表Ｂ－３：用户模块的页面设计

界面	XML	功能描述
登录界面	activiy_login.xml	登录页面
注册界面	activiy_register.xml	注册页面
改密界面	activiy_respsw.xml	更改密码的页面

根据界面流的设计可以确定各个界面的访问入口，以及界面之间切换关系，流程图如图Ｂ－８所示。



6.1.2 控制层

表Ｂ－５：登录模块的组件映射

事件	组件名	入口	动作	出口
输入登录用户名	et_user_name	LoginActivity.java	Edit	LoginActivity.java
输入登录密码	et_psw	LoginActivity.java	Edit	LoginActivity.java
登录	btn_login	LoginActivity.java	OnClick	MainActivity.java
跳转至注册界面	tv_register	LoginActivity.java	OnClick	RegisterActivity.java
跳转至改密界面	tv_find_psw	LoginActivity.java	OnClick	ResetPswActivity.java
输入注册用户名	et_user_name	RegisterActivity.java	Edit	RegisterActivity.java
输入注册密码	et_psw	RegisterActivity.java	Edit	RegisterActivity.java

再输入注册密码	et_psw_again	RegisterActivity.java	Edit	RegisterActivity.java
输入注册手机号	et_phoneNum	RegisterActivity.java	Edit	RegisterActivity.java
注册	btn_register	RegisterActivity.java	OnClick	LoginActivity.java
输入改密用户名	et_user_name_res	ResetPswActivity.java	Edit	ResetPswActivity.java
输入对应手机号	et_phoneNum_res	ResetPswActivity.java	Edit	ResetPswActivity.java
输入新密码	et_psw_res	ResetPswActivity.java	Edit	ResetPswActivity.java
再输入新密码	et_psw_again_res	ResetPswActivity.java	edit	ResetPswActivity.java
改密	btn_respsw	ResetPswActivity.java	OnClick	LoginActivity.java

6.1.3 模型层

模型组件负责完成业务逻辑。用户模块的的业务逻辑主要是完成数据库的操作，提交用户的信息到据库中。具体的模型组件见表Ｂ－6 所示。

表Ｂ－6：用户模块的模型组件

模型组件	描述
DBOpenHelper.java	数据库的基本操作,为复用组件
MD5Util.java	密码加密算法
LoginActivity.java	定义登录界面的相关操作
RegisterActivity.java	定义注册界面的相关操作
ResetPswActivity.java	定义改密界面的相关操作

6.2 账目模块设计

账目模块是本软件的主要模块，负责对账单的一系列处理展示以及新闻浏览，查看和编辑个人信息。

6.2.1 表示层设计

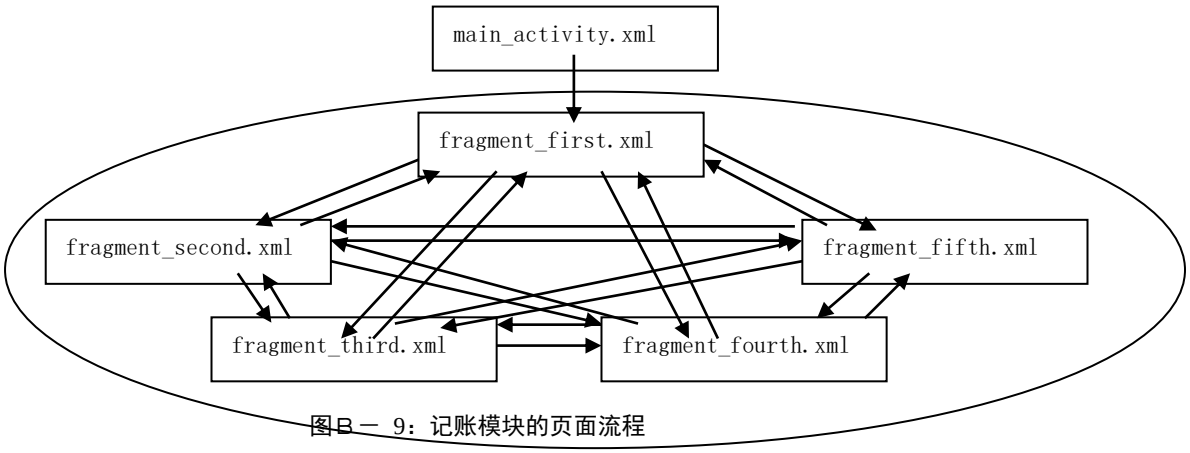
根据上述的功能介绍，总结出用户登录的页面如表Ｂ-7 所示。

表Ｂ－7：登录管理模块的页面设计

界面	XML	功能描述
主页面	main_activity.xml	管理主页面,承载导航栏
记账主页	fragment_firt.xml	承载记账按钮与相关信息
支出记账详情页	activity_add.xml	编辑并保存支出账目的信息
收入记账详情页	activity_income.xml	编辑并保存收入账目的信息
账单界面	fragment_second.xml	展示年月账单

报表界面	fragment_third.xml	展示年月报表
新闻界面	fragment_fourth.xml	显示财经网新闻
个人信息界面	fragment_fifth.xml	显示个人账户余额与登出按钮

根据界面流的设计可以确定各个界面的访问入口，以及界面之间切换关系，页面的流程图如图B－1 5所示。



图B－ 9：记账模块的页面流程

6.2.2 控制层设计

账目模块的控制层主要是设计用户的记账界面转换的流程控制。表B－8列出了每个activity的组件、入口、动作，以及出口。

表B－8：登录管理的控制层设计

事件	组件名	入口	动作	出口
选择模块	tab_layout_view	MainActivity.java	OnClick	MainActivity.java
模块切换	viewpager_content_view	MainActivity.java	Sweep	MainActivity.java
记账	btn_add	FirstFragment.java	OnClick	AddActivity.java
设置预算	btn_budget	FirstFragment.java	OnClick	FirstFragment.java
选择年份	year_spinner	SecondFragment.java	OnClick	SecondFragment.java
选择月份	month_spinner	SecondFragment.java	OnClick	SecondFragment.java
退出登录	btn_exit	FifthFragment.java	Onclick	LoginActivity.java
支付宝余额	btn_alipay	FifthFragment.java	OnClick	FifthFragment.java
微信余额	btn_wechat	FifthFragment.java	OnClick	FifthFragment.java
校园卡余额	btn_campusCard	FifthFragment.java	OnClick	FifthFragment.java
银行卡余额	btn_bankCard	FifthFragment.java	OnClick	FifthFragment.java

6.2.3 模型层设计

模型组件见表B－9。

表Ｂ－９：账目模块的模型组件

模型组件	描述
DBOpenHelper.java	数据库的基本操作,为复用组件
AddActivity.java	编辑账单信息
pubFun.java	日期转换

