
西南民族大学

本科生毕业设计(论文)

基于 Android 系统的网约导游 APP 的设计与开发
Design and Implementation of Online Hailing Guide APP Based on
Android

教学单位 计算机科学技术学院

姓 名 佟哲

学 号 201631109053

年 级 2016 级

专 业 物联网工程

指导教师 谢盈

职 称 副教授

2020 年 5 月 4 日

摘要

随着我国 GDP 的不断上升以及国民生活水平的持续提高，旅行越来越成为多数人的节假日休闲的不二之选。但随着旅游业的蓬勃发展也衍生出许多问题，究其根源还是旅行社平台本身的弊端，例如无法做到灵活地传递信息，无法信息透明化，无法准确获取用户需求等，导致出了一些对导游、对客户的价格不透明问题。基于以上原因，本文设计并开发了一款线上网约导游 APP “旅猫导游宝”，目的是在导游和客户之间建立沟通和交互的桥梁。

当前移动设备可以说是完全的与现代人的生活融合在一起，并且开发基于 Android 的 APP 不仅可以节省开发成本，又拥有广大用户群体，因为 Android 作为当前应用最为广泛的开源手机系统，开发基于 Android 系统的 APP 相较于其他环境更易于 APP 的推广和应用。

目前的旅行类系统主流旅行取向仍为定制路线或旅行贴士，并未针对大部分用户群体提供针对性的解决方案。本文设计并实现的线上网约导游 APP “旅猫导游宝”将针对大部分在导游选择上心怀顾虑的用户群体，使得用户能雇佣到合适的导游为其量身打造真正的个人出行方案，让假期真正地放松。“旅猫导游宝”系统为用户提供注册登录功能，同时对于游客用户，完成了用户可以根据景点下单，多导游接单后供游客进行选择确定接单导游的功能，实现了订单完成后的评价功能，以及对导游用户的，导游资格认证功能以及查看订单及接单功能。最后经测试表明，系统实现所有预期功能，能够稳定运行。

关键词：旅游；导游；Android；APP；

Abstract

With the develop of the citizen's standard of living, more and more people choose to take a journey as the leisure activities in holiday. But it also brings some negative social phenomenon like some tourist can't trust the tour guide from travel agency. In my opinion, it's mainly because the travel agency can't deliver the message flexible in public, since it's self-interest may being involved. It's also leads to some guides are treated unfair but they can do nothing to change. So that design and development this APP which named 'Kitty Guider' to unite the travelers and guides is meaningful.

Nowadays mobile devices are popular in citizens, nobody can live without a smartphone. And because of the popular of Android system, design and development the APP based on it can not only save the money, but also easy to generalize.

Different from some kinds of excellent travel APP in service, which glad to make route for the user, I tend to provide a new choice for those who worry about the quality of the travel agency's service. It can help them hiring a suitable guide who can make the best suitable plan can help people enjoy the whole vacation easily. This APP provide the register and login functions, and the functions of placing an order, choosing the guide and evaluating the order for travelers, as well as the functions of order receiving and qualification authentication for guides. At last, this program can work well as the wishes after testing.

Keywords: Tourism; Guide; Android; APP;

目 录

第 1 章 绪论.....	1
1.1 选题背景	1
1.2 国内外研究现状	1
1.3 本文的组织结构	2
第 2 章 相关技术介绍.....	3
2.1 Java 语言.....	3
2.2 Android 系统.....	3
2.3 PHP 脚本.....	4
2.4 ThinkPHP 架构.....	4
2.5 MySQL	5
第 3 章 需求分析.....	6
3.1 系统可行性分析	6
3.2 功能需求分析	6
3.3 性能需求分析	8
第 4 章 系统设计与实现.....	9
4.1 总体设计	9
4.2 详细设计	9
4.3 数据库设计	19
第 5 章 系统的测试.....	22
5.1 测试流程及结果	22
5.2 测试结果分析	30
第 6 章 总 结.....	31
6.1 工作总结	31
6.2 展望	31
参考文献.....	32
致 谢.....	33

第 1 章 绪论

1.1 选题背景

随着社会生产力进步像是农产品的自给自足或是工业上的大量国际订单，社会居民的生活水平也由此在不断地提升，在满足了温饱等基础需求的情形下，人们开始追求温饱之外的精神生活，由此，旅游产业应运而生。人们在工作之余进行短途或长途的旅行，一方面丰富了人生阅历，体会到异地的风土人情，另一方面也使得长期紧绷的神经得到了放松。可是发展永远会伴随着一些问题的出现，举个例子，随着各地旅游业的发展，除了美丽的风光和特色的民俗之外，黑导游恶导游的事件也层出不穷。而人们在挑选旅行社的时候可能根本得不到自己本次旅行导游的一些专业信息，故而会对自己旅途的导游素质感到担忧，一旦旅行陷入了与导游的纠纷之中，那基本上这次的旅行也就失去了最初的意义。此外，有些行动派或是出差到异地的人们到达一个陌生的地方，只是需要一个方便的短期导游，而这时候他们如果去求助旅行社就显得太过麻烦。并且笔者注意到市面上虽说有很多旅行 APP，并且大部分都推出了旅游路线规划的服务，但是相较于专业导游的服务，这种旅游路线规划服务对于用户来说还是不够周到。

而对于导游来说，虽然作为旅行社的主要成员，却并不能完全的发挥自身的才干，大部分的工作和订单都要等待旅行社的指派。同时导游也无法同游客进行下单前的沟通，故而在旅途中导游只能进行单向的输出，实际上有时导游的指导和讲解并不能迎合或者满足游客，甚至有时会因与游客意见相左而产生冲突。这都是由于旅行社作为中间的信息传递平台却缺乏灵活性所导致的不良后果。为了利益的最大化，从双方抽取中介费是必要手段，这不仅拉高了游客出行的门槛，同时也是一种对于导游的剥削^[1]。

当下移动通讯设备已经完全融入现代人的生活之中，可以说人手一个手机，仅我国手机用户便以逾 9 亿。而 Android 系统基于其开放的空间和模式，有着更广泛的用户基础，在这样的条件下，开发基于 Android 系统的线上导游预约 APP 将会更灵活地解决这些导游与游客的问题。相较于普通旅行社的电话或线下咨询，使用手机一键下单，既迎合了当下人们的使用习惯，使人们以更便捷的方式完成出游前的准备，同时也减轻导游被旅行社所制约的情况。基于此背景，本课题将通过设计一款 APP 来为用户提供更灵活、更直接的导游服务。

本程序的开发设计主要为构建一个基于移动端的网上预约导游平台，无论游客用户还是导游用户都能够使其利用随身携带的 Android 手机随时随地访问使用本程序。“旅猫导游宝”系统设计包括了客户端和服务器端两部分。而本文的主要工作是要对整体框架进行简单的分析，以及重点描述该 APP 客户端的技术设计和开发。

1.2 国内外研究现状

随着移动通讯技术比如移动设备的不断进步和发展的这些年，持有移动设备的用户在

全世界范围内占比超过 60%，可见其成长速度之快^[2]。在旅游行业 APP 中，主流的服务还是提供旅行建议、旅行线路规划等，在网约导游这部分还有相当大的市场。同时在共享经济的时代浪潮中，一些像叫车、购物甚至在线订餐等方式都早已经成为人们的生活习惯，很大程度上取代了过去打电话进行预定的麻烦方式，像滴滴专车、神州出行、淘宝、美团、去哪儿等软件也均成为人们手中的必备品，为人们的生活提供着服务。这些程序能够无时无刻的为用户提供位置信息等实用技术，^[3]。其于 2010 年 4 月宣布免费开放其 API，开发者能够通过使用其所提供的接口，在百度地图的基础上进行自身的二次开发，发布更多具备不同功能的地图应用^[4]。Android 开发者能够使用由 Android 提供的开发库，开发者可以根据需求不同随意使用地图，安卓支持 GPS、网络地图、地理定位服务的 API，而且也支持由第三方的包，借助这些包和手机的 GPS 定位以及开放的地图 API，能实现一系列的位置服务功能^{[5][6]}。

同时从旅游业来讲，政府积极推进导游自由职业化，所以类似滴滴专车这种网约车模式的网约导游也必将成为一种新兴的旅行模式。其实网约导游这个概念早在 2016 便被提出^[7]。参考国家旅游局发布的公告，大量的数据表明跟团旅游的市场萎缩是完全可以预见的。可见随着时间的推进以及网约导游理念的增强，将会有越来越多的人选择这种“互联网+导游”的服务模式。

1.3 本文的组织结构

本文将研究的关注点放到在线导游预约 app 上来，结合当前导游市场的现状和互联网的发展，以 Android 系统技术为纽带，探索这种网约模式的优点。文章一共分为六章：

第一章是绪论部分，首先对背景和研究现状进行简略的介绍，接着论述了该文章的研究目的和论文结构

第二章是开发环境及技术介绍，在这一章节里介绍了开发过程中所采用到的开发技术，其中着重介绍了 Android 这种操作系统，Java、PHP 等开发技术以及混合开发的开发思维。

