寻宝猫系统详细设计报告

1 引言

二手交易需求始终在大学中存在，目前山东大学校内更多的是通过 QQ 群的方式进行交易，交易方式较为简陋，规则并不完善，对买家和卖家都造成了一定的影响。因此项目小组决定制作寻宝猫 APP，作为一种全新的二手交易方式，以求让学生享受更加便捷与规范的二手交易方式。

1.1 标识

本系统适用于 Android 以及 IOS 系统下安装及使用。

1.2 系统概述

寻宝猫是一个轻量级的二手交易系统,它提供接口以实现学生可以使用山东大学的统一认证平台进行登录，并查看或者购买系统中上架的二手物品，或者发布自身的二手物品到平台上。同时也提供聊天和对自身购买或者发布历史的管理功能。并且，也为系统管理员提供接口以维护网站的正常运行。

编写寻宝猫系统系统详细设计说明书的目的在于,从详细设计的角度明确寻宝猫系统项目的业务品种、功能范围，使系统开发人员和产品管理人员明确产品功能，可以有针对性的进行系统开发、测试、验收等各方面的工作。

读者：项目经理、概要设计人员，编码人员，测试人员。

1.3 文档概述

本系统用户分为普通用户与管理员用户，相应分为两个管理系统，普通用户在 APP 上进行二手物品的交易，本系统的交易功能如下：买家首先发起聊天，如果看中了商品，则发出购买的请求，并等待卖家的确认。 卖家如果长时间没有收到消息或拒绝了买家的购买请求，则交易中断。如果卖家同意卖出商品，则交易进入待确认状态，商品下架。此时平台会提醒卖家与买家进行沟通，商讨商品的交易方式，时间，地点 等。如果买卖双方完成了交易，并点击确认交易，则交易完成。并且，长时间未确认交易也会默认为交易完成。

管理员用户可以发布公告、强制下架违规商品、进行基础信息的修改，以及用户评级防止部分用户恶意下单。

1.4 功能需求

（1）用户可以在应用程序浏览商品；
（2）可提交订单、发起交易；
（3）具有购物车功能；
（4）用户能通过商品名称来检索商品；
（5）用户能查看当前订单和以往订单；
（6）管理员能查找客户信息；
（7）用户可以收藏商品；
（8）买家和买家可以进行聊天沟通