项 目 编 号	200602006
文 档 编 号	12
密 级	内部

Android 记账 APP 详细设计

V1.0

第五组

评 审 日 期： 2020 年 11 月 12 日

目 录

1. 导言

1.1 目的

该文档的目的是描述《Android 记账 APP》项目的详细设计，其主要内容包括：

- 系统功能简介
- 系统详细设计简述
- 各个模块的三层划分
- 最小模块组件的伪代码

本文档的预期的读者是：

- 开发人员
- 项目管理人员
- 测试人员

1.2 范围

该文档定义了系统的各个模块和模块接口，但未确定单元的具体实现，这部分内容将在实现中确定。

1.3 缩写说明

HR

Human Resource（人力资源管理）的缩写。

AS

Android Studio 安卓开发平台。

MVC

Ｍｏｄｅｌ－Ｖｉｅｗ－Ｃｏｎｔｒｏｌ（模式－视图－控制）的缩写，表示一个三层的结构体系。

1.4 术语定义

无

1.5 引用标准

[1]《企业文档格式标准》 V1.1
北京长江软件有限公司
[2]《软件详细设计报告格式标准》 V1.1
北京长江软件有限公司软件工程过程化组织

1.6 参考资料

[1]《Android Studio 开发实战——从零基础到 APP 开发》〔中〕欧阳燊
清华大学出版社

1.7 版本更新信息

本文档版本更新记录如表C-1：

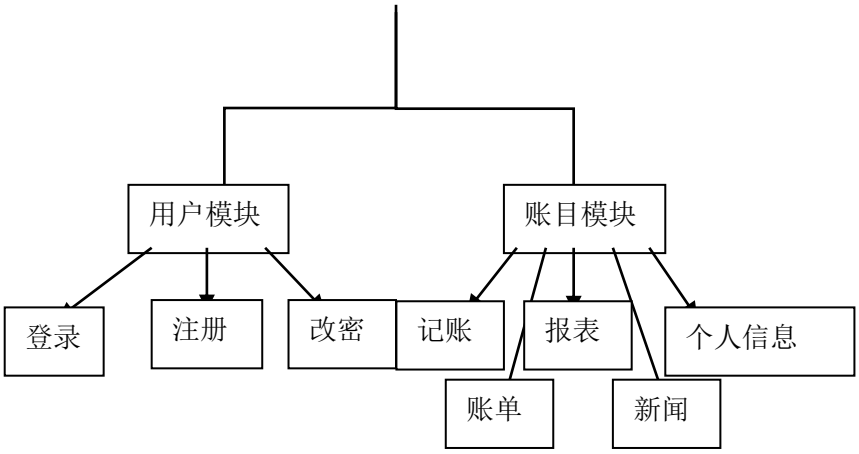
表C-1 版本更新记录

修改编号	修改日期	修改后版本	修改位置	修改内容概述
000	2020.11.9	1.0	全部	初始发布版本

2 系统设计概述

根据《Android 记账 APP》的概要设计，本系统按照功能角度分解，可以分为登录模块和账目模块。根据页面流的设计，登录模块分为用户登录、用户注册、找回密码等 3 个模块，账目模块分为记账、查看账单、查看报表、浏览新闻、个人信息等 5 个模块，他们的关系如图C-1，以下将分小节对各个部分分别进行详细设计。

Android 记账 APP



C-1 模块设计

3 详细设计概述

由于本软件采用ＭＶＣ的三层设计模式，采用面向对象的Ｊａｖａ语言以及 XML 的脚本语言。所以，基本采用面向对象的设计方法。在整个的开发过程中，尽可能采用复用的原则，例如采用标签库，统一数据库的基本操作，统一结果显示等。

本文档的详细设计主要是按照ＭＶＣ的三个层次分别编制视图层、控制层和模型层模块的伪代码。为下一步的编码提供基础。伪代码(Pseudocode)是一种算法描述语言。使用伪代码的目的是为了使被描述的算法可以容易地以任何一种编程语言实现。因此，伪代码必须结构清晰，代码简单，可读性好，并且类似自然语言。

4 登录模块的详细设计

登录模块主要实现用户本地登录的身份验证，登陆界面视图层、控制层和模型层三个层次的模块如表Ｃ－２所示。

表Ｃ－２：职位管理模块的三层模块

视图		控制器	模型
组件	布局文件		
et_user_name et_psw btn_login tv_register tv_find_psw	Activity_login.xml	LoginActivity.java	DBOpenHelper.java MD5Util.java

4.1 视图层

视图层主要实现表示层的功能，视图层包括组件。

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:orientation="vertical" android:layout_width="match_parent"
    android:layout_height="match_parent">
    <!--登录界面,用 LinearLayout-->
    <LinearLayout
        ……
        //设置相关属性
        ……
    >
        <!—用户名输入框-->
        <LinearLayout

            ……
            //基本参数自动设置
            ……
        >

            <ImageView
                android:id="@+id/imageView"
                ……
                //设置用户名的图标为图片控件背景
                ……
             />

            <EditText
                android:id="@+id/et_user_name"
                ……
                //设置缺省文本为“请输入用户名”
```

```
        ····
    />
</LinearLayout>

<!—密码输入框-->
<LinearLayout
    ····
        //设置相关属性
        ····
>

    <ImageView
        android:id="@+id/imageView2"
        ····
        //设置密码图标为图片控件背景
        ····
    />

    <EditText
        android:id="@+id/et_psw"
        ····
        //设置缺省文本为“请输入密码”
        ····
     />
</LinearLayout>

<!—登录按钮-->
<Button
    android:id="@+id/btn_login"
        ····
        //设置相关属性
        ····
    />
<!--显示 tv register , find_psw -->
<LinearLayout
    ····
    //设置相关属性
    ····
>
    <TextView
        android:id="@+id/tv_register"
        ····
        //设置相关属性
        ····
     />

    <TextView
        android:id="@+id/tv_find_psw"
        ····
        //设置相关属性
        ····
```

```
            />
        </LinearLayout>
    </LinearLayout>
</LinearLayout>
```

4.2 控制层

职位管理控制层共有 22 个 Action 文件，表Ｃ－4 是 AddJobAction 的伪代码描述。

表Ｃ－４: LoginActivity.java 的伪代码

```
/**
  * @System:  Online CV System
  * @Version: 1.0
  * @Copyright (C) 2020 by Group5.
  * @Class:   LoginActivity
  */
package com.example.jizhang_master;

import android.app.Activity;
import android.content.Intent;
import android.content.SharedPreferences;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.os.Bundle;
import android.text.TextUtils;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

import androidx.annotation.Nullable;
import androidx.appcompat.app.AppCompatActivity;

import com.example.jizhang_master.utils.MD5Util;

import static com.example.jizhang_master.VaryActivity.budget;
import static com.example.jizhang_master.VaryActivity.expend;
import static com.example.jizhang_master.VaryActivity.income;

public class LoginActivity extends AppCompatActivity {

    。。。。
    //定义组件接口变量
    。。。。
    @Override
    protected void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
```

```
        setContentView(R.layout.activity_login);
        init();
        initLoginStatus();

    }

    private void initLoginStatus() {
        。。。。。
        //用 SharedPreference 获取登录信息，判断当前是否登录并获取登录的用户名
        。。。。。
    }

    private void init() {


        //从 activity_login.xml 中获取控件 id
       。。。。。

        //立即注册控件的点击事件
        tv_register.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
             。。。
            }
        });

        //找回密码控件的点击事件
        tv_find_psw.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
               //用 Intent 类实现跳转到改密界面
            }
        });

        //登录按钮的点击事件
        btn_login.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                 //先调用 checkData 函数判断用户名密码是否匹配
                //判断通过用 Intent 类实现跳转到主界面
                }
            }
        });
    }

    /**
     * 判断用户名是否存在或密码是否正确，返回错误类型
     */
    private int checkData(String userName, String md5Psw){

      //调用 DBOpenHelper 打开数据库，将用户输入的用户名和密码放入搜索条件在数据库中搜
索，若有对应条目，则返回 1，无则返回 0.
    }
```

```
    //获取用户本月收入支出与预算信息并存入全局变量
    public void getVary(){
        //获取全局变量
    }



    /**
     *保存登录状态和登录用户名到 SharedPreferences 中
     */
    private void saveLoginStatus(boolean status,String userName){
        。。。。。
    }
    /**
     * 注册成功的数据返回至此
     * @param requestCode 请求码
     * @param resultCode 结果码
     * @param data 数据
     */
    @Override
    //显示数据， onActivityResult
    //startActivityForResult(intent, 1); 从注册界面中获取数据
    //int requestCode , int resultCode , Intent data
    // LoginActivity -> startActivityForResult -> onActivityResult();
    protected void onActivityResult(int requestCode, int resultCode, Intent data) {
        //super.onActivityResult(requestCode, resultCode, data);
        super.onActivityResult(requestCode, resultCode, data);
        。。。。。
        }
    }
}
```

4.3 模型层

表Ｃ－5 是 DBOpenHelper 的伪代码描述。

表Ｃ－５: DBOpenHelper 的伪代码

```
package com.example.jizhang_master;

import android.content.Context;
import android.database.sqlite.SQLiteDatabase;
```