第三章是需求分析，首先对导游预约程序所需要的经济技术进行可行性分析，紧接着对“旅猫导游宝”所需要实现的功能进行了需求分析，一一地罗列出了该“旅猫导游宝”系统将会实现的功能。

第四章是系统设计，也就是系统的编码实现部分。根据需求分析介绍了系统开发的设计流程，包括服务器的设计实现、客户端功能设计以及数据库的设计。

第五章是系统测试的介绍，这部分根据系统功能给出了详细的测试数据以及结果，并且将设计过程中所出现的问题进行了记录和总结。

第六章是总结章节，在这一章节针对本次设计进行了总结，并且分析了本次研究的不圆满之处，也确定了将来需要改进和补充的工作。

第 2 章 相关技术介绍

2.1 Java 语言

从开源和用户群体这两个角度出发进行 Android 开发自然不能缺少 Java 编程的能力，Java 技术作为一门编程语言和开发平台，基于 Java 虚拟机技术，具备着跨平台的优势，其高兼容性在软件开发领域处于很重要的地位，广受大批程序员的追捧^[8]。Java 编程语言兼具编译型以及解释型的语言的特点，程序运行可以实现一次编译，多次解析^[9]。该语言是一种分布式的面向对象语言，具有面向对象、程序的高移植性、易学性、解释执行、多线程、健壮性等特点。Java 是一种面向对象的语言，对对象中的类、继承、对象、封装、接口、多态、包等都可以提供很好的支持。虽然在类与类之间只支持单继承，但可以使用接口实现多继承。同时由于虚拟机机制，Java 程序表现了很强的可移植性。Java 语言支持多线程，还提供多线程间的同步机制，每个线程都有自己的 run()方法。Java 的异常处理、垃圾回收机制等也都为其健壮性提供了强有力的保障。最后 Java 还提供安全机制来防范被恶意代码攻击。

2.2 Android 系统

而说回到 Android 系统。它是基于 Linux 的平台，其组成包含四个部分包括 OS、APP、中间件和用户 UI，是一个开放以及专门搭建移动端的系统^[10]。

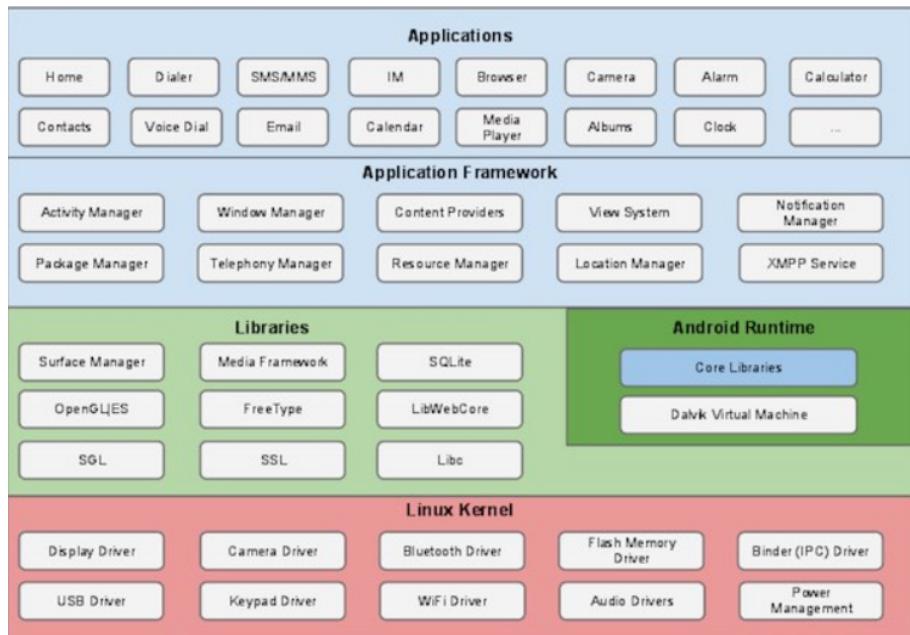


图 2.1 Android 架构图

Android 操作系统架构图如图 2.1 所示，它是一个软件组件的栈。

Android 采用软件叠层（software stack）作为其架构，该架构分为三部分，即底层操作系统、用户界面以及中间件和应用程序软件。底层只能提供基本的功能，以 Linux 内核，编程语言是 C 语言；中间层又分为库函数和虚拟机部分，该层可以为开发者提供 C/C++ 的

语言开发方法；最上层面向所有开发设计者，包括各种 APP，这一层主要支持 Java 语言进行编写。简洁地讲 Android 即是运行在 Linux 内核上的基于 Java 的操作系统。

而本次我使用的 Android 开发工具是 Android Studio，其为 Google 公司基于 IntelliJ IDEA 推出的集成开发环境，并且其提供了丰富且大量的工具辅助开发者进行开发，比如各种性能测试工具、网络监控工具等。Android Studio 同时也为开发者提供了功能丰富的模拟器以及强大的布局编辑器，为开发者提供了很大的便利。

基于 Android 后台机制的处理模式，即每个应用程序都在后台建立一个独立的进程，虽然牺牲部分内存，但也提升了应用程序运行的稳定性。当然稳定性并非其最大特点，Android 系统的最大优势在于其开放性：首先所有的 Android 开发者都可以加入 Android 联盟当中，其次广泛的硬件兼容也是其开放性的一大体现，这为 Android 提供了加大的平台，同时手机厂商也不再为框架所束缚。

故而，尽管使用原生开发也就是 Java 以及 Android Studio 的开发工具对终端进行 Android 开发这种开发形式有着其独特的优点，比如其可实现功能齐全，可以访问所有手机功能。同时运行速度快、反应速度快，支持大量图形处理不卡顿。不过其也存在一些缺点，比如开发时间过长以及成本较高、可移植性较差等，所以，因为 Android 自己给出了一定的开发接口，让更多的开发者可以参与其中，在 Android 这种高兼容性的特点的基础上，我们在使用 Java 语言混合使用 PHP 语言对该 APP 进行整体的开发设计。

2.3 PHP 脚本

PHP 作为一种面向对象的主要应用于服务器端通用的开源脚本语言，其语法混合了多种语言包括 C、Java 等的语法以及部分独创的 PHP 语法。PHP 语言是可以免费使用的，同时其语法与 C 语言较为相似，面向对象的概念使其语法较于 C 语言麻烦的地址操作显得更加简洁，所以其更易上手操作以及学习。同时 PHP 语言也可以很容易的与市面上流行的部分数据库兼容，所以基于经济、上手难度，我将其作为我们后台开发的选择。

这种混合开发的优点也很直观，首先是本人对于使用 PHP 后台开发有一定操作经验和基础，这也是我选择采用其进行后台开发的主要原因，同时这种混合开发也有着其天然优点，相较于 Java 语言，PHP 更加直观以及简单，使用其对后台进行开发有着降低开发成本、缩短开发周期等优点，这样也就达成了符合技术主流和可行性的条件，故而这种混合开发被我锁定为最优的开发选择。

2.4 ThinkPHP 架构

对于开发者而言，因为有框架的 MVC 支持，基于 ThinkPHP 开发，开发者只要把精力集中在控制器、视图模版、和模型的开发上，这样可以加快开发速度，而且系统分层清晰，各层各司其职，耦合非常小。ThinkPHP 的 MVC 支持架构如图 2.2 所示。

同时本着简单快速的原则，我们的后端架构选择了 ThinkPHP 架构，这是一个快速、兼容且简单的国产 PHP 开发框架。这个框架为我们提供了丰富的查询语言支持、自动创建

目录的功能、分布式数据库支持、灵活简单的项目配置等，总的来说开发速度很快，最符合我们的开发需求。当然其也有着不可避免的劣势，比如过于死板，像是 TP 默认初始化了很多配置，使用起来很方便，但自然也会影响效率。不过这并不影响我们的开发，所以其实完全可以将其忽略不计。