1.5 基线

1. 寻宝猫系统可行性分析报告 V1.0
2. 寻宝猫系统需求规格说明 V1.0
3. 寻宝猫软件架构文档 V1.0

2 引用文件

1.中华人民共和国国家标准 GB T-8567-2006

3 详细系统设计

寻宝猫二手交易平台有前台系统和后台系统两个功能，用户操作流程如图 5－1 所示。

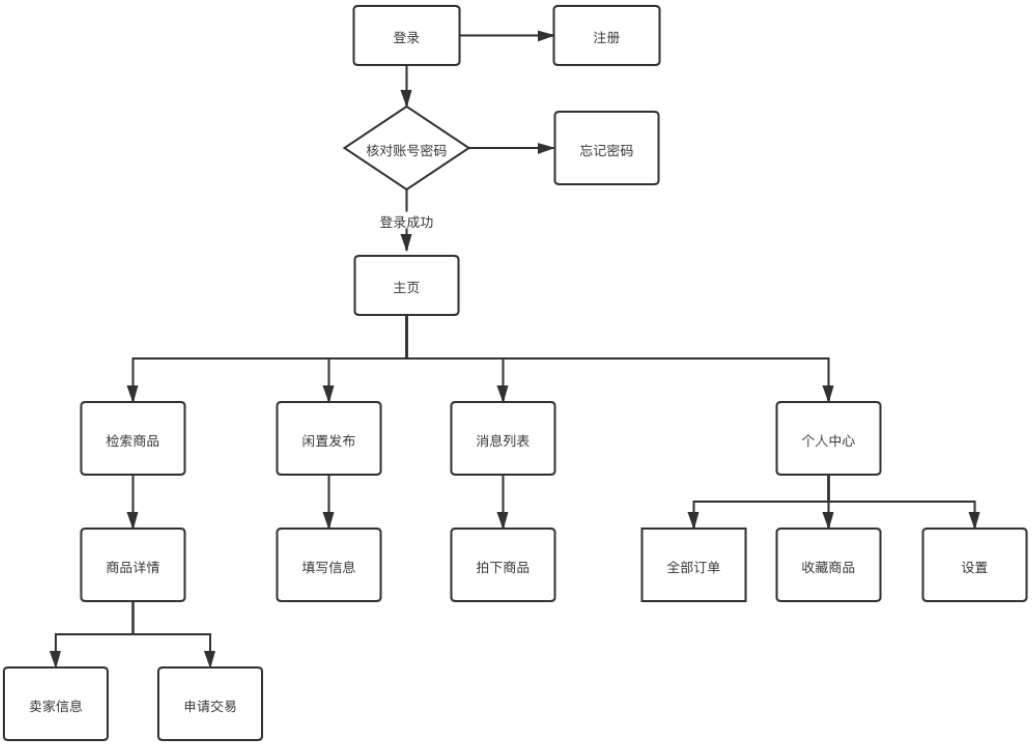

图 3－1 寻宝猫系统功能图

3.1 前台功能

（1）新用户注册：普通用户必须先注册，并绑定在校生身份认证，才能进行二手用品的信息检索、商品浏览、用户间交流等操作。

（2）二手物品分类搜索：用户可根据对目标用品的分类浏览进行浏览，并可以按照访问量、收藏量等对商品进行排序。同时，支持输入关键字、模糊查找进行本站所有的二手商品查找浏览。

（3）个人二手商品上架:用户可以发布自己的闲置物品用于交易，用户需要填写相关信息供买家进行分类检索。

（4）消息列表功能模块：买家用户可以向商品的卖家发送消息，询问商品信息或交易方式。同时，该功能模块可以直接引向商品拍下步骤。

（5）订单查询功能：客户确认购买后即生成一个唯一的订单号，客户可以在此功能模块查看自己所有状态的订单。

（6）前台页面管理：注册会员对自己资料的修改，账户的管理和商品收藏夹的管理。

3.2 后台管理

（1）管理员管理：网站管理人员实现对系统后台的管理，对系统所有信息的控制。还需

要对系统进行维护，同时还对普通用户和商家的基本信息进行管理。

（2）订单添加/删除/修改管理功能：对于用户提交的订单进行审核，并实现对订单的添加/删除/修改管理。

（3）个人信息管理：审核注册用户的在校生身份，并根据举报或反馈等管理用户权限。

3.3 交易主要处理内容

（1）页面模块化：系统界面的设计使用模块进行处理，如把页面的头、尾页面内容，数据库的连接等做成单独的文件,在其它页面设计中需要时可以把其他页面的相应的内容包括进去了，有利于页面风格比较统一以及提高开发系统的效率。

（2）二手商品信息模板的应用：商品信息代码的生成是基于模块的，更换商品信息内容时只要将商品新的一些必要的信息录入，然后自动修改商品信息的模板。

（3）功能较完善，管理方便：功能方面包括二手商品的展示、用户注册及登录、商品的在线查询、定购等各个方面，完整地实现了二手商品交易系统的管理要求，同时由于应用了模板，对系统的管理维护非常方便。

4. 数据描述

(1)商品信息表：

商品信息表中储存的是用户所上传的商品的基本信息，用于保存商品的相关信息，并用于软件的查询商品操作中。商品信息表的结构如表 4-1 所示：

字段	字段名	类型	备注
商品 ID	ItemID	int	不能为空，主键
商品名称	ItemName	string	不能为空
商品价格	ItemPrice	int	不能为空
商品类别	ItemClass	string	不能为空
新旧程度	ItemNewOrOld	int	不能为空
商品库存	ItemNum	int	不能为空
商品描述	ItemMsg	string	不能为空
商品图片	ItemImage	PNG 图片	可以为空
商品标识	ItemDisc	int	不能为空，外键

表 4－1 商品信息表

(2)用户信息表：

该表用于储存用户信息，由于在用户登录时，我们使用的是山东大学的统一认证接口，因此用户在登录时并不需要查询数据库，而是在创建，查询用户订单，和用户的个人中心中查询或显示用户的相关信息。用户信息表的结构如表 4-2 所示。

字段	字段名	类型	备注
用户 ID	UserID	int	不能为空,主键
用户名	UserName	string	不能为空
用户性别	UserClass	int	可以为空
用户手机号	UserPhoneNum	int	可以为空
用户头像	UserImage	PNG 图片	可以为空
用户标识	UserDisc	int	不能为空，外键

表 4－2 用户信息表

(3)订单信息表：

该表用于储存用户创建的订单，在创建订单，查询用户订单，管理用户订单时均需要用到。订单信息表的结构如表 4-3 所示。

字段	字段名	类型	备注
订单 ID	OrderID	int	不能为空，主键
订单生成日期	OrderDate	Date	不能为空，自动产生
下单人真实姓名	OrdererName	string	可以为空
总价	OrderPrice	int	不能为空，自动产生
订单标识	OrderDisc	int	不能为空，外键