```
import android.database.sqlite.SQLiteDatabase.CursorFactory;
import android.database.sqlite.SQLiteOpenHelper;


public class DBOpenHelper extends SQLiteOpenHelper {
    public DBOpenHelper(Context context, String name, CursorFactory factory,
                        int version){
        super(context, name, factory, version);
    }

    @Override
    //首次创建数据库的时候调用，一般可以执行建库，建表的操作
    //Sqlite 没有单独的布尔存储类型,它使用 INTEGER 作为存储类型,0 为 false,1 为 true
    public void onCreate(SQLiteDatabase db){
        //user table
        db.execSQL(
            °°°°°
        );



        //costDetail_tb
        db.execSQL(
            °°°°°
        );
    }

    @Override//当数据库的版本发生变化时，会自动执行
    public void onUpgrade(SQLiteDatabase db,int oldVersion,int newVersion){

    }

}
```

5 注册模块的详细设计

注册主要是获取新的用户的相关信息并存入数据库，注册模块的视图层、控制层和模型层三个层次的模块如表Ｃ－6 所示。

表Ｃ－６：注册模块的三层模块

视图		控制器	模型
组件	布局文件		
et_user_name et_psw et_psw_again et_phoneNum btn_register	activity_register.xml	RegisterActivity.java	DBOpenHelper.java MD5Util.java

5.1 视图层

```
<?xml version="1.0" encoding="utf-8"?>
<!--注册界面-->
<!--这里的布局放置是： 1 个 ImageView 控件，用于显示用户头像；3 个 EditText 控件，用于输入用户名、密码、再次输入密码；1 个 Button 控件为注册按钮-->
<!-- activity_register.xml 为 LinearLayout 布局-->
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/activity_register"
        。。。。
        //设置相关属性
        。。。。
>


    <!--四个编辑框-->
    <EditText
        android:id="@+id/et_user_name"
        。。。。
        //设置相关属性
        。。。。
    />
    <EditText
        android:id="@+id/et_psw"
        。。。。
        //设置相关属性
        。。。。
```

```
    />
    <EditText
        android:id="@+id/et_psw_again"
        ····
        //设置相关属性
        ····

    />
    <EditText
        android:id="@+id/et_phoneNum"
        ····
        //设置相关属性
        ····
    />

    <Button
        android:id="@+id/btn_register"
        ····
        //设置相关属性
        ····
    />
</LinearLayout>
```

5.2 控制层

```
package com.example.jizhang_master;


import android.content.ContentValues;
import android.content.Intent;
import android.content.SharedPreferences;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.os.Bundle;
import android.text.TextUtils;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.RelativeLayout;

import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;

import com.example.jizhang_master.utils.MD5Util;

public class RegisterActivity extends AppCompatActivity {

    //定义控件变量
```

```
。。。。。。

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    //设置页面布局 ,注册界面
    setContentView(R.layout.activity_register);

    init();
}

private void init() {


    //从 activity_register.xml 页面中获取对应的 UI 控件
    。。。。。。

    //注册按钮
    btn_register.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
        //点击后调用 isExistUserName 判断用户名是否存在，若存在则弹框提示，
       若不存在则调用 saveRegisterInfo 函数保存数据到数据库，并用 Intent 类
       带账户名参数跳转至登录界面

            }
        }
    });
}
```

```
/**
 * 获取控件中的字符串
 */
private void getEditString() {
    。。。。。
}

/**
 * 从 database 中读取输入的用户名，判断 database 中是否有此用户名
 */
private boolean isExistUserName(String userName) {
    //调用 DBOpenHelper
}

/**
 * 保存账号和密码到 database
```

```
         */
        private void saveRegisterInfo(String userName, String psw, String
phoneNumber) {
            //调用 DBOpenHelper 打开数据库，将账号和密码存入数据库
        }
    }
```

5.3 模型层

同为 DBOpenHelper.java

6 改密模块的详细设计

找回密码主要是实现更改用户对应密码，改密模块的视图层、控制层和模型层三个层次的模块如表Ｃ－７所示。

表Ｃ－７：改密模块的三层模块

视图		控制器	模型
组件	布局文件		
et_user_name_res et_phoneNum_res et_psw_res et_psw_again_res btn_respsw	activity_respsw.xml	ResetPswActivity.java	DBOpenHelper.java MD5Util.java

6.1 视图层

改密模块视图层包括 XML 布局文件等。详见文件。

6.2 控制层

改密模块控制层详见文件。

6.3 模型层

详见文件。

7 记账模块的详细设计

记账模块主要实现用户记录账目的功能，记账模块的视图层、控制层和模型层三个层次的模块如表Ｃ－8 所示。

表Ｃ－８：记账模块的三层模块

视图		控制器	模型
组件	布局文件		
btn_add btn_budget tv_month_expend tv_month_income et_money btn_ex1 btn_date et_note btn_submit	Fragment_first.xml activity_add.xml activity_income.xml	FirstFragment.java AddActivity.java IncomeActivity.java	DBOpenHelper.java

7.1 视图层

记账模块视图层包括 3 个 XML 文件。以下展示记账主视图的 XML 文件伪代码。

```
<?xml version="1.0" encoding="utf-8"?>

    <LinearLayout
         。。。。。
        //设置相关属性
        。。。。。
    >

        <TextView
            android:text="本月支出（元）"
             。。。。。
            //设置相关属性
            。。。。。
```

```
    />

    <TextView
        android:id="@+id/tv_month_expend"
         。。。。。
        //设置相关属性
        。。。。。
     />

    <LinearLayout
         。。。。。
        //设置相关属性
        。。。。。
    >

        <TextView
            android:text="本月收入"
            。。。。。
            //设置相关属性
            。。。。。
         />

        //显示本月收入
        <TextView
            android:id="@+id/tv_month_income"
            。。。。。
            //设置相关属性
            。。。。。
         />




        //预算设置按钮
        <Button
            android:id="@+id/btn_budget"
             。。。。。
            //设置相关属性
            。。。。。
        />

    </LinearLayout>

</LinearLayout>

//记账按钮
<Button
    android:id="@+id/btn_add"
     。。。。。
    //设置相关属性
    。。。。。
```

```
    />


    //提示文字
    <TextView
        android:id="@+id/tv_day_account"
        。。。。。
        //设置相关属性
        。。。。。
    />
    //今日账单列表
    <ListView
        android:id="@+id/list_view"
         。。。。。
        //设置相关属性
        。。。。。
    />
</LinearLayout>
```

7.2 控制层

以下详细描述具体记账的控制层伪代码。

```
package com.example.jizhang_master;

import android.app.Activity;
import android.app.DatePickerDialog;
。。。。。。


public class AddActivity extends AppCompatActivity {

    //定义变量
    。。。。。
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_add);
        initView();
        initicon();

        。。。。。
    }

    //初始化数据
    private void initView() {
        //获取各个图标的控件 id 并关联变量并创建一个 textView 数组
    }
```

```
        /**
         * 设置点击后颜色变化
         * @param i
         */
        private void set_color(int i){
           //定义一个全局变量 ex_num 记录当前选择的图标号
           //输入 i 为图标号，将 i 对应的标签字体颜色改为亮色，将其余标签字体颜色改
为暗色，将 ex_num 设置为 i。
        }



        /**
         * 图标点击监听
         */
        private void initicon() {
            //为每个图标设置点击响应，点击时调用 setColor（i），i 为对应图标号
        }


        /**
         * 账户监听,获取账户选择结果
         */
        Class                  spinnerSelectedListener                  implements
AdapterView.OnItemSelectedListener {
           // txtAccount = accountList[arg2];
        }

        /**
         * 打开日历，获取日期
         */
        private void openDate() {
            //调用 DatePickerDialog
        }

        /**
         * 设置日期
         */
        private  DatePickerDialog.OnDateSetListener  mDateSetListenerSatrt  =  new
DatePickerDialog.OnDateSetListener() {
            public void onDateSet(DatePicker view, int year, int monthOfYear,
                                  int dayOfMonth) {
               。。。。。。
            }
        };