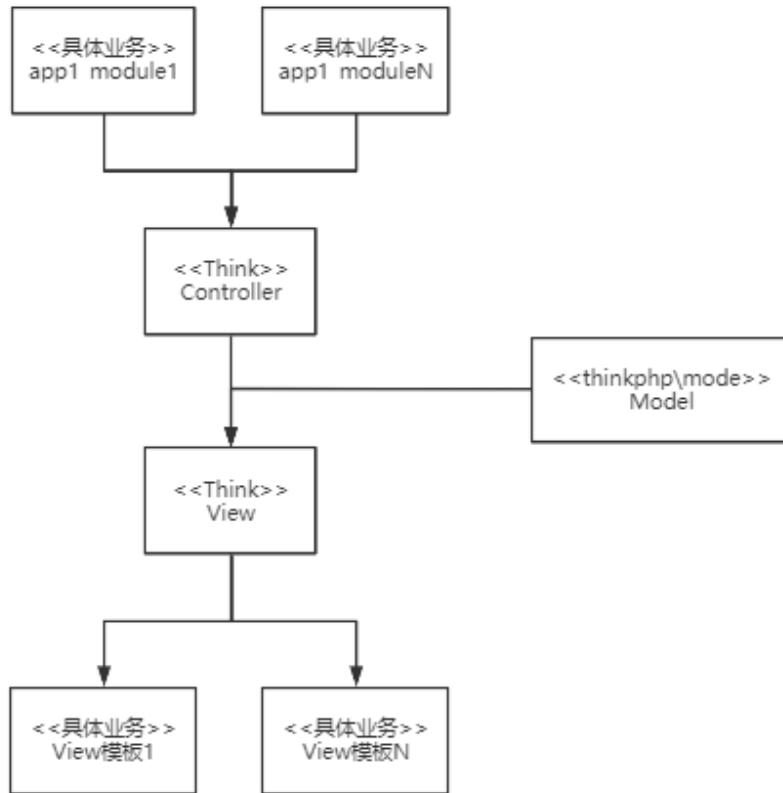


图 2.2 ThinkPHP 的 MVC 支持架构图

2.5 MySQL

最后为数据库开发工具—MySQL，其原为一个开源关系数据库管理系统，由 MySQL AB 公司开发，后被 SUN 公司收购。2009 年 Oracle 公司将 SUN 公司收购，MySQL 正式成为 Oracle 公司旗下产品^[11]。

MySQL 为体积小、成本低、速度快以及免费等原因，一般中小型网站开发都选择 Linux 加 MySQL 作为网站的数据库^[12]。当然这也是我们此次开发选择其作为我们数据库开发工具的原因。

MySQL 的源代码使用 C/C++ 进行编写也就因此具有很高的可移植性，并且还使用多种编译器进行测试，这些都为其提供了保证。同时 MySQL 为大部分的编程语言比如 C、C++、Java、PHP、Python 等都提供了应用程序接口。MySQL 支持多用户同时访问、支持同时运行多个线程来完成不同的工作，对中央处理器资源可以达到充分利用。其本身具备管理检查以及优化操作的工具并且可以处理拥有上千万条记录的大型数据库^[13]。

以上便为本次设计开发所涉及的主要开发环境及技术。

第 3 章 需求分析

3.1 系统可行性分析

由于经费及技术限制，很多问题并不是说提出就一定可以得到圆满的解决方法，这时候在开发前进行可行性分析就显得尤为重要。同时，预先进行可行性分析也可以将开发中所将有遇到的问题进行一定程度上的预估，同时也可以很大程度上缩减开发中所花费的时间以及经费成本。故可行性分析这部分本论文计划从技术、经济两方面进行简单分析。

首先本 APP 灵感来自滴滴专车的网约车服务，部分功能对标网约车软件，所以技术层面来讲开发实现该 APP 是存在理论可行性的。而本次开发所选用的开发环境为 Android 平台，而 Android 平台的一大特点便是开源及兼容。当下主流的移动端平台便是苹果公司的 IOS 操作系统以及谷歌公司的 Android 操作系统。苹果的创新具有限定条件，这种天然约束归咎于其自我封闭性。而 Android 是开源的，允许第三方修改，这在很大程度上容许第三方厂家根据自身需求来更改版本，来适应自身选用的硬件，相较于苹果的封闭，Android 这种开源给开发人员提供了更广阔的创新空间，故而 Android 版本升级更快，并且流畅度并不比 IOS 差，其 UI 定制性强、系统完全开源，对于 APP 开发设计方面功能稳定而且简洁^[14]。故而选择 Android 不仅降低了我们开发难度，同时也节约了我们的开发成本。同时服务器端开发选用 Think PHP 框架进行的搭建。

开发该网约导游 APP 经济预算也是非常低的，以我个人能力便可轻松承担，故可以说完全具备开发的经济可行性。首先开发选用的软件包括 Android Studio 等大部分均可在网站上免费下载，所以在软件方面经济支出基本为 0，并且在开发过程中也并不需要雇佣他人作为帮手，仅本人便可完成整个开发流程^[15]。

3.2 功能需求分析

3.2.1 应用场景描述

该 APP 客户端主要分为 3 个模块：

1. 登录注册模块：在登陆界面游客用户以及导游用户需要输入用户的账号和密码，以及选择是游客用户或者导游用户，若账号密码正确则登陆成功进入主界面；若账号密码不正确则登陆失败。注册界面用户需填写必要信息如账号（手机号）、密码、姓名和个人描述。注册成功后登陆进入主界面。

2. 订单模块：对于游客用户，此模块为用户提供搜索地点附近的景点推荐，用户可根据景点和地区进行导游预约下单，下单后进入等待接单状态；对于导游用户来说，此模块为用户提供接单的界面，在这个界面导游用户可以查看处于等待接单的状态的订单并进行接单操作。当导游接单后，游客用户再登录订单详情页面进行对接单导游进行选择。

3. “我的”模块：为用户提供查看自身资料及退出登录的功能。并且对于导游用户，可以在“我的”模块中对自身导游资格进行认证申请。

下图 3.1 及 3.2 分别为游客用户以及导游用户的用例图：

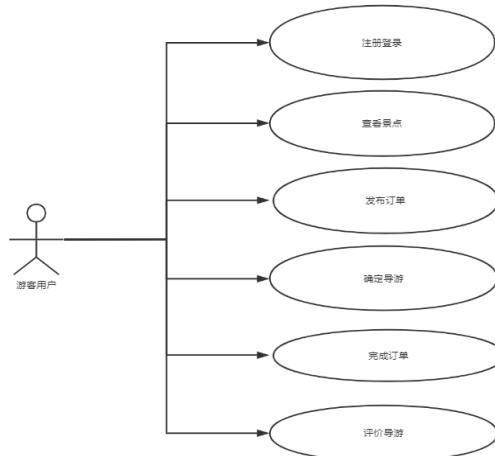


图 3.1 游客用户用例图

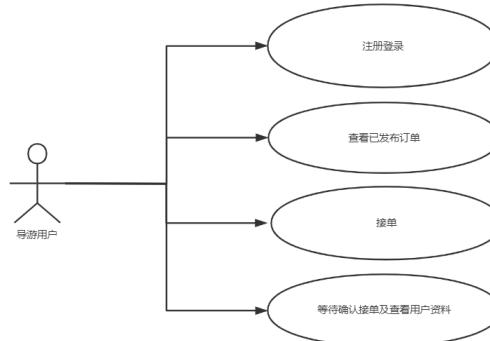


图 3.2 导游用户用例图

3.2.2 登录注册模块

当用户首次打开本系统，首先其操作将为注册。而在这一模块，游客用户与导游用户的操作并无太大差别。当用户完成注册，将注册成功的账号正确输入登录界面，便可完成登陆操作。登录模块的用例图如图 3.3 所示：

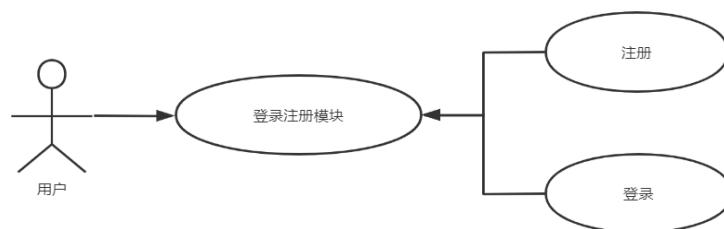


图 3.3 登录注册模块用例图

故而该模块将满足用户对于注册及登录功能的需求。

3.2.3 订单模块

用户登陆本系统后，首先进入的首页便为订单模块的一部分。首先对于游客用户，

在首页会提供搜索景点的功能，并在搜索到景点后可以完成下单操作。当游客用户按提示正确完成下单操作后，导游用户可以在导游首页浏览未接单订单并按自身需求进行接单申请。当有导游进行接单申请后，游客用户可以在订单详情页查看订单接单导游个人信息并依据自身需求对接单导游进行选择。最后双方完成交易，游客用户可对订单进行评价。订单模块用例图如图 3.4 所示：

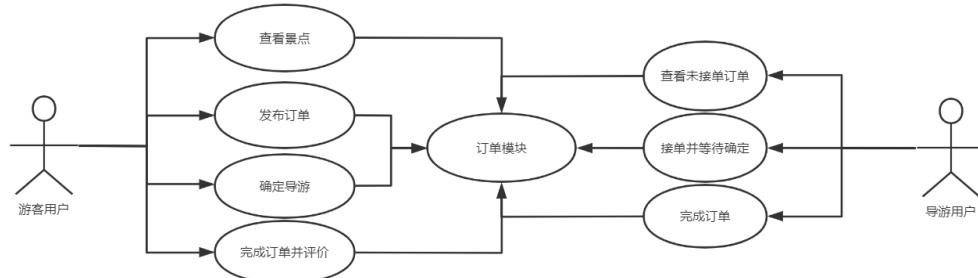


图 3.4 订单模块用例图

3.2.4 “我的”模块

用户可以在“我的”模块中对自身资料进行查看。导游用户的导游资格认证操作也在这个模块中实现：导游上传自身导游凭证，在后台审核后获得导游认证。“我的”模块用例图如图 3.5 所示：