表 4－3 订单信息表

(4)聊天信息表：

由于本系统中实现了用户之间的聊天功能，而用户的聊天记录是需要保存的。因此，需要建立聊天信息表，聊天信息表的具体结构如表 4-4 所示。

字段	字段名	类型	备注
聊天信息 ID	MsgID	int	不能为空，主键
聊天内容	Msg	string	不能为空
聊天信息标识	MsgDisc	int	不能为空，外键

表 4－4 聊天信息表

(5)公告信息表：

公告信息是以类特殊的聊天信息，因为它的接收方是全体用户，而如果使用聊天信息的格式存储，必然会使得存储的冗余数据过多。因此，这里单独存储公告信息。另外，由于有部分公告是限时公告，因此，需要通过字段标识出公告的这个性质。公告的状态有有效和失效两个状态。公告信息表的结构如表 4-5 所示。

字段	字段名	类型	备注
公告 ID	AllMsgID	int	不能为空，主键
公告状态	AllMsgStatus	bool	不能为空
公告内容	AllMsg	string	不能为空

表 4－5 公告信息表

(6)用户—商品联系表：

用户与商品之间是一个一对多的联系关系，一个用户可以上传多个商品信息，但是一个商品信息只可能关联一个用户。用户和商品信息共同唯一确定一个联系。另外，由于商品在系统中有不同的商品状态，不同的商品状态可以对商品进行的操作也不相同，因此，需要通过一个字段标识商品的状态，之所以不在商品表中存储状态信息，是因为个人认为在联系表中存储状态一则更贴合实际，二则在修改商品状态的时候不需要读取商品的其他信息，速度较快。根据之前的分析，商品的状态有：待出售，已被拍下和已出售三种状态。因此，定义用户—商品联系表的结构如表 4-6 所示。

字段	字段名	类型	宽度
用户标识	UserDisc	int	用户信息表的外键
商品标识	ItemDisc	int	商品信息表的外键
商品状态	ItemStatus	int	不能为空，自动产生

表 4－6 用户—商品联系表

(7)用户—订单联系表：

用户与商品之间是一个一对多的联系关系，一个用户可以上传多个订单信息，但是一个

订单信息只可能关联一个用户。与用户—商品联系表类似，订单也有不同的状态，将其放到联系表中的原因与上面的相同，这里不再赘述了。订单的状态有：待确认，已确认和已完成三种。用户—订单联系表的结构如表 4-7 所示。

字段	字段名	类型	宽度
用户标识	UserDisc	int	用户信息表的外键
订单标识	OrderDisc	int	订单信息表的外键
订单状态	OrderStatus	int	不能为空，自动确定

表 4－7 用户—订单联系表

(8)商品—订单联系表：

商品与订单之间是一个一对多的联系关系，一个商品同时可能有多个订单请求购买（因为商品的库存可以是多个），但是一个订单信息只可能关联一个商品。由于在之前的表中已经存储了订单的状态信息，并且，在用户层面，用户也是先看到订单，通过订单定位到商品，因此在查询时会先查询用户—订单联系表，再查询商品—订单联系表，因此这里没有必要再存储了商品的状态信息了。商品—订单联系表的结构如表 4-8 所示。

字段	字段名	类型	宽度
商品标识	ItemDisc	int	商品信息表的外键
订单标识	OrderDisc	int	订单信息表的外键

表 4－8 商品—订单联系表

(9)用户—聊天信息联系表：

用户与聊天信息之间是一个一对多的联系关系，一个用户同时可能有多条发送出去的聊天信息，但是一条聊天信息只可能由一个用户所发送。一条聊天信息由发送方，接收方和信息内容唯一确定，因此，用户—聊天信息联系表的结构如表 4-9 所示。

字段	字段名	类型	宽度
发送方标识	UserDisc1	int	用户信息表的外键
接收方标识	UserDisc2	int	用户信息表的外键
聊天信息标识	MsgDisc	int	聊天信息表的外键