        /**
         *判断选择的日期是否晚于当前日期
         */
```

```
        private Boolean afterNow(){
            //调用 calendar.getTime()获取选中的日期,
              调用 Calendar.getInstance().getTime()获取当前日期，比较先后顺序
        }


        /**
         * 获取数据至字符串
         */
        private void transdata(){
            //将输入框中的各个数据存入到对应变量
        }

        //更改记录本月数据的全局变量
        private void changeVary(){
           。。。。。。

        }
        /**
         * 保存输入数据至数据库
         */
        private void saveInfo() {

           。。。。。。
        }
        /**
         *判断输入数据是否合法
         */
        private boolean check(){
            //判断是否输入金额，否则弹框提示
            //调用 afterNow 函数判断日期是否合法，否则弹框提示
            //检查是否选择类别，否则弹框提示
            //检查无误返回 true
        }

        @Override
        protected void onStart(){
            super.onStart();
        }

        @Override
        protected void onResume() {
            super.onResume();
            //点击左上角 x 回到主页面
            findViewById(R.id.btn_return).setOnClickListener(new
View.OnClickListener() {
                @Override
                public void onClick(View view) {
                    //调用 Intent 类跳转到主界面
                }
            });
```

```
            //点击收入进入收入页面
            findViewById(R.id.btn_income).setOnClickListener(new
View.OnClickListener() {
                @Override
                public void onClick(View view) {
                    //调用 Intent 类跳转到收入界面            }
            });

            //点击确定
            findViewById(R.id.btn_submit).setOnClickListener(new
View.OnClickListener() {
                @Override
                public void onClick(View view) {
                    //调用 check 函数检查数据是否合法，若是则调用 Intent 类跳转到主
界面
                }
            });

            //点击“再记一笔”
            findViewById(R.id.btn_again).setOnClickListener(new
View.OnClickListener() {
                @Override
                public void onClick(View view) {
                    //调用 Intent 类跳转到支出界面
                });
        }

        //保存当月支出、收入和预算信息
        public void commitVary(){
            //保存对应数据到全局变量
        }

        @Override
        protected void onStop() {
            super.onStop();
            finish();
        }
    }
```

7.3 模型层

详见文件。

8 账单模块的详细设计

账单模块的视图层、控制层和模型层三个层次的模块如表Ｃ－9 所示。

表Ｃ－９：账单模块的三层模块

视图		控制器	模型
组件	布局文件		
year_spinner month_spinner lv_expense	fragment_second.xml fragment_one_item.xml	SecondFragment.java	DBOpenHelper.java

8.1 视图层

视图层包括两个 xml 布局文件，其中 fragment_second.xml 是主要界面布局,fragment_one_item.xml 是列表项目布局。以下展示主要界面布局的伪代码。

```
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout
    。。。。。
>

    <LinearLayout
        。。。。。 >

        //年份选择下拉栏
        <Spinner
            android:id="@+id/year_spinner"
            。。。。。
        />

        //月份选择下拉栏
```

```
            <Spinner
                android:id="@+id/month_spinner"
                ◦◦◦◦◦
            />

            //条目列表
            <ListView
                android:id="@+id/lv_expense"
                ◦◦◦◦
            >

            </ListView>

        </LinearLayout>

    </FrameLayout>
```

8.2 控制层

```
    package com.example.jizhang_master;

    import android.app.Activity;
    import android.content.DialogInterface;
    import android.content.Intent;
    import android.content.SharedPreferences;
    import android.database.Cursor;
    import android.database.sqlite.SQLiteDatabase;
    ◦◦◦◦◦◦◦


    public class SecondFragment extends Fragment {

       //初始化数据


        public SecondFragment() {
            // Required empty public constructor
        }

        @Override
        public View onCreateView(LayoutInflater inflater, ViewGroup container,
Bundle savedInstanceState) {

            //定义控件变量
            initSpinner( v);
            getData(monthID,yearID,v);
            resetpic(v);
            return v;
        }
```

```
        private void resetpic(final View v){
            handler=new Handler(Looper.myLooper()){
                @Override
                public void handleMessage(Message msg){
                    super.handleMessage(msg);
                   。。。。。
                }
            };
        }

        //初始化选择器
        private void initSpinner(View v){
           。。。。。

        }

        /**
         * 选择 年份 事件 监听器
         */
        class               year_spinnerSelectedListener               implements
AdapterView.OnItemSelectedListener{
            public void onItemSelected(AdapterView<?> arg0, View arg1, int arg2,
                                       long arg3) {
                。。。。。
            }
            public void onNothingSelected(AdapterView<?> arg0) {
            }
        }
        /**
         * 选择 月份 事件 监听器
         */
        class              month_spinnerSelectedListener               implements
AdapterView.OnItemSelectedListener {
            public void onItemSelected(AdapterView<?> arg0, View arg1, int arg2,
                                       long arg3) {
                。。。。。
            }
                public void onNothingSelected(AdapterView<?> arg0) {
                }

            }

        //获取数据
        private void getData(String monthID,String yearID,View v){
               。。。。。
        }

        //提交当月数据
        public void commitVary(){
```

```
            。。。。。
        }
    }
```

8.3 模型层

详见文件。

9 报表模块的详细设计

报表模块主要实现了输出某月或某年用户消费或收入的饼图的功能，报表模块的视图层、控制层和模型层三个层次的模块如表Ｃ－１０所示。

表Ｃ－１０：报表模块的三层模块

视图		控制器	模型
组件	布局文件		
year_spinner month_spinner btn_change ll_expense_piechart	fragment_third.xml	ThirdFragment.java	DBOpenHelper.java

9.1 视图层

详见文件

9.2 控制层

详见文件

9.3 模型层

详见文件

10 新闻模块的详细设计

新闻模块主要实现浏览财经网内容的功能，新闻模块的视图层、控制层和模型层三个层次的模块如表Ｃ－１１所示。

表Ｃ－１１：新闻模块的三层模块

视图		控制器	模型
组件	布局文件		
wv_web	fragment_fourth.xml	FourthFragment.java	

10.1 视图层

详见文件。

10.2 控制层

伪代码如下

```
package com.example.jizhang_master;

import android.graphics.Bitmap;
import android.net.http.SslError;
import android.os.Build;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.webkit.SslErrorHandler;
import android.webkit.WebChromeClient;
```

```
import android.webkit.WebResourceRequest;
import android.webkit.WebView;
import android.webkit.WebViewClient;

import androidx.annotation.Nullable;
import androidx.annotation.RequiresApi;
import androidx.fragment.app.Fragment;

public class FourthFragment extends Fragment {
    private WebView mWebview;
    @Nullable
    @Override
    public View onCreateView(LayoutInflater inflater, @Nullable ViewGroup
container, Bundle savedInstanceState) {
        。。。。。
    }

    //设置 web 视图
    class MyWebViewClient extends WebViewClient {
        。。。。。
    }

    class MyWebChromeClient extends WebChromeClient {
        。。。。。
    }

}
```

10.3 模型层

详见文件。

11 个人信息模块的详细设计

个人信息模块主要实现展示账户余额以及退出登录的功能，个人信息模块的视图层、控制层和模型层三个层次的模块如表Ｃ－１２所示。

表Ｃ－１２：个人信息的三层模块

视图		控制器	模型
组件	布局文件		

tv_userID btn_weixin btn_alipay btn_bankCard btn_campusCard	fragment_fifth.xml	FifthFragment.java	DBOpenHelper.java

11.1 视图层

客户端管理视图层包括ＪＳＰ组件、Ｆｏｒｍ组件以及标签库等。共有 11 个ＪＳＰ文件，详见ｄｅｍｏ和伪代码文件 WSZP-PD-DD-Client-ViewJSP.doc。共有 10 个Ｆｏｒｍ文件，每个Ｆｏｒｍ模块的伪代码详见文件WSZP-PD-DD-Client-ViewForm.doc。标签库详见12.1描述。

11.2 控制层

```
package com.example.jizhang_master;

import android.content.ContentValues;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.SharedPreferences;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;

import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.appcompat.app.AlertDialog;
import androidx.fragment.app.Fragment;



public class FifthFragment extends Fragment {
    //定义变量
    。。。。。。
    @Nullable
    @Override
```