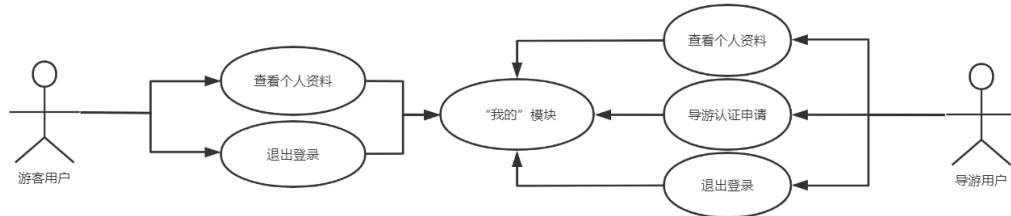


图 3.5 “我的”模块用例图

3.3 性能需求分析

在数据类型上，系统除支持一般结构性事务数据即可，故而对于大数据量的结构化数据的存储和查询数据量支持能力需求也并不高。其次在数据库需求方面，根据本系统数据的特点，将采用标准 MySQL 语句，以便将来的扩展和移植。并且对于数据库有以下具体性能要求：首先是独立性强，对系统结构影响比较小，且可在多种操作系统、服务器下运行；其次支持高级语言，支持汉字；支持主流的网络协议；具有一定容错能力。

在数据库访问并发性上，数据库需要支持超过多用户的并发访问能力。而在服务器管理端的并发性上，服务器平台需具备多人访问操作的能力。

一般数据查询响应时间<5 秒。

系统可以提供 24 小时不间断的服务，需要具备兼容各大主流浏览器的能力。

第 4 章 系统设计与实现

4.1 总体设计

如图 4.1，此为该系统总体设计架构图

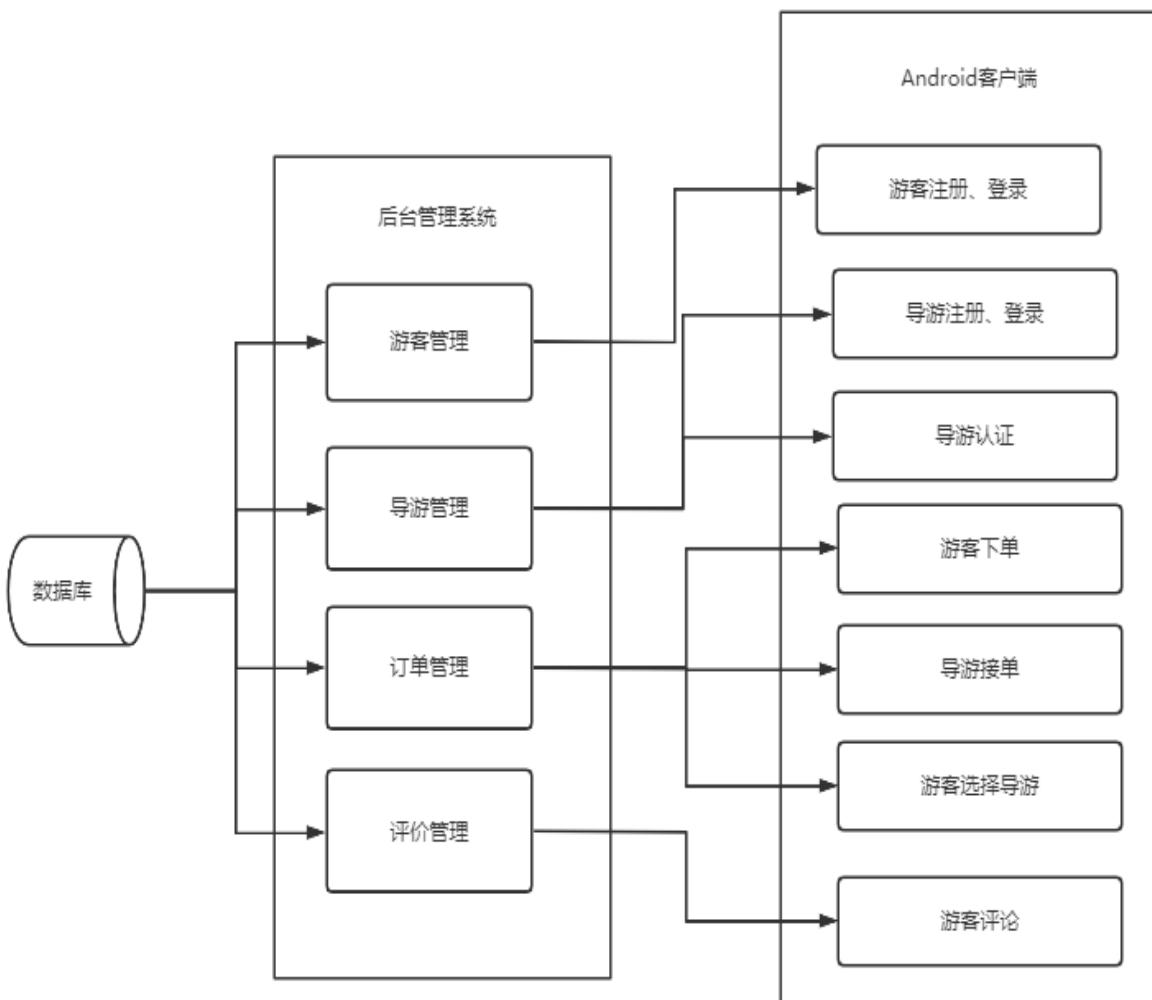


图 4.1 总体设计架构图

4.2 详细设计

4.2.1 登录注册功能模块

该 APP 的客户端用户分为两类：即游客与导游。当游客使用该程序时，首先打开 APP，直接跳转到用户登录界面，若用户为首次登陆尚未注册，则需要点击下方注册按钮跳转到注册页面。而导游步骤与游客大致相同，除了需要在注册页面选择注册导游账号，后面同样的登陆与注册步骤。不过在注册成功后导游还需等待后台对其他资格认证结束，认证成功才能显示为认证导游。当登陆成功后，用户便可以跳转到此 APP 的主界面。注册时用户

的活动图图如图 4.2 所示：

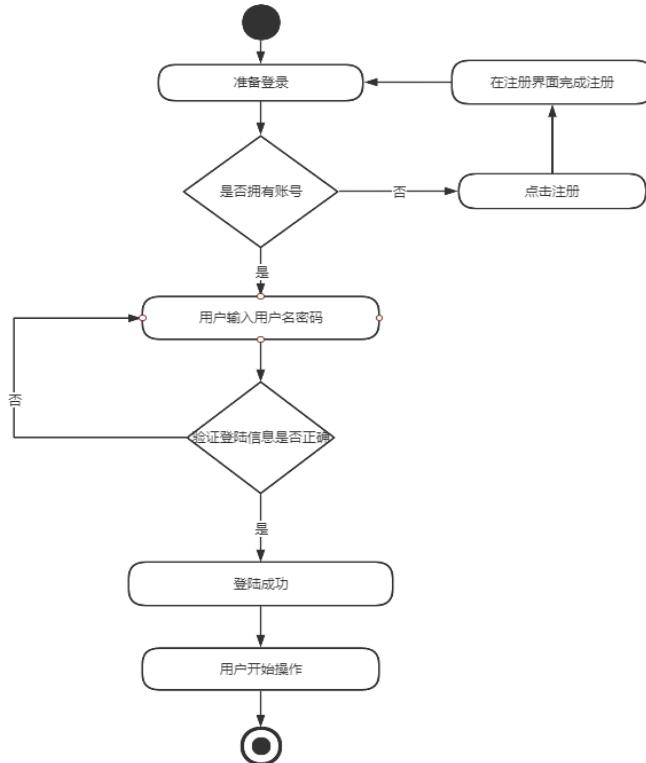


图 4.2 用户登录流程图

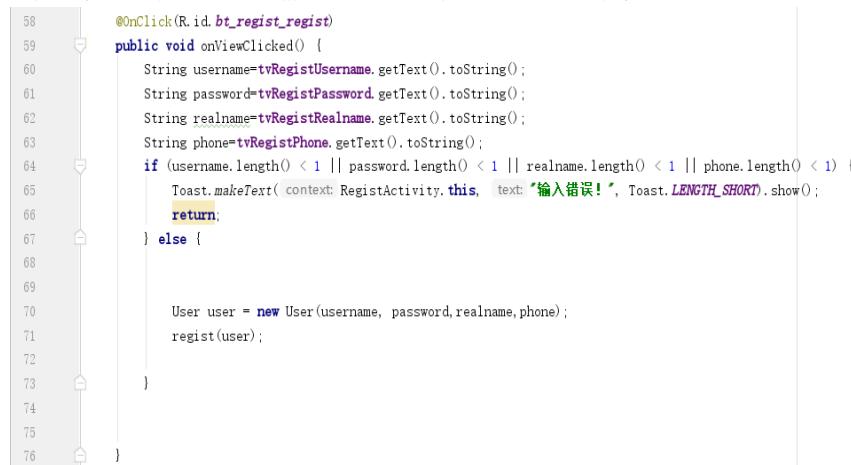
登录注册功能模块主要用于实现用户的注册登录需求。用户在注册结束后可凭该账户进行登录操作。