表 4－9 用户—聊天信息联系表

5 程序设计说明

图 5－1 为本程序系统的实际的基本的处理流程。

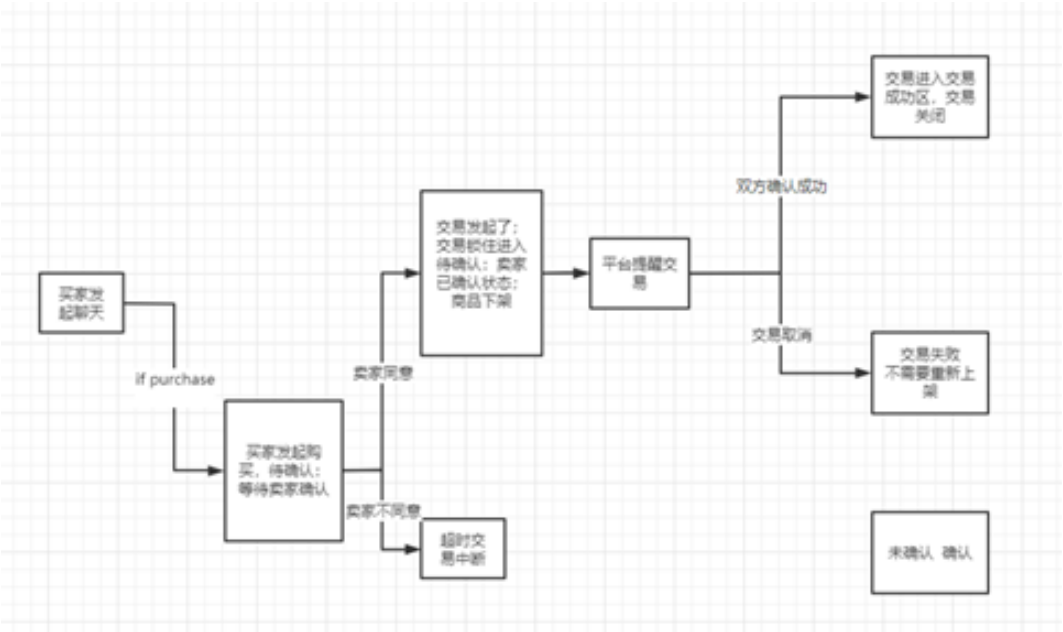

图 5－1 处理流程图

5.1 程序描述

（1）连接数据库模块代码

```
"use
strict";

          /* global Parse */

          import _ from "lodash";

          import moment from "moment";

          Parse.Cloud.define("agenda", function(request, response) {
            const Agenda = Parse.Object.extend("Agenda");
            const query = new Parse.Query(Agenda);

            function queryAgenda() {
              return new Promise(function(resolve, reject) {
                query.include("speakers");
                query.exists("day");
                query.find({ useMasterKey: true }).then(function(results) {
                  const json = results.map(function(result) {
                    return result.toJSON();
                  });
                  buildSchedule(json).then(function(formattedSchedule) {
                    resolve(formattedSchedule);
                  });
                });
```

```
    });
  }

  function buildSchedule(schedule) {
    let dayArrayLookup;
    const sortedSchedule = [{}, {}];

    return new Promise(function(resolve, reject) {
      _.forEach(schedule, function(session) {
        // Go ahead and add new time identifier to each session
        session.sortTime = moment(session.startTime.iso)
          .utcOffset(0)
          .format("X");
        session.displayTime = moment(session.startTime.iso)
          .utcOffset(0)
          .format("h:mma");

        dayArrayLookup = session.day - 1;

        // Check to see if timeblock exists on day
        if (!sortedSchedule[dayArrayLookup][session.sortTime]) {
          sortedSchedule[dayArrayLookup][session.sortTime] = [];
        }

        // Add to appropriate timeblock

sortedSchedule[dayArrayLookup][session.sortTime].push(session);
      });

      resolve(sortedSchedule);
    }); // Promise
  } // buildSchedule

  Parse.Promise.when([queryAgenda()]).then(
    function(results) {
      response.success(results);
    },
    function(error) {
      response.error(error);
    }
  );
});
```