```
        public View onCreateView(LayoutInflater inflater, @Nullable ViewGroup
container, Bundle savedInstanceState) {
            //建立连接
            。。。。。
            initAccount();
            return view;
        }

        //初始化账户信息
        public void initAccount(){
           //调用 DBOpenHelper 打开数据库，获取当前登录用户的账户信息并填入对应控
件。

        }


        @Override
        public void onViewCreated(@NonNull View view, @Nullable Bundle
savedInstanceState) {
            super.onViewCreated(view, savedInstanceState);
            //调用 Intent 类跳转到登录界面

        /**
         * 账户余额设置弹框
         *
         */
        public void alert_edit(final Button btn, final String account) {
           。。。。。
        }

        public void saveAccount(String money,String account){
            //保存账户信息到数据库
            。。。。。
        }

    }
```

11.3 模型层

详见文件

12 配置文件

13.1 AndroidMainfest.Xml 配置文件

```
    <?xml version="1.0" encoding="utf-8"?>
    <manifest xmlns:android="http://schemas.android.com/apk/res/android"
        package="com.example.jizhang_master" >

    //申请权限
        <uses-feature android:name="android.hardware.camera.autofocus" />
        <uses-permission android:name="android.permission.INTERNET" />
        <uses-permission  android:name="android.permission.MANAGE_WRITE_SETTINGS"
/>

    //设置基本参数
        <application
            android:usesCleartextTraffic="true"
            android:allowBackup="true"
            android:icon="@drawable/icon"
            android:label="@string/app_name"
            android:roundIcon="@mipmap/ic_launcher_round"
            android:supportsRtl="true"
            android:theme="@style/AppTheme" >
            <activity
                android:name=".MainActivity"
                android:label="@string/title_activity_main" >
            </activity>

    //注册 activity
             <activity
                android:name=".LoginActivity"
                android:label="This is LoginActivity"
                android:theme="@style/AppTheme.NoActionBar" />


            <activity
                android:name=".RegisterActivity"
                android:theme="@style/AppTheme.NoActionBar" />
            <activity android:name=".FindScanActivity"/>
            <activity android:name=".ScanResultActivity"/>
            <activity android:name=".MortgageActivity"
                android:theme="@style/AppTheme.NoActionBar"/>
            <activity android:name=".CalculatorActivity"
```

```
            android:theme="@style/AppTheme.NoActionBar"/>
        <activity
            android:name=".ResetPswActivity"
            android:theme="@style/AppTheme.NoActionBar" />
        <activity
            android:name=".NavigationActivity"
            android:theme="@style/AppTheme.NoActionBar" />
        <activity
            android:name=".SplashActivity"
            android:theme="@style/AppTheme.NoActionBar" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>

        </activity>
        <activity android:name=".AddActivity"
            android:theme="@style/AppTheme.NoActionBar"/>

        <activity android:name=".IncomeActivity"
            android:theme="@style/AppTheme.NoActionBar"/>

        <activity android:name=".VaryActivity"
            android:theme="@style/AppTheme.NoActionBar"/>
    </application>