如图 4.3 所示,这是用户登录界面的 layout 文件,登陆界面下较醒目位置设计为该 APP 的名称,下方为用户账号及密码的输入框,使用的是 Edit Text 作为文本框、Radio Button 作为切换控件、Button 作为登录、注册按钮。

```
24 <TextView  
25     android:layout_width="match_parent"  
26     android:layout_height="100dp"  
27     android:layout_marginTop="1px"  
28     android:text="猫咪导游宝"  
29     android:textSize="30sp"  
30     android:textStyle="bold"  
31     android:gravity="center"  
32     android:background="@color/snow" />  
  
33 <LinearLayout  
34     android:layout_width="match_parent"  
35     android:layout_height="wrap_content"  
36     android:orientation="vertical"  
37     android:layout_marginTop="30px">  
38     <EditText  
39         android:id="@+id/tv_login_username"  
40         android:layout_width="match_parent"  
41         android:layout_height="wrap_content"  
42         android:hint="账号"  
43         android:text=""  
44         android:inputType="text"  
45         android:maxLines="1"  
46         android:singleLine="true" />  
47     <EditText  
48         android:id="@+id/tv_login_password"  
49         android:layout_width="match_parent"  
50         android:layout_height="wrap_content"  
51         android:hint="密码"  
52         android:text=""  
53         android:imeOptions="actionUnspecified" />  
54 </LinearLayout>  
55 <RadioGroup  
56     android:layout_width="match_parent"  
57     android:layout_height="50dp"  
58     android:orientation="horizontal" />  
59     <RadioButton  
60         android:id="@+id/tb_login1"  
61         android:layout_width="0px"  
62         android:checked="true"  
63         android:layout_height="match_parent"  
64         android:layout_weight="1"  
65         android:text="用户"/>  
66     <RadioButton  
67         android:id="@+id/tb_login2"  
68         android:layout_width="0px"  
69         android:layout_height="match_parent"  
70         android:layout_weight="1"  
71         android:text="导游"/>  
72 </RadioGroup>  
73 <Button  
74     android:id="@+id/bt_login"  
75     android:layout_width="match_parent"  
76     android:layout_height="wrap_content"  
77     android:layout_margin="10dp"  
78     android:text="登录"  
79     android:textStyle="bold" />  
80 <Button  
81     android:id="@+id/bt_login_regist"  
82     android:layout_width="match_parent"  
83     android:layout_height="wrap_content"  
84     android:layout_margin="10dp"  
85     android:text="注册"  
86     android:textStyle="bold" />
```

图 4.3 登陆界面布局文件代码

注册第一步，获取用户输入的注册信息。逻辑代码如图 4.4 所示：



```

58     @OnClick(R.id.bt_regist_register)
59     public void onViewClicked() {
60         String username=tvRegistUsername.getText().toString();
61         String password=tvRegistPassword.getText().toString();
62         String realname=tvRegistRealname.getText().toString();
63         String phone=tvRegistPhone.getText().toString();
64         if (username.length() < 1 || password.length() < 1 || realname.length() < 1 || phone.length() < 1) {
65             Toast.makeText(context, RegistActivity.this, text: "输入错误!", Toast.LENGTH_SHORT).show();
66             return;
67         } else {
68
69             User user = new User(username, password, realname, phone);
70             regist(user);
71
72         }
73     }
74
75
76 }

```

图 4.4 获取注册信息代码

注册第二步，将注册信息发送给服务器。代码如图 4.5、4.6 所示：



```

79     private void regist(User user) {
80         OkHttpClient okHttpClient = new OkHttpClient();
81         String url = new Url().url +
82             "Api/User/Register?username=" + user.getUsername() +
83             "&password=" + user.getPassword() +
84             "&realname=" + user.getRealname() +
85             "&description=" + user.getDescription();
86         Log.i(TAG, msg: "url: " + url);
87         final Request request = new Request.Builder()
88             .url(url)
89             .get()
90             .build();
91         Call call = okHttpClient.newCall(request);

```

图 4.5 注册功能实现代码



```

92     call.enqueue(new Callback() {
93
94     @Override
95     public void onFailure(Call call, IOException e) {
96         Log.i(TAG, msg: "onFailure: " + e.getMessage() + e.toString());
97         Looper.prepare();
98         Toast.makeText(context, RegistActivity.this, text: "网络错误", Toast.LENGTH_LONG).show();
99         Looper.loop();
100    }
101    @Override
102    public void onResponse(Call call, Response response) throws IOException {
103        String ajax = response.body().string();
104        Log.i(TAG, msg: "onResponse: " + ajax);
105        Gson gson = new Gson();
106        Ajax ajaxe = gson.fromJson(ajax, Ajax.class);
107        if (ajaxe.getStatus() > 0) {
108            Looper.prepare();
109            Toast.makeText(context, RegistActivity.this, ajaxe.getMsg(), Toast.LENGTH_LONG).show();
110            //handler.sendMessage(1);
111            Looper.loop();
112        } else {
113            Looper.prepare();
114            Toast.makeText(context, RegistActivity.this, ajaxe.getMsg(), Toast.LENGTH_LONG).show();
115            Looper.loop();
116        }
117    });
118 }

```

图 4.6 注册功能实现代码

这里以游客用户为例，因导游用户与游客用户注册设计基本相似，故不多赘述。

接着是登录功能，登陆界面中登录功能是通过一个 Radio Button 控件实现切换导游和游客用户的登录。用户登录功能实现代码如图 4.7、4.8、4.9 所示：



```

67     @OnClick(R.id.bt_login)
68     public void onBtLoginClicked() {
69         if (isGuide) {
70             guideLogin();
71         } else {
72             userLogin();
73         }
74     }
75
76     private void userLogin() {
77
78         OkHttpClient okHttpClient = new OkHttpClient();
79         String url = new Url().url
80             + "/Api/User/Login?username=" + tvLoginUsername.getText().toString()
81             + "&password=" + tvLoginPassword.getText().toString();
82         final Request request = new Request.Builder().url(url).get().build();
83         Call call = okHttpClient.newCall(request);
84         call.enqueue(new Callback() {
85             @Override
86             public void onFailure(Call call, IOException e) {
87                 Log.i(TAG, "连接失败: " + e.getMessage() + e.toString());
88                 Looper.prepare();
89                 Toast.makeText(context, LoginActivity.this, text: "连接失败" + e.getMessage() + e.toString(), Toast.LENGTH_LONG).show();
90                 Looper.loop();
91             }
92         });
93     }
94
95     private void guideLogin() {
96
97         OkHttpClient okHttpClient = new OkHttpClient();
98         String url = new Url().url
99             + "/Api/User/Login?username=" + tvLoginUsername.getText().toString()
100            + "&password=" + tvLoginPassword.getText().toString();
101        final Request request = new Request.Builder().url(url).get().build();
102        Call call = okHttpClient.newCall(request);
103        call.enqueue(new Callback() {
104            @Override
105            public void onFailure(Call call, IOException e) {
106                Log.i(TAG, "连接失败: " + e.getMessage() + e.toString());
107                Looper.prepare();
108                Toast.makeText(context, LoginActivity.this, text: "连接失败" + e.getMessage() + e.toString(), Toast.LENGTH_LONG).show();
109                Looper.loop();
110            }
111        });
112    }
113
114    private void checkLogin() {
115        if (isGuide) {
116            tvLoginUsername.setText("导游名");
117            tvLoginPassword.setText("导游密码");
118        } else {
119            tvLoginUsername.setText("用户名");
120            tvLoginPassword.setText("密码");
121        }
122    }
123
124    private void init() {
125        tvLoginUsername = findViewById(R.id.tv_login_username);
126        tvLoginPassword = findViewById(R.id.tv_login_password);
127        bt_login = findViewById(R.id.bt_login);
128        bt_guide = findViewById(R.id.bt_guide);
129        bt_visitor = findViewById(R.id.bt_visitor);
130        bt_guide.setCheck(true);
131    }
132
133    @Override
134    protected void onCreate(Bundle savedInstanceState) {
135        super.onCreate(savedInstanceState);
136        setContentView(R.layout.activity_login);
137        ButterKnife.bind(this);
138        init();
139        checkLogin();
140    }
141
142    @Override
143    public void onBackPressed() {
144        Intent intent = new Intent(LoginActivity.this, MainActivity.class);
145        startActivity(intent);
146        finish();
147    }