（2）用户登录界面代码

```
Import
```

```
React
from
"react"
;
          import { connect } from "react-redux";
          import { skipLogin } from "../actions";
          import F8Colors from "../common/F8Colors";
          import F8Fonts from "../common/F8Fonts";
          import { Text, Heading1 } from "../common/F8Text";
          import {
            Animated,
            Dimensions,
            Image,
            StatusBar,
            View,
            TouchableOpacity,
            StyleSheet
          } from "react-native";
          import LoginButton from "../common/LoginButton";

          /* Config/Constants
          ================================================================================
          ==== */

          const SKIP_BTN_HEIGHT = 24,
            WINDOW_WIDTH = Dimensions.get("window").width,
            WINDOW_HEIGHT = Dimensions.get("window").height,
            VERTICAL_BREAKPOINT = WINDOW_HEIGHT <= 600,
            HEADER_HEIGHT = VERTICAL_BREAKPOINT ? 220 : 285,
            SKIP_BTN_MARGIN_TOP = VERTICAL_BREAKPOINT ? 15 : 23,
            WHENWHERE_PADDING_TOP = VERTICAL_BREAKPOINT ? 12 : 18,
            RENDER_ARROW_SECTION = VERTICAL_BREAKPOINT ? false : true,
            LOGIN_PADDING_BOTTOM = VERTICAL_BREAKPOINT ? 20 : 33,
            CONTENT_PADDING_H = VERTICAL_BREAKPOINT ? 15 : 20;

          /*
          ================================================================================
          ====
          <LoginScreen />
          --------------------------------------------------------------------------------
          -------

          Props:
            ?
```

```
================================================================================
==== */

class LoginScreen extends React.Component {
  state = {
    anim: new Animated.Value(0)
  };

  componentDidMount() {
    Animated.timing(this.state.anim, { toValue: 3000, duration:
3000 }).start();
  }

  render() {
    return (
      <View style={styles.container}>
        <StatusBar barStyle="default" />
        <View style={styles.header}>
          <Image
            source={require("../common/img/pattern-dots.png")}
            style={styles.headerPattern}
          />
          <Image
            resizeMode="cover"
            source={require("./img/illustration.png")}
            style={styles.headerIllustration}
          />
          <Image source={require("./img/logo.png")} />
        </View>
        <View style={styles.content}>
          <View style={styles.mainHeadingSection}>
            <Animated.View style={this.fadeIn(500, 5)}>
              <Heading1 style={styles.h1}>
                Facebook Developer Conference
              </Heading1>
            </Animated.View>
            <Animated.Text
              style={[styles.whenWhereText, this.fadeIn(1200, 10)]}
            >
              APRIL 18 + 19 / SAN JOSE, CALIFORNIA
            </Animated.Text>
          </View>
```

```
          {this.renderArrowSection()}

          <Animated.View style={[styles.loginSection, this.fadeIn(1900,
20)]}>
            <Text style={styles.loginComment}>
              Use Facebook to find your friends at F8.
            </Text>
            <LoginButton source="First screen" />
            <TouchableOpacity
              onPress={_ => this.props.dispatch(skipLogin())}
              style={styles.skipButton}
            >
              <Text style={styles.skipText}>SKIP FOR NOW</Text>
            </TouchableOpacity>
          </Animated.View>
        </View>
      </View>
    );
  }

  renderArrowSection() {
    if (RENDER_ARROW_SECTION) {
      return (
        <Animated.View style={[styles.arrowSection, this.fadeIn(1500,
15)]}>
          <Image source={require("./img/arrow.png")} />
        </Animated.View>
      );
    } else {
      return null;
    }
  }

  fadeIn(delay, from = 0) {
    const { anim } = this.state;
    return {
      opacity: anim.interpolate({
        inputRange: [delay, Math.min(delay + 500, 3000)],
        outputRange: [0, 1],
        extrapolate: "clamp"
      }),
      transform: [
        {
          translateY: anim.interpolate({
```

```
            inputRange: [delay, Math.min(delay + 500, 3000)],
            outputRange: [from, 0],
            extrapolate: "clamp"
          })
        }
      ]
    };
  }
}

/* StyleSheet
================================================================================
==== */

const styles = StyleSheet.create({
  container: {
    flex: 1,
    backgroundColor: F8Colors.bianca
  },

  //header styles
  header: {
    height: HEADER_HEIGHT,
    alignItems: "center",
    justifyContent: "flex-end"
  },
  headerPattern: {
    position: "absolute",
    left: 0,
    top: 0,
    right: 0,
    height: HEADER_HEIGHT - 30
  },
  headerIllustration: {
    position: "absolute",
    left: 0,
    width: WINDOW_WIDTH,
    bottom: 80
  },

  content: {
    flex: 1,
    justifyContent: "space-around",
    paddingHorizontal: CONTENT_PADDING_H
```

```
  },

  h1: {
    marginTop: 16,
    textAlign: "center"
  },
  whenWhereText: {
    marginTop: WHENWHERE_PADDING_TOP,
    textAlign: "center",
    color: F8Colors.tangaroa,
    fontFamily: F8Fonts.helvetica
  },

  arrowSection: {
    alignItems: "center",
    justifyContent: "center"
  },
  loginSection: {
    paddingBottom: LOGIN_PADDING_BOTTOM,
    alignItems: "center",
    paddingHorizontal: 20
  },
  loginComment: {
    textAlign: "center",
    fontSize: 15,
    color: F8Colors.pink,
    fontFamily: F8Fonts.fontWithWeight("helvetica", "semibold"),
    marginBottom: 23
  },
  skipButton: {
    marginTop: SKIP_BTN_MARGIN_TOP,
    height: SKIP_BTN_HEIGHT,
    alignSelf: "stretch",
    alignItems: "center",
    justifyContent: "center"
  },
  skipText: {
    color: F8Colors.colorWithAlpha("tangaroa", 0.5),
    fontFamily: F8Fonts.helvetica
  }
});
/* Export
================================================================================
==== */
```

```
module.exports = connect()(LoginScreen);
```