</manifest>
```

项 目 编 号	**200602006**
文 档 编 号	**17**
密 级	内部

Android 记账 APP 测试设计

V1.0

第五组

评 审 日 期： 2020 年 11 月 23 日

目 录

1 导言

1.1 目的

该文档的目的是描述网上招聘系统项目客户端的系统测试设计，其主要内容包括：

- 测试总体设计
- 测试用例设计

本文档的预期的读者是：

- 项目管理人员
- 测试人员

1.2 范围

该文档为 Android 记账 APP 的系统测试设计，其中包括功能测试和性能测试的用例描述以及性能测试的测试脚本，为测试人员进行功能测试和性能测试提供标准和依据，以及详尽的测试步骤和方法。

1.3 缩写说明

JSP

Java Server Page（Java 服务器页面）的缩写，一个脚本化的语言。

MVC

Ｍｏｄｅｌ－Ｖｉｅｗ－Ｃｏｎｔｒｏｌ（模式－视图－控制）的缩写，表示一个三层的结构体系。

1.4 术语定义

功能性测试

按照系统需求定义中的功能定义部分对系统实行的系统级别的测试。

非功能性测试

按照系统需求定义中的非功能定义部分（如系统的性能指标，安全性能指标等）对系统实行的系统级别的测试。

测试用例

测试人员设计出来的用来测试软件某个功能的一种情形。

1.5 引用标准

[1] 《企业文档格式标准》
北京长江软件有限公司
[2] 《软件测试设计报告格式标准》
北京长江软件有限公司软件工程过程化组织

1.6 参考资料

[1] 《LoadRunner 使用手册》
北京长江软件有限公司编制

1.7 版本更新信息

本文档的更新信息如表F-1.

表F-1 版本更新记录

修改编号	修改日期	修改后版本	修改位置	修改内容概述
000	2020. 11. 27	1.0	全部	初始发布版本

2.测试设计

由于本次测试主要是针对需求进行的系统测试，包括功能测试和性能测试的技术，功能测试是执行指定的工作流程，性能测试是将功能测试过程中的测试用例改为 100 条以测试软件的性能。

2.1 测试范围

系统测试依据的系统的应用工作流：

1） 注册登录：在登录界面，输入未注册的信息，若不存在该用户则会提示。然后点击“注册”按钮进入注册页面，输入信息并点击“确定”按钮，返回登录页面，再输入密码，密码正确后进入主页面。点击“我的”按钮进入我的信息界面，点击“退出登录”按钮进入登录页面。点击“忘记密码”按钮进入改密页面，输入信息后点击“确定”按钮，返回登录页面。

2） 记账页面：登录后进入主界面，首先看到记账页面，上方显示本月支出、本月收入和本月预算，点击“本月预算”按钮会弹窗设置预算金额。再下方是“记一笔”按钮，点击后进入账目信息页面。下方是今日账单的明细表。

3） 记录账目信息：在记账页面点击“记一笔”按钮后进入账目信息页面，最上方有“支出”、“收入”两个按钮，点击跳转到对应的记账类别信息页面，第二排左边为“选择账户”下拉框，可以不选择，右边是“金额”编辑框，点击输入数额。下面是若干个对应消费或收入类型的图标，点击变色表示选中。再下方左侧为日期选择框，点击可选择日期，默认为当日日期。右边是“备注信息”编辑框，点击可输入备注信息。最下方有两个按钮，点击“再记一笔”按钮则提交数据并再次返回到该页，点击“确定”按钮则提交数据并返回主页面。点击左上角“X”按钮直接返回主页面。

4） 账单页面：点击主界面下方“账单”按钮，进入账单页面。点击“年份”下拉框选择年份，点击“月份”下拉框选择月份，默认数值为当年当月。下方是对应年月的具体账目信息。账目信息显示消费或收入类型及其对应图标、日期、备注和金额。长按某条账目可删除该数据，单击某条账目跳转到账目修改页面。

5） 账目修改页面：单击某条账目跳转到该页面，页面主体与账目信息页面一制，但对应信息显示为选择的账目的信息，可以修改信息，然后点击下方“确定”按钮，返回主界面。

6） 报表页面：点击主界面下方“报表”按钮，进入报表页面。点击“年份”下拉框选择年份，点击“月份”下拉框选择月份，默认数值为当年当月。下方是显示当月或当年支出或收入的总金额，点击右边的图标可以切换收入或支出视图。最下方是对应月或年的支出或收入饼状图，不同类型的支出或消费由对应比例以不同颜色显示，有线条标出对应类型和金额。

7） 新闻页面： 点击主界面下方“新闻”按钮，进入新闻页面。页面显示“财经网”的内容，用户可以点击并浏览相关新闻。

8） 个人信息页面：点击主界面下方“我的”按钮，进入个人信息页面。最上方显示当前登录用户名。下方有四个账户及其对应余额，默认无余额。点击某个账户可弹窗设置余额。

2.2 测试覆盖设计

由于本次测试是系统测试，测试的依据是系统需求，测试的设计应该满足对需求的覆盖，所以，采用的测试方法主要是黑盒测试，包括等价类划分（有效测试和无效测试）、边界值和错误猜测法等。表 F－2 就是测试用例覆盖矩阵。

表 F-2：测试用例功能/性能覆盖矩阵

序号	功能项	测试用例	优先级
01	所有页面的转移正确	TestCase-FUNC-01	中
02	账单信息正确	TestCase-FUNC-02	高
03	报表饼状图正确	TestCase-FUNC-03	高
04	正常记账流程-有（无）效数据	TestCase-FUNC-04	高
05	登录页面－无效数据	TestCase-FUNC-05	高
06	注册页面－无效数据	TestCase-FUNC-06	高
07	改密页面－无效数据	TestCase-FUNC-07	高
08	记账后当月信息能正确更新	TestCase-FUNC-08	高
09	记账后账户余额能正确更新	TestCase-FUNC-09	高
10	删除账目后当月信息能正确更新	TestCase-FUNC-10	中
11	删除账目后账户余额能正常更新	TestCase-FUNC-11	中
12	新闻页面能正常浏览新闻	TestCase-FUNC-12	中
13	大量数据的性能测试	TestCase-Perf-1	高

3.测试用例

按照上面的测试矩阵表，设计相应的测试用例如下。

3.1 用例一：页面转移的正确性

这个测试用例的测试编号是 TestCase-FUNC-01，测试内容是测试所有转移页面的正确性，同时所有的页面都按照需求有正确的显示。表 F-3 是这个测试用例的具体设计。

表 F-3： TestCase-FUNC-01 测试用例

<table>
<tr><td colspan="3">测试项目名称： Android 记账 APP</td></tr>
<tr><td>测试用例编号：CV-FUNC-01</td><td rowspan="2">测试人员：
付广庆</td><td rowspan="2">测试时间：
2020/11/23</td></tr>
<tr><td>测试项目标题：转移页面的正确性</td></tr>
<tr><td colspan="3">测试内容：
验证页面能够随按钮响应进行相应的跳转。</td></tr>
<tr><td colspan="3">测试环境与系统配置：
详见《测试计划》</td></tr>
<tr><td>测试输入数据</td><td colspan="2">点击注册->点击改密->登录->点击“记一笔”->点击“确定”->点击账单->点击报表->点击新闻->点击我的->点击“退出”</td></tr>
<tr><td colspan="3">测试次数：每个测试过程做 2 次。</td></tr>
<tr><td colspan="3">预期结果：
对于正常数据能够转到相应页面，异常数据能够报错</td></tr>
<tr><td colspan="3">测试过程：
打开 APP,进行测试输入数据中的操作</td></tr>
<tr><td colspan="3">测试结果：对于正常数据能够转到相应页面，异常数据能够报错</td></tr>
<tr><td colspan="3">测试结论： 测试成功</td></tr>
<tr><td colspan="3">实现限制：</td></tr>
<tr><td colspan="3">备注：</td></tr>
</table>

3.2 用例二：显示账单信息

这个测试用例的测试编号是 TestCase-FUNC-02，测试内容是测试账单页面账目显示的正确性。表 F-4 是这个测试用例的具体设计。

表 F-4：TestCase-FUNC-02 测试用例

<table>
<tr><td colspan="3">测试项目名称： Android 记账 APP</td></tr>
<tr><td>测试用例编号：TestCase-FUNC-02</td><td rowspan="2">测试人员：
卢鹏宇</td><td rowspan="2">测试时间：
2020/11/23</td></tr>
<tr><td>测试项目标题：账单信息正确</td></tr>
<tr><td colspan="3">测试内容：
查看账单页面账目是否正确显示</td></tr>
<tr><td colspan="3">测试环境与系统配置：
Android 虚拟机 7.0</td></tr>
<tr><td>测试输入数据</td><td colspan="2">无</td></tr>
<tr><td colspan="3">测试次数：
至少 2 次</td></tr>
<tr><td colspan="3">预期结果：
账单正确显示。</td></tr>
<tr><td colspan="3">测试过程：
打开 APP，点击账单按钮，更改年份和月份，查看账单是否正确显示</td></tr>
<tr><td colspan="3">测试结果：账单显示正确</td></tr>
<tr><td colspan="3">测试结论： 测试成功无异常。</td></tr>
<tr><td colspan="3">实现限制：无</td></tr>
<tr><td colspan="3">备注：无</td></tr>
</table>

3.3 用例三：显示报表信息

这个测试用例的测试编号是 TestCase-FUNC-03，测试内容是测试报表页面的正确性，同时所有的页面都按照需求有正确的显示。表 F-5 是这个测试用例的具体设计。

表 F-5：TestCase-FUNC-03 测试用例

<table>
<tr><td colspan="3">测试项目名称： Android 记账 APP</td></tr>
<tr><td>测试用例编号：TestCase-FUNC-03</td><td rowspan="2">测试人员：
卢鹏宇</td><td rowspan="2">测试时间：
2020/11/23</td></tr>
<tr><td>测试项目标题：报表饼状图正确</td></tr>
<tr><td colspan="3">测试内容：
查看报表是否正确显示，统计数据是否有误。</td></tr>
<tr><td colspan="3">测试环境与系统配置：
虚拟机 Android7.1.2</td></tr>
<tr><td>测试输入数据</td><td colspan="2">无</td></tr>
<tr><td colspan="3">测试次数：
至少测试 2 次</td></tr>
<tr><td colspan="3">预期结果：
在选择年份和月份后页面能正确显示对应饼状图</td></tr>