```

图 4.7 登录功能实现代码



```

149     @Override
150     public void onResponse(Call call, Response response) throws IOException {
151         String ajax = response.body().string();
152         Log.i(TAG, msg: "onResponse: " + ajax);
153         Gson gson = new Gson();
154         try {
155             //登陆成功
156             User user = gson.fromJson(ajax, User.class);
157             if (user.getId() > 0) {
158                 Log.i(TAG, user.toString());
159                 SharedPreferences sharedPreferences = getSharedPreferences(name: "guide", Context.MODE_PRIVATE);
160                 SharedPreferences.Editor editor = sharedPreferences.edit();
161                 editor.putString("username", user.getUsername());
162                 editor.putString("realname", user.getRealname());
163                 editor.putString("description", user.getDescription());
164                 editor.putInt("userid", user.getId());
165                 editor.commit();
166                 Intent intent = new Intent(packageContext: LoginActivity.this, MainActivity.class);
167                 startActivity(intent);
168                 finish();
169                 Looper.prepare();
170                 Toast.makeText(context: LoginActivity.this, text: "success" + user.toString(), Toast.LENGTH_LONG).show();
171                 Looper.loop();
172             } else {
173                 //登陆失败
174                 Looper.prepare();
175                 Toast.makeText(context: LoginActivity.this, text: "fail", Toast.LENGTH_LONG).show();
176                 Looper.loop();
177             }
178         } catch (Exception e) {
179             Looper.prepare();
180             Toast.makeText(context: LoginActivity.this, text: "fail", Toast.LENGTH_LONG).show();
181             Looper.loop();
182         }
183     }
184
185 }

```

图 4.8 登录功能实现代码



```

172     } else {
173         //登陆失败
174         Looper.prepare();
175         Toast.makeText(context: LoginActivity.this, text: "fail", Toast.LENGTH_LONG).show();
176         Looper.loop();
177     }
178 } catch (Exception e) {
179     Looper.prepare();
180     Toast.makeText(context: LoginActivity.this, text: "fail", Toast.LENGTH_LONG).show();
181     Looper.loop();
182 }
183
184
185 }

```

图 4.9 登录功能实现代码

这里也以游客用户为例，因导游用户与游客用户登录功能设计基本相似，故不多赘述。

4.2.2 用户订单模块

游客用户通过输入城市名称来获取周围景点，并通过单击选择想要去的景点进入下单页面。接着再下单页面通过输入需求后完成下单操作，此时该订单数据会以未接单的状态被数据库记录。导游用户此时可以在订单列表查看未接单状态的订单，并根据自身条件进行申请操作，进入等待被接单的状态。接着游客用户可以在自己的订单中查看接单导游个人资料，如用户名以及资质，并进行挑选以及锁定心仪导游，此时订单将以已接单的状态

被数据库记录。流程图如图 4.10 所示：

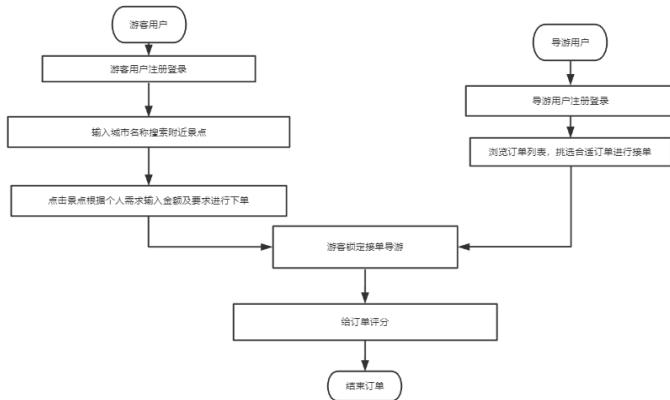


图 4.10 订单发布及接单流程图

首页对于游客用户其功能主要体现为通过输入城市名返回周围景点名称，这是为游客用户设计的功能。如图 4.11 所示为首页界面的布局代码。在本界面可以通过底部的导航栏跳转至其他模块。在首页中采用的是一个大的线性布局，搜索功能由一个 EditText 文本框和一个 Button 按钮实现，返回结果由 ListView 控件实现。

```

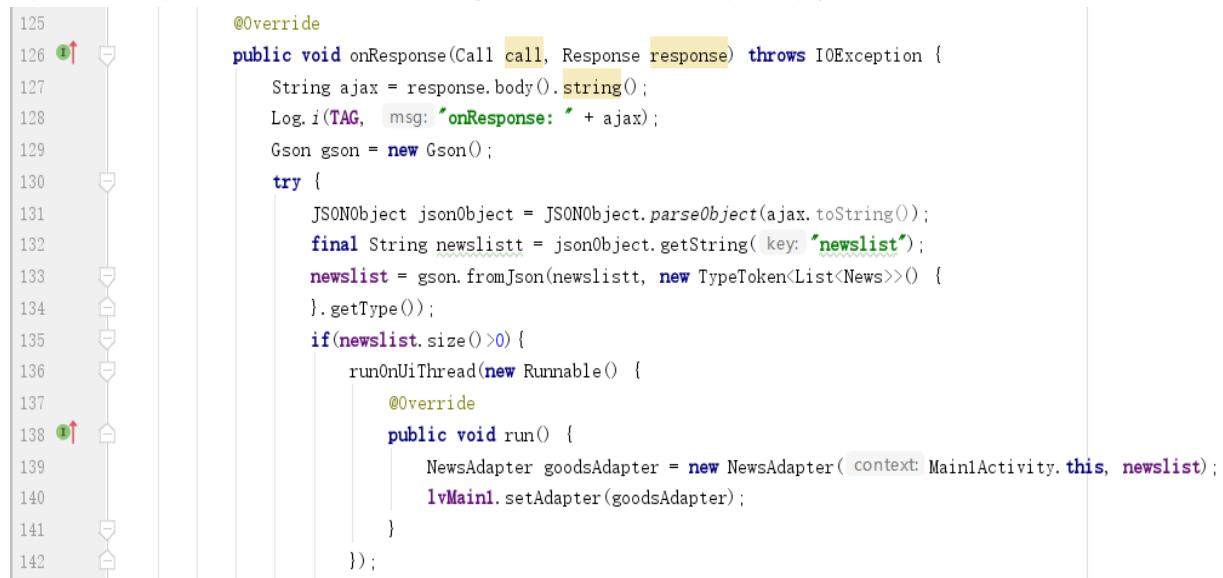
32 <EditText
33     android:id="@+id/et_main1"
34     android:layout_width="match_parent"
35     android:layout_height="50dp"
36     android:layout_margin="10dp"
37     android:hint="请输入城市名进行景点搜索"
38     android:background="@android:drawable/editbox_background_normal"/>
39
40
41 <View
42     android:layout_width="match_parent"
43     android:layout_height="1px"
44     android:background="@color/gray1dp"/>
45
46
47 <Button
48     android:id="@+id/bt_main1"
49     android:layout_width="match_parent"
50     android:layout_height="wrap_content"
51     android:layout_marginLeft="10dp"
52     android:layout_marginRight="10dp"
53     android:clickable="false"
54     android:text="查询"/>
55
56 <ListView
57     android:id="@+id/lv_main1"
58     android:layout_width="match_parent"
59     android:layout_height="match_parent"></ListView>

```

图 4.11 首页模块布局代码

游客通过在搜索栏输入地名来了解周边大致景点，这时系统获取周边景点后，同时生

成获取到的景点列表显示在主界面供游客用户选取。实现代码如图 4.12 所示：



```

125
126     @Override
127     public void onResponse(Call call, Response response) throws IOException {
128         String ajax = response.body().string();
129         Log.i(TAG, msg: "onResponse: " + ajax);
130         Gson gson = new Gson();
131         try {
132             JSONObject jsonObject = JSONObject.parseObject(ajax.toString());
133             final String newslistt = jsonObject.getString(key: "newslist");
134             newslist = gson.fromJson(newslistt, new TypeToken<List<News>>() {
135                 .getType());
136             if(newslist.size() > 0) {
137                 runOnUiThread(new Runnable() {
138                     @Override
139                     public void run() {
140                         NewsAdapter goodsAdapter = new NewsAdapter(context: MainActivity.this, newslist);
141                         lvMain.setAdapter(goodsAdapter);
142                     }
143                 });
144             }
145         } catch (Exception e) {
146             e.printStackTrace();
147         }
148     }