运行结果如下图 5—2 所示：

图 5—2 运行示例图

5.2 功能

初始化模块：

- 功能描述：系统初始时，由系统调用，经过身份验证，进入不同的模块。
- 输入项：用户名：学生的学号；密码：统一认证的密码。
- 输出项：有效用户和无效用户。

5.3 性能

要求登录快速，灵活，处理购买事项时不会出现死锁等数据库矛盾。买卖物品时相应迅速，服务器容量够大，能够承载较多用户的操作。

5.4 输入项

名称	标识	数据类型	输入方式	输入媒介	输入来源
学号	Student_number	Long long	用户输入	键盘	TextBox1
密码	Password	Varchar	用户输入	键盘	TextBox2
UserName	UserName	Varchar	参数传递	系统	Login.asp

表 5—1 寻宝猫系统登录输入项表

5.5 输出项

名称	标识	数据类型	输出方式	输出媒介
学号	Student_number	Long long	写入数据库	数据库
密码	Keywords	Varchar	写入数据库	数据库
UserName	UserName	Varchar	写入数据库	数据库

表 5－2 寻宝猫系统登录输出项表

5.6 算法

本系统在设计之初，并未设计算法，待以后迭代时，会设计以大数据人工智能为基础的推荐算法。

5.7 流程逻辑

网上购书系统的流程逻辑图如图 5－3 所示。

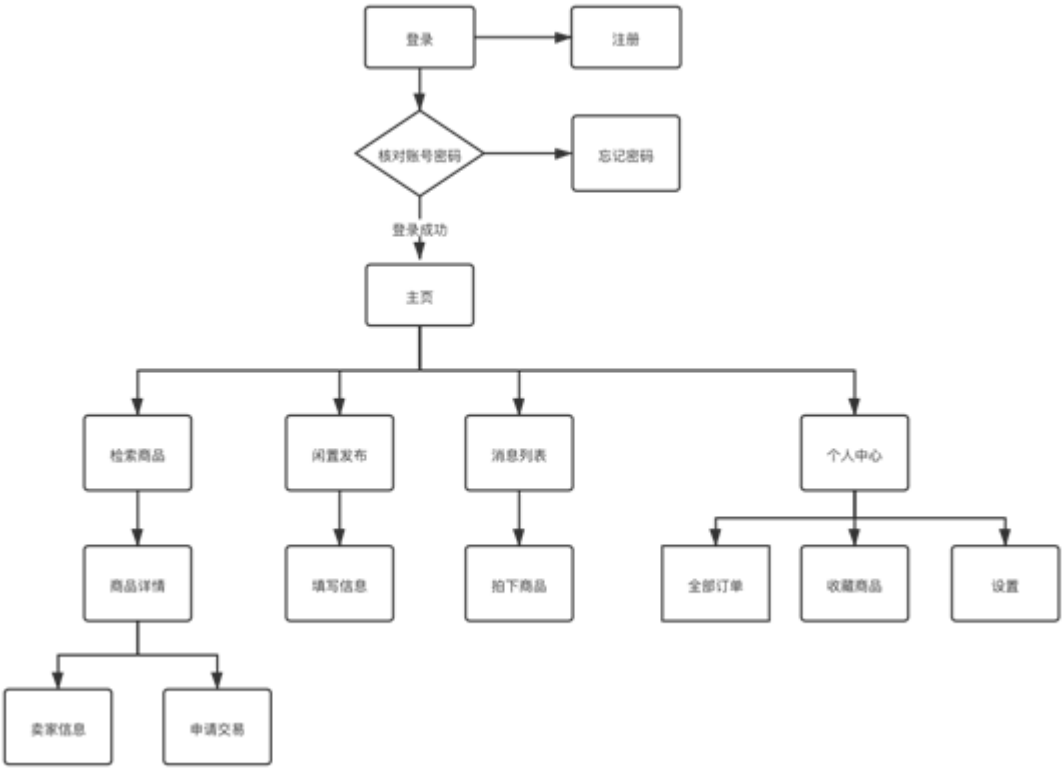

图 5－3 流程逻辑图

6 接口

用户信息、管理员信息及商品信息的接口图如图 6-1 和图 6-2 所示

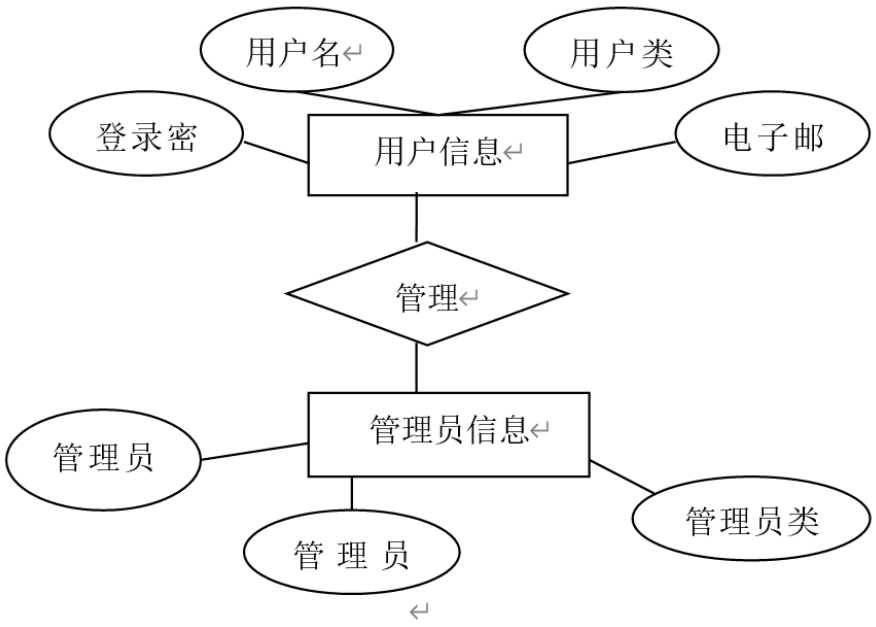

图 6－1 用户信息接口图