<tr><td colspan="3">测试过程：
打开 APP，点击报表按钮，选择年份和月份后，查看饼状图显示是否正确。</td></tr>
<tr><td colspan="3">测试结果：饼状图显示正确</td></tr>
<tr><td colspan="3">测试结论： 测试成功，无异常。</td></tr>
<tr><td colspan="3">实现限制：无</td></tr>
<tr><td colspan="3">备注：无</td></tr>
</table>

3.4 用例四：正常记账流程测试

这个测试用例的测试编号是 TestCase-FUNC-04，测试内容是测试正常的记账流程是否能实现，同时所有的页面都按照需求有正确的显示。表 F-6 是这个测试用例的具体设计。

表 F-6：TestCase-FUNC-04 测试用例

<table>
<tr><td colspan="3">测试项目名称： Android 记账 APP</td></tr>
<tr><td>测试用例编号：TestCase-FUNC-04</td><td rowspan="2">测试人员：
黄戈里</td><td rowspan="2">测试时间：
2020/11/23</td></tr>
<tr><td>测试项目标题：正常记账流程-有（无）效数据</td></tr>
<tr><td colspan="3">测试内容：
-- 验证未填入金额时点击确定是否能记账。
-- 验证选择迟于当日的日期时点击确定是否能成功记账。
-- 验证未选择收入或支出类型时点击确定是否能成功记账。
-- 验证在以上选项都选择或正确填写后能否成功记账。</td></tr>
<tr><td colspan="3">测试环境与系统配置：
虚拟机 Android7.1.2</td></tr>
<tr><td>测试输入数据</td><td colspan="2">1.金额：无，日期：2020/11/23，类别：交通，
2.金额：5，日期：2020/11/26，类别：交通
3.金额：5，日期：2020/11/23，类别：无
4.金额：5，日期：2020/11/23，类别：交通</td></tr>
<tr><td colspan="3">测试次数：每个测试过程做 2 次。</td></tr>
<tr><td colspan="3">预期结果：
-- 用户不填写金额就点击确定会弹框提示“请填写金额”
-- 用户选择迟于当日的日期点击确定会弹框提示“不能预知未来”
-- 用户未选择收入或支出类型就点击确定会弹框提示“请选择类别”
-- 用户完整且正确填写账目信息并点击确定后，弹框提示“记账成功”并跳转至主页面。</td></tr>
<tr><td colspan="3">测试过程：
打开 APP，点击“记一笔”，按测试输入数据进行记账操作</td></tr>
<tr><td colspan="3">测试结果：
符合预期</td></tr>
<tr><td colspan="3">测试结论：测试成功，功能实现</td></tr>
<tr><td colspan="3">实现限制：</td></tr>
<tr><td colspan="3">备注：</td></tr>
</table>

3.5 用例五：登录页面的无效性测试

这个测试用例的测试编号是 TestCase-FUNC-05，测试内容是测试登录页面在非正常输入时系统的异常处理，同时所有的页面都按照需求有正确的显示。表 F-7 是这个测试用例的具体设计。

表 F-7：TestCase-FUNC-05 测试用例

<table>
<tr><td colspan="3">测试项目名称： Android 记账 APP</td></tr>
<tr><td>测试用例编号：TestCase-FUNC-05</td><td rowspan="2">测试人员：
付广庆</td><td rowspan="2">测试时间：
2020/11/23</td></tr>
<tr><td>测试项目标题：登录页面－无效数据</td></tr>
<tr><td colspan="3">测试内容：
对于登录页面，测试其对异常数据的处理</td></tr>
<tr><td colspan="3">测试环境与系统配置：虚拟机 Android7.1.2</td></tr>
<tr><td>测试输入数据</td><td colspan="2">1.用户名：xxx，密码：123
2.用户名：fgq，密码：233333</td></tr>
<tr><td colspan="3">测试次数：每个测试过程做 2 次。</td></tr>
<tr><td colspan="3">预期结果：
对于数据 1，弹框提示用户名不存在；对于数据 2，弹框提示密码错误</td></tr>
<tr><td colspan="3">测试过程：
打开 APP，在登录界面输入测试数据，并点击登录按钮。</td></tr>
<tr><td colspan="3">测试结果：
符合预期</td></tr>
<tr><td colspan="3">测试结论：测试成功，页面异常处理正常</td></tr>
<tr><td colspan="3">实现限制：无</td></tr>
<tr><td colspan="3">备注：</td></tr>
</table>

3.6 用例六：注册页面的无效性测试

这个测试用例的测试编号是 TestCase-FUNC-06，测试内容是测试注册页面在非正常输入时系统的异常处理，同时所有的页面都按照需求有正确的显示。表 F-8 是这个测试用例的具体设计。

表 F-8：TestCase-FUNC-06 测试用例

<table>
<tr><td colspan="3">测试项目名称： Android 记账 APP</td></tr>
<tr><td>测试用例编号：TestCase-FUNC-06</td><td rowspan="2">测试人员：
付广庆</td><td rowspan="2">测试时间：
2020/11/23</td></tr>
<tr><td>测试项目标题：注册页面－无效数据</td></tr>
<tr><td colspan="3">测试内容：
对于登录页面，测试其对异常数据的处理</td></tr>
<tr><td colspan="3">测试环境与系统配置：
虚拟机 Android7.1.2</td></tr>
<tr><td>测试输入数据</td><td colspan="2">1.用户名：xxx，密码：123 再次输入密码：1234，手机：12341244556
2.用户名：fgq，密码：123，再次输入密码：123，手机：13333333333</td></tr>
<tr><td colspan="3">测试次数：每个测试过程做 1 次。</td></tr>
<tr><td colspan="3">预期结果：
对于输入 1，弹框提示“两次输入密码不一致”；对于输入 2，弹框提示用户名已存在。</td></tr>
<tr><td colspan="3">测试过程：
打开 APP，在注册界面输入测试数据，并点击注册按钮。</td></tr>
<tr><td colspan="3">测试结果：
符合预期</td></tr>
<tr><td colspan="3">测试结论：测试成功，页面异常处理正常</td></tr>
<tr><td colspan="3">实现限制：无</td></tr>
<tr><td colspan="3">备注：无</td></tr>
</table>

3.7 用例七：改密页面的无效性测试

这个测试用例的测试编号是 TestCase-FUNC-07，测试内容是测试改密页面在非正常输入时系统的异常处理，同时所有的页面都按照需求有正确的显示。表 F-9 是这个测试用例的具体设计。

表 F-9：TestCase-FUNC-07 测试用例

<table>
<tr><td colspan="3">测试项目名称： Android 记账 APP</td></tr>
<tr><td>测试用例编号：TestCase-FUNC-07</td><td rowspan="2">测试人员：
付广庆</td><td rowspan="2">测试时间：
2020/11/23</td></tr>
<tr><td>测试项目标题：改密页面－无效数据</td></tr>
<tr><td colspan="3">测试内容：
对于改密页面，测试其对异常数据的处理</td></tr>
<tr><td colspan="3">测试环境与系统配置：
虚拟机 Android7.1.2</td></tr>
<tr><td>测试输入数据</td><td colspan="2">1.用户名：fgq，密码：1234，再次输入新密码：123，手机号：18367748286
2.用户名：fgq，密码：1234，再次输入新密码：1234，手机号：1234567
3.用户名：xxx，密码：123，再次输入新密码：123，手机号：18367748286</td></tr>
<tr><td colspan="3">测试次数：每个测试过程做 1 次。</td></tr>
<tr><td colspan="3">预期结果：
对于输入 1，弹框提示“两次输入密码不一致”。对于输入 2，弹框提示“手机号不正确”。对于输入三，弹框提示“用户名不存在”。</td></tr>
<tr><td colspan="3">测试过程：
打开 APP，在改密界面输入测试数据，并点击确定按钮。</td></tr>
<tr><td colspan="3">测试结果：
符合预期</td></tr>
<tr><td colspan="3">测试结论： 测试成功，能识别异常数据。</td></tr>
<tr><td colspan="3">实现限制：无</td></tr>
<tr><td colspan="3">备注：无</td></tr>
</table>

3.8 用例八：记账后当月信息更新测试

这个测试用例的测试编号是 TestCase-FUNC-08，测试内容是验证在用户记账后当月的支出、收入、预算信息是否能够正确更新，同时所有的页面都按照需求有正确的显示。表 F-10 是这个测试用例的具体设计。

表 F-10：TestCase-FUNC-08 测试用例

<table>
<tr><td colspan="3">测试项目名称： Android 记账 APP</td></tr>
<tr><td>测试用例编号：TestCase-FUNC-08</td><td rowspan="2">测试人员：
李泽勤</td><td rowspan="2">测试时间：
2020/11/23</td></tr>
<tr><td>测试项目标题：记账后当月信息更新测试</td></tr>
<tr><td colspan="3">测试内容：
验证在用户记账后当月的支出、收入、预算信息是否能够正确更新</td></tr>
<tr><td colspan="3">测试环境与系统配置：
虚拟机 Android7.1.2</td></tr>
<tr><td>测试输入数据</td><td colspan="2">1.支出,金额：5，日期：2020/11/23，类别：交通
2.收入，金额：6，日期：2020/11/22，类别：奖金</td></tr>
<tr><td colspan="3">测试次数：每个测试过程做 1 次。</td></tr>