```

图 4.12 首页功能实现代码

接着游客点击某个景点，系统跳转进入订单生成页面。下单界面根据游客用户选择的景点的信息，以及游客用户输入的要求、价格、日期，将其发送给服务器，并由此生成订单。其逻辑代码如图 4.13 所示：



```

84
85     private void orderrAdd() {
86         String content = etItemAdd1.getText().toString();
87         String description = etItemAdd2.getText().toString();
88         String price = etItemAdd3.getText().toString();
89         String ordertime = etItemAdd4.getText().toString();
90         SharedPreferences sharedpreferences = getSharedPreferences(name: "guide", Context.MODE_PRIVATE);
91         int userid = sharedpreferences.getInt(key: "userid", defaultValue: 0);
92         String username = sharedpreferences.getString(key: "username", defaultValue: "");
93         String realname = sharedpreferences.getString(key: "realname", defaultValue: "");
94         OkHttpClient okHttpClient = new OkHttpClient();
95         String url = new Uri().uri +
96             "?Api/Order/Add?content=" + content +
97             "&userid=" + userid +
98             "&username=" + realname +
99             "&description=" + description +
100            "&price=" + price +
101            "&ordertime=" + ordertime;
102         Log.i(TAG, msg: "url: " + url);
103         final Request request = new Request.Builder()
104             .url(url)
105             .get()
106             .build();
107         Call call = okHttpClient.newCall(request);
108         call.enqueue(new Callback() {
109             @Override
110             public void onFailure(Call call, IOException e) {
111                 Log.i(TAG, msg: "onFailure: " + e.getMessage() + e.toString());
112                 Looper.prepare();
113                 Toast.makeText(context: OrderAddActivity.this, text: "网络错误", Toast.LENGTH_LONG).show();
114                 Looper.loop();
115             }
116         });

```

图 4.13 下单功能实现代码

导游的订单查询功能页面分为两个模块，即查看未接单订单的导游首页模块以及查看已接单订单的订单模块。这两个模块布局均为线性布局，由一个 List View 控件控制。布局

代码如图 4.14 所示：

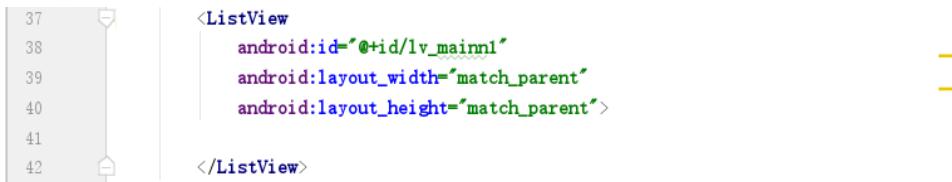


图 4.14 导游首页及订单模块布局代码

首先在首页界面主要功能为向数据库对未接单订单数据进行请求，获取未接单订单列表，当获取到订单列表后，将列表显示出来。当导游用户单击其中任意订单后，跳转进入该订单的导游订单详情页。其逻辑代码如图 4.15 所示：



图 4.15 显示并跳转至接单界面实现代码

接着进入导游接单界面。该界面由七个 Text View 控件以及一个申请 Button 按钮构成，其中文本框分别显示由服务器请求获取到的景点、对导游要求、单日价格、旅程开始时间、接单导游、下单时间以及订单状态。单击申请按钮对应抢单（即下单）操作，申请成功便进入待接单状态。接着向服务器发送申请指令，服务器增加抢单数据，其逻辑代码如图 4.16 所示：



图 4.16 向服务器发送接单请求实现代码

游客用户查看、确认接单导游及评价功能将满足游客用户挑选接单导游的需求，当导游用户发起接单申请后，游客可在游客订单详情页面进行查看，并且可以单击接单导游查看信息，并挑选出心仪导游，在完成订单后跳转至评论界面。其布局设计皆为导游接单界面基础上加以改动，游客订单详情界面主要改动为返回接单导游数值的 List View，而导游详情界面则将 Text View 返回的信息改为导游详情，同时 Button 作用为锁定接单导游。其

中评价界面的布局代码如图 4.17 所示：



图 4.17 评价布局代码

向服务器请求并返回接单导游列表，接着游客进入导游详情界面及锁定导游功能其逻辑代码如图 4.18 所示：

```

75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115

```

```

private void init() {
    apply = (Apply) getIntent().getSerializableExtra( name: 'apply' );
    OkHttpClient okHttpClient = new OkHttpClient();
    String url = new Url().url +
        "Api/Guide/Find?id=" + apply.getGuideid();
    Log.i(TAG, msg: 'url: ' + url);
    final Request request = new Request.Builder()
        .url(url)
        .get() //默认就是GET请求，可以不写
        .build();
    Call call = okHttpClient.newCall(request);
    call.enqueue(new Callback() {
        @Override
        public void onFailure(Call call, IOException e) {...}

        @Override
        public void onResponse(Call call, Response response) throws IOException {
            String ajax = response.body().string();
            Log.i(TAG, msg: 'onResponse: ' + ajax);
            Gson gson = new Gson();
            final Guide guide = gson.fromJson(ajax, Guide.class);
            if (guide.getId() > 0) {
                runOnUiThread() -> {
                    tvItem1Detail1.setText("账号：" + guide.getUsername());
                    tvItem1Detail2.setText("姓名：" + guide.getRealname());
                    tvItem1Detail3.setText(guide.getDescription());
                    tvItem1Detail3.setText("级别：" + guide.getRank());
                });
            } else {
                Looper.prepare();
                Toast.makeText( context: GuideActivity.this, text: "错误", Toast.LENGTH_LONG).show();
                Looper.loop();
            }
        }
    });
}

```

图 4.18 锁定导游功能实现代码

当返回订单状态为已接单后显示为评论，代码如图 4.19 所示：



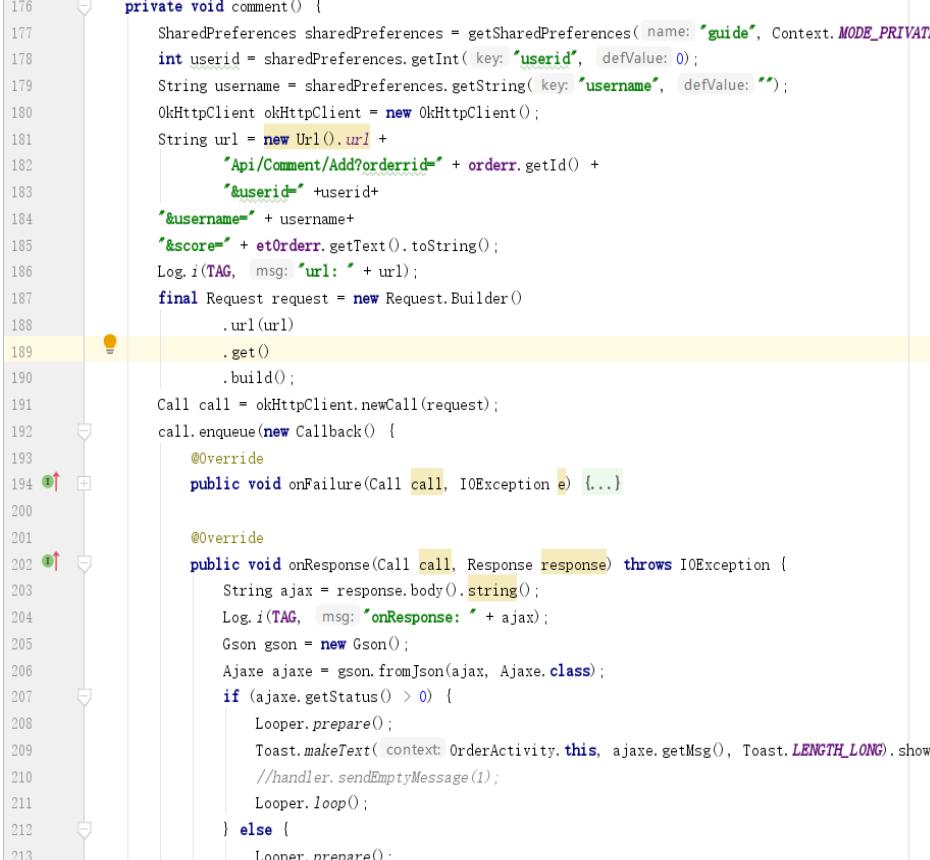
```

101 if (orderr.getState().equals("未接单")) {
102     OkHttpClient okHttpClient = new OkHttpClient();
103     String url = new Url().url + "/Api/Apply/Index?orderid=" + orderr.getId();
104     Log.i(TAG, url);
105     final Request request = new Request.Builder().url(url).get().build();
106     Call call = okHttpClient.newCall(request);
107     call.enqueue(new Callback() {
108         @Override
109         public void onFailure(Call call, IOException e) {
110             Log.i(TAG, msg: "连接失败: " + e.getMessage() + e.toString());
111             Looper.prepare();
112             Toast.makeText(context, OrderActivity.this, text: "连接失败" + e.getMessage() + e.toString(), Toast.LENGTH_LONG).show();
113             Looper.loop();
114         }
115
116         @Override
117         public void onResponse(Call call, Response response) throws IOException {
118         }
119     });
120
121     } else {
122         lvOrderr.setVisibility(View.GONE);
123         etOrderr.setVisibility(View.VISIBLE);
124         btIteml.setVisibility(View.VISIBLE);
125     }
126
127 }