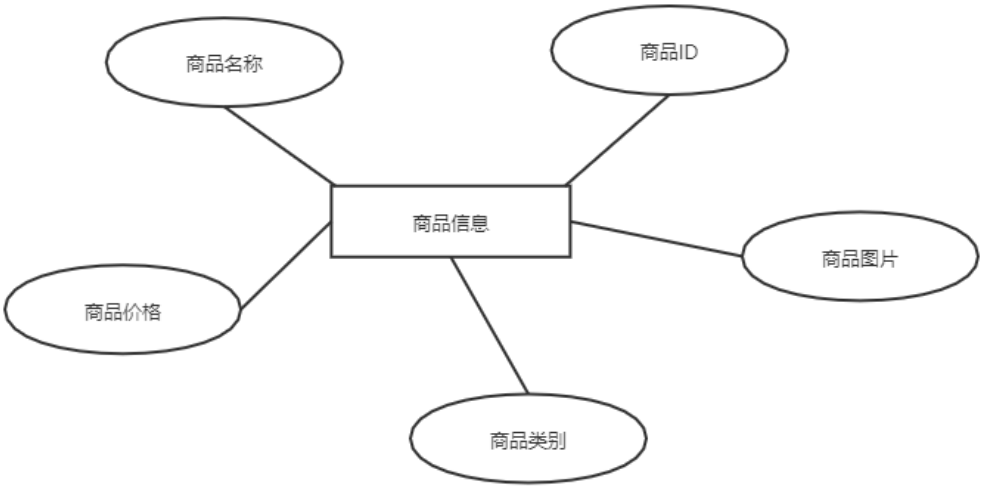

图 6－2 商品信息接口图

7 注释设计

说明准备在本程序中安排的注释，如：

a. 加在模块首部的注释；

b. 加在各分枝点处的注释；

c. 对各变量的功能、范围、缺省条件等所加的注释；

d. 对使用的逻辑所加的注释等等。

8 限制条件

系统只能在 Android 和 IOS 上运行，暂不支持网页登陆。

9 测试计划

9.1 计划执行的测试

9.1.1 测试目标

通过测试，达到以下目标：

1. 测试已实现的产品是否达到设计的要求，包括：各个功能点是否业已实现，业务流程是否正确。
2. 产品是否运行稳定，系统性能是否在可接受范围。
3. Bug 数和缺陷率是否控制在可接受的范围之内，产品能否发布。

9.1.2 测试方案

采用以黑盒测试为主，白盒测试为辅的测试方式，检查寻宝猫系统各模块的输入输出是否符合需求中的要求，并检查系统对异常情况的成熟能力。

9.1.3 测试条目

一、用户接口模块的测试

（1） 用户信息维护和测试

（2） 商品查询和测试

（3） 订购商品测试

（4） 聊天功能测试

二、管理员接口模块测试

（1） 商品信息维护和测试

（2） 用户信息维护和测试

（3） 公告发布测试

三、数据服务模块的测试

包括客户的查询订单信息的保存测试、订单处理数据的保存测试、销售情况的查询和分析数据的保存测试。

9.2 测试要求

步骤	动作	负责人	相关文档或记录	要求
1	打包、编译	开发人员	无	确认可测试
2	审核并提交测试	产品经理	审核报告	产品经理审核并签字
3	接收测试	测试负责人	接收任务单	确认产品有无重大缺陷、是否可以继续测试
4	执行测试	测试负责人	BUG 记录，测试总结报告	对产品质量进行评价