<tr><td colspan="3">预期结果：
点击确定按钮后返回主页面，显示本月支出增加 5，本月收入增加 6，本月预算减少 5。</td></tr>
<tr><td colspan="3">测试过程：
打开 APP，按照测试数据记账，返回主页面查看信息是否正确。</td></tr>
<tr><td colspan="3">测试结果：
符合预期</td></tr>
<tr><td colspan="3">测试结论： 测试成功，当月信息能够成功更新</td></tr>
<tr><td colspan="3">实现限制：无</td></tr>
<tr><td colspan="3">备注：无</td></tr>
</table>

3.9 用例九：记账后账户余额更新测试

这个测试用例的测试编号是 TestCase-FUNC-09，测试内容是验证在用户记账后对应的账户余额信息是否能够正确更新，同时所有的页面都按照需求有正确的显示。表 F-11 是这个测试用例的具体设计。

表 F-11：TestCase-FUNC-09 测试用例

<table>
<tr><td colspan="3">测试项目名称： Android 记账 APP</td></tr>
<tr><td>测试用例编号：TestCase-FUNC-09</td><td rowspan="2">测试人员：
解博文</td><td rowspan="2">测试时间：
2020/11/23</td></tr>
<tr><td>测试项目标题：记账后账户余额更新测试</td></tr>
<tr><td colspan="3">测试内容：
验证在用户记账后对应的账户余额信息是否能够正确更新</td></tr>
<tr><td colspan="3">测试环境与系统配置：
虚拟机 Android7.1.2</td></tr>
<tr><td>测试输入数据</td><td colspan="2">1.支出，账户：微信，金额：5，日期：2020/11/23，类型：餐饮
2.收入，账户：支付宝，金额：10，日期：2020/11/23，类型：薪水
3.支出，账户：银行卡，金额：100，日期：2020/11/2，类型：医疗
4.支出，账户：校园卡，金额：3，日期：2020/11/15，类型：餐饮</td></tr>
<tr><td colspan="3">测试次数：每个测试过程做 1 次。</td></tr>
<tr><td colspan="3">预期结果：
对于输入 1，记账完成后查看微信余额减少 5；对于输入 2，记账完成后查看支付宝余额增加 10；对于输入 3，记账完成后查看银行卡余额减少 100；对于输入 4，记账完成后校园卡余额减少 3。</td></tr>
<tr><td colspan="3">测试过程：
打开 APP，输入测试数据，点击“我的”按钮查看账户余额是否正确更新</td></tr>
<tr><td colspan="3">测试结果：
符合预期</td></tr>
<tr><td colspan="3">测试结论： 测试成功，记账后能够正确更新对应余额信息</td></tr>
<tr><td colspan="3">实现限制：无</td></tr>
<tr><td colspan="3">备注：无</td></tr>
</table>

3.10 用例十：删除账目后当月信息更新测试

这个测试用例的测试编号是 TestCase-FUNC-10，测试内容是验证删除账目后当月支出、收入、预算信息能否正确更新，同时所有的页面都按照需求有正确的显示。表 F-12 是这个测试用例的具体设计。

表 F-12：TestCase-FUNC-10 测试用例

<table>
<tr><td colspan="3">测试项目名称： Android 记账 APP</td></tr>
<tr><td>测试用例编号：TestCase-FUNC-10</td><td rowspan="2">测试人员：
宋依航</td><td rowspan="2">测试时间：
2020/11/23</td></tr>
<tr><td>测试项目标题：删除账目后当月信息更新测试</td></tr>
<tr><td colspan="3">测试内容：
验证删除账目后当月支出、收入、预算信息能否正确更新</td></tr>
<tr><td colspan="3">测试环境与系统配置：
虚拟机 Android7.1.2</td></tr>
<tr><td>测试输入数据</td><td colspan="2">无</td></tr>
<tr><td colspan="3">测试次数：每个测试过程做 1 次。</td></tr>
<tr><td colspan="3">预期结果：
删除某条当月账目后，查看主页面的当月支出和预算或者收入数值正确更新。</td></tr>
<tr><td colspan="3">测试过程：
打开 APP，点击“账单”按钮，删除某条账目，点击“首页”按钮，查看当月信息是否正确更新</td></tr>
<tr><td colspan="3">测试结果：符合预期</td></tr>
<tr><td colspan="3">测试结论： 测试成功，删除账目后，当月信息能正确更新。</td></tr>
<tr><td colspan="3">实现限制：无</td></tr>
<tr><td colspan="3">备注：无</td></tr>
</table>

3.11 用例十一：删除账目后账户余额更新测试

这个测试用例的测试编号是 TestCase-FUNC-11，测试内容是验证删除账目后当月支出、收入、预算信息能否正确更新，同时所有的页面都按照需求有正确的显示。表 F-13 是这个测试用例的具体设计。

表 F-13：TestCase-FUNC-11 测试用例

<table>
<tr><td colspan="3">测试项目名称： Android 记账 APP</td></tr>
<tr><td>测试用例编号：TestCase-FUNC-11</td><td rowspan="2">测试人员：
解博文</td><td rowspan="2">测试时间：
2020/11/23</td></tr>
<tr><td>测试项目标题：删除账目后账户余额更新测试</td></tr>
<tr><td colspan="3">测试内容：
-- 验证删除账目后当月支出、收入、预算信息能否正确更新</td></tr>
<tr><td colspan="3">测试环境与系统配置：
虚拟机 Android7.1.2</td></tr>
<tr><td>测试输入数据</td><td colspan="2">无</td></tr>
<tr><td colspan="3">测试次数：
执行测试过程 2 次</td></tr>
<tr><td colspan="3">预期结果：
删除某条账目后，查看个人信息页面的对应余额正确更新。</td></tr>
<tr><td colspan="3">测试过程：
打开 APP，创建一条选择账户的账单，在账单页面中长按删除，点击“我的”按钮前往个人信息页面查看余额是否正确更新</td></tr>
<tr><td colspan="3">测试结论：
符合预期</td></tr>
<tr><td colspan="3">实现限制： 无</td></tr>
<tr><td colspan="3">备注： 无</td></tr>
</table>

3.12 用例十二：新闻页面功能测试

这个测试用例的测试编号是 TestCase-FUNC-12，测试内容是测试新闻页面能否正常显示，同时所有的页面都按照需求有正确的显示。表 F-14 是这个测试用例的具体设计。

表 F-14：TestCase-FUNC-12 测试用例

<table>
<tr><td colspan="3">测试项目名称： Android 记账 APP</td></tr>
<tr><td>测试用例编号：TestCase-FUNC-12</td><td rowspan="2">测试人员：
宋依航</td><td rowspan="2">测试时间：
2020/11/23</td></tr>
<tr><td>测试项目标题：新闻页面新闻浏览功能测试</td></tr>
<tr><td colspan="3">测试内容：
测试新闻页面能否正常显示</td></tr>
<tr><td colspan="3">测试环境与系统配置：
虚拟机 Android7.1.2</td></tr>
<tr><td>测试输入数据</td><td colspan="2">无输入，点击“新闻”</td></tr>
<tr><td colspan="3">测试次数：每个测试过程做 1 次。</td></tr>
<tr><td colspan="3">预期结果：
正常显示新闻网页并可以进行浏览</td></tr>
<tr><td colspan="3">测试过程：
打开 APP，点击“新闻”按钮，查看是否能正常浏览新闻</td></tr>
<tr><td colspan="3">测试结果：
符合预期</td></tr>
<tr><td colspan="3">测试结论： 测试成功</td></tr>
<tr><td colspan="3">实现限制：无</td></tr>
<tr><td colspan="3">备注：无</td></tr>
</table>

3.13 用例十三：大量数据性能测试

这个测试用例的测试编号是 TestCase-Perf-1，测试内容是测试记录 100 条数据时软件的性能情况。表 F-15 是这个测试用例的具体设计。

表 F-15：TestCase-Perf-1 测试用例

<table>
<tr><td colspan="3">测试项目名称： Android 记账 APP</td></tr>
<tr><td>测试用例编号：TestCase-Perf-1
测试项目标题：大量数据的性能测试</td><td>测试人员：
李泽勤</td><td>测试时间：
2020/11/23</td></tr>
<tr><td>案例编号：CV-Perf-1</td><td></td><td></td></tr>
<tr><td colspan="3">测试内容：
记录 100 条数据时软件的性能情况</td></tr>
<tr><td colspan="3">测试环境与系统配置：
虚拟机 Android7.1.2</td></tr>
<tr><td>测试输入数据</td><td colspan="2">100 条符合要求的数据</td></tr>
<tr><td colspan="3">测试次数：每个测试过程做 1 次。</td></tr>
<tr><td colspan="3">预期结果：
软件能正常运行</td></tr>
<tr><td colspan="3">测试过程：
打开 APP，录入 100 条正常数据，观察软件运行性能。</td></tr>
<tr><td colspan="3">测试结果：
符合预期</td></tr>
<tr><td colspan="3">测试结论： 测试成功</td></tr>
<tr><td colspan="3">实现限制：无</td></tr>
<tr><td colspan="3">备注：无</td></tr>
</table>