表 9－1 测试要求表

9.3 测试用例

模块名称	测试进度安排	测试目的	测试内容

基本数据输入（input）	系统完成后进行	测试是否达到系统的要求	输入简单的数据来测试
非法数据输入（error）	基本数据测试完成并通过后进行	测试系统对于一些非法输入数据的反应	输入一些特殊的字符或字符串来测试
空数据输入（zero）	可以和非法测试数据一起进行	测试系统对空白信息的反应	在所有数据项上输入空值来测试

表 9－2 测试用例表

9.4 测试进度表

测试阶段	测试任务	工作量估计	人员分配	起止时间
第一阶段	功能测试	2 日	1 人	2021.6.17-2021.6.18
第二阶段	界面测试	1 日	1 人	2021.6.18-2021.6.18
第三阶段	链接测试	1 日	1 人	2021.6.19-2021.6.19
第四阶段	兼容性测试	1 日	1 人	2021.6.20-2021.6.20
第五阶段	性能测试	2 日	1 人	2021.6.21-2021.6.22
第六阶段	测试总结	1 日	测试负责人	2021.6.23-2021.6.23

表 9－3 测试进度表

9.5 需求的可追踪性

暂停标准和再启动要求：

1. 冒烟测试，发现一级错误（大于等于 1）、二级错误（大于等于 2）暂停测试返回继续开发。
2. 软件项目需暂停以进行调整时，测试应随之暂停，并备份暂停点数据。
3. 软件项目在其开发生命周期内出现重大估算，进度偏差，需暂停或终止时，测试应随之暂停或终止，并备份暂停或终止点数据。
4. 如有新的项目需求，则在原测试计划下做相应的调整。
5. 若开发暂停，则相应测试也暂停，并备份暂停点数据。
6. 若项目中止，则对已完成的测试工作做测试活动总结。项目再启动时，测试进度重新安排或顺延。

9.6 测试风险

本次测试过程，受以下条件制约：

1. bug 的修复情况
2. 模块功能的实现情况
3. 系统整体功能的实现情况
4. 代码编写的质量
5. 人员经验以及对软件的熟悉度
6. 人员调整导致研发周期延迟

测试时间的缩短导致某些测试计划无法执行

10 尚未解决的问题

在软件逻辑设计方面还存在以下几个待优化问题：对于买家，想要查看需要的商品十分麻烦，需要通过查看以往的聊天记录进行搜索，同时 过长时间的消息无法确定商品是否还在，需要一一询问。 对于卖家，发售的商品过一段时间或许会被其他商品或者其他的水群

记录给“刷屏”，并且 部分以图片形式介绍的商品极其容易被人所忽略，因此出售物品也较为困难。 除此之外，交易群聊中人员有限，且群聊容纳人数有限，这就导致需求双方人数较少，因此交易也比较困难。

11 注解

11.1 定义

11.2 参考资料

[1]Abraham Silberschatz,Henry F.Korth,S.Sudarshan 著，《数据库系统概念》， 机械工业出版社

[2]Shari Lawrence Pfleeger,Joanne M.Atleez 著，《软件工程》，人民邮电出版社

[3]张海藩，《软件工程导论》， 清华大学出版社

[4]刘竹林编著，《软件工程案例开发与实践》，清华大学出版社

12 总结

寻宝猫系统是一个轻量级的二手交易系统,它提供接口以实现学生可以使用山东大学的统一认证平台进行登录，并查看或者购买系统中上架的二手物品，或者发布自身的二手物品到平台上。同时也提供聊天和对自身购买或者发布历史的管理功能。并且，也为系统管理员提供接口以维护网站的正常运行。

寻宝猫希望能够构建一个山东大学校内二手交易平台,改变之前只能够通过二手群交易的方式，将交易模式规范化，商品查询便捷化，便于同学们进行二手物品的交易以及各种委托的发布。同时，交易方式采取线下交易的方式，保证交易的安全性。

寻宝猫平台是一个比较大的系统，它涉及到购物流程和商品管理等，在这次设计中，小组完成了用户注册、登录，书籍信息显示、购买，生成订单等基本功能。通过这次设计，对 React-Native 和 MySQL 技术有了更深一层的认识和应用，取得了更大的进步。