```

图 4.19 显示评论界面实现代码

用户输入评分并上传，系统将评论内容发送至服务器，其逻辑代码如图 4.20 所示：



```

176 private void comment() {
177     SharedPreferences sharedPreferences = getSharedPreferences(name: "guide", Context.MODE_PRIVATE);
178     int userid = sharedPreferences.getInt(key: "userid", defaultValue: 0);
179     String username = sharedPreferences.getString(key: "username", defaultValue: "");
180     OkHttpClient okHttpClient = new OkHttpClient();
181     String url = new Url().url +
182         "/Api/Comment/Add?orderid=" + orderr.getId() +
183         "&userid=" + userid +
184         "&username=" + username +
185         "&score=" + etOrderr.getText().toString();
186     Log.i(TAG, msg: "url: " + url);
187     final Request request = new Request.Builder()
188         .url(url)
189         .get()
190         .build();
191     Call call = okHttpClient.newCall(request);
192     call.enqueue(new Callback() {
193         @Override
194         public void onFailure(Call call, IOException e) {
195         }
196
197         @Override
198         public void onResponse(Call call, Response response) throws IOException {
199             String ajax = response.body().string();
200             Log.i(TAG, msg: "onResponse: " + ajax);
201             Gson gson = new Gson();
202             Ajax ajaxe = gson.fromJson(ajax, Ajax.class);
203             if (ajaxe.getStatus() > 0) {
204                 Looper.prepare();
205                 Toast.makeText(context, OrderActivity.this, ajaxe.getMsg(), Toast.LENGTH_LONG).show();
206                 //handler.sendMessage();
207                 Looper.loop();
208             } else {
209                 Looper.prepare();
210             }
211         }
212     });
213 }

```

图 4.20 上传评价内容实现代码

4.2.3 导游认证功能

该功能中导游上传导游认证图片并交由后台人员进行审核并处理从而达到导游认证的目的，其布局代码如图 4.21 所示：

```

<Button
    android:id="@+id/bt_update"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="40px"
    android:text="上传导游认证图片"/>
<Button
    android:id="@+id/bt_my"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="40px"
    android:text="退出登录"/>

```

图 4.21 导游认证布局代码

其系统操作分为四部：导游选择上传认证图片、系统从本机获取图片、获取从相册返回的 Uri 最后完成上传图片。其逻辑代码如图 4.22 及 4.23 所示：

```

private void getImage() {
    if (Build.VERSION.SDK_INT < Build.VERSION_CODES.KITKAT) {
        startActivityForResult(new Intent(Intent.ACTION_GET_CONTENT).setType("image/*"),
                REQUEST_PICK_IMAGE);
    } else {
        Intent intent = new Intent(Intent.ACTION_OPEN_DOCUMENT);
        intent.addCategory(Intent.CATEGORY_OPENABLE);
        intent.setType("image/*");
        startActivityForResult(intent, REQUEST_PICK_IMAGE);
    }
}

@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    if (resultCode == Activity.RESULT_OK) {
        switch (requestCode) {
            case REQUEST_PICK_IMAGE:
                if (data != null) {
                    realPathFromUri = RealPathFromUriUtils.getRealPathFromUri(context: this, data.getData());
                    upImage();
                    Log.i(TAG, msg: "图片路径"+realPathFromUri);
                    //Toast.makeText(this, "图片路径"+realPathFromUri, Toast.LENGTH_SHORT).show();
                } else {
                    Toast.makeText(context: this, text: "图片损坏, 请重新选择", Toast.LENGTH_SHORT).show();
                }
                break;
        }
    }
}

```

图 4.22 导游上传认证照片代码实现

```

227     private void upImage() {
228         String url = new Url0.url +           "/Api/Guide/update";
229         Log.i(TAG, msg: 'url: ' + url);
230         OkHttpClient mOkHttpClient = new OkHttpClient();
231         File file = new File(realPathFromUri);
232         MultipartBody.Builder builder = new MultipartBody.Builder()
233             .setType(MultipartBody.FORM)
234             .addFormDataPart( name: "image", filename: "HeadPortrait.jpg",
235                 RequestBody.create(MediaType.parse("image/png"), file));
236
237         RequestBody requestBody = builder.build();
238
239         Request request = new Request.Builder()
240             .url(url)
241             .post(requestBody)
242             .build();
243         Call call = mOkHttpClient.newCall(request);
244         call.enqueue(new Callback() {
245             @Override
246             public void onFailure(Call call, IOException e) {...}
247
248             @Override
249             public void onResponse(Call call, Response response) throws IOException {
250                 //Log.e(TAG, "成功 "+response);
251                 String ajax = response.body().string();
252                 Log.i(TAG, msg: 'onResponse: ' + ajax);
253                 Gson gson = new Gson();
254                 final Ajax ajax = gson.fromJson(ajax, Ajax.class);
255                 if(ajax.getStatus()>0) {
256                     image = ajax.getMsg();
257                     Log.i(TAG, msg: '服务器图片名称: ' + image);
258                     runOnUiThread( () -> {
259                         Toast.makeText( context: Mainn3Activity.this, text: '服务器图片名称: ' + image, Toast.LENGTH_SHORT).show();
260                     });
261                 }
262             }
263         });
264     }
265
266     @Override
267     protected void onActivityResult(int requestCode, int resultCode, Intent data) {
268         super.onActivityResult(requestCode, resultCode, data);
269     }

```

图 4.23 导游上传认证照片代码实现

4.3 数据库设计

管理员在后台操作服务器，能够对数据库内的游客、导游的个人资料进行保存和及时修改，特别是导游，管理员需在后台审核通过导游的认证，导游才能将自身资质显示在个人资料中，这也是游客对于专业导游和业余导游的直观区别方法。同时管理员也可以在后台对订单和评价进行修改和调整最后存储到数据库中。流程图如图 4.24 所示：

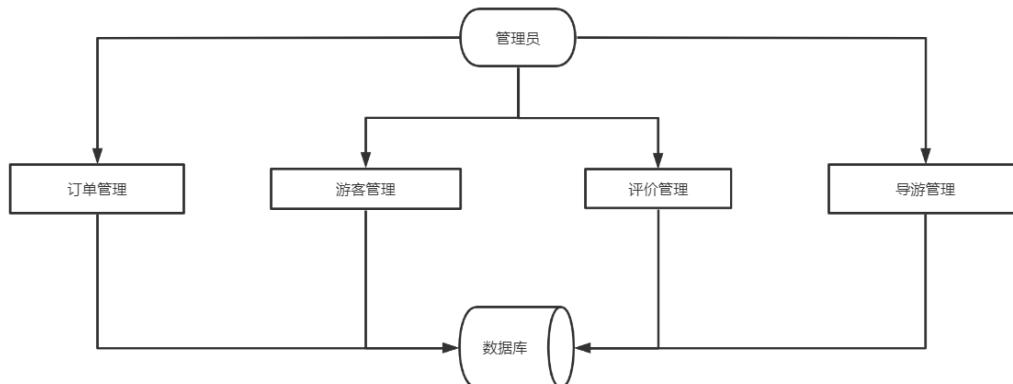


图 4.24 数据库流程图

游客用户的注册表，从未注册的游客用户先注册一个账号同时根据信息登录，使用时用户需要填写用户名和密码，如表 4-1 所示：

表 4-1 游客用户的注册表

名	类型	长度	小数点	许可 null
Id	Int	11	0	✓
Username	Varchar	20	0	✓
Password	Varchar	20	0	✓
Realname	Varchar	20	0	
Description	Varchar	255	0	
Type	Varchar	255	0	

导游用户的注册表，与游客用户相似，同时导游表负责保存导游的认证等级，如表 4-2 所示：

表 4-2 导游用户的注册表

名	类型	长度	小数点	许可 null
Id	Int	11	0	✓
Username	Varchar	255	0	
Password	Varchar	11	0	
Realname	Varchar	255	0	
Rank	Varchar	255	0	
Image	Varchar	255	0	

订单的注册表，管理员登陆后可查询订单的一些信息，比如游客、导游 id，以及关键的订单状态等，如表 4-3 所示：

表 4-3 订单的注册表

名	类型	长度	小数点	许可 null
Id	Int	11	0	✓
Username	Varchar	255	0	
Guideid	Varchar	255	0	
Guidename	Varchar	255	0	
Content	Varchar	255	0	
Description	Varchar	255	0	
Price	Int	11	0	
Createtime	Timestamp	0	0	
Ordertime	Timestamp	0	0	
State	Varchar	255	0	

评分的注册表，记载游客用户对于订单的评价情况，如表 4-4 所示：

表 4-4 评分的注册表

名	类型	长度	小数点	许可 null
Id	Int	11	0	✓
Userid	Varchar	255	0	
Username	Varchar	255	0	
Orderid	Varchar	255	0	
Score	Int	11	0	
Content	Varchar	255	0	
Create	Timestamp	0	0	

第 5 章 系统的测试

5.1 测试流程及结果

5.1.1 注册及登录功能测试

注册及登录界面测试用例，如表 5-1 所示

表 5-1 注册及登录界面测试用例

输入	期待结果	实际出现结果	结果
按指示注册账号	注册成功	注册成功	期待结果与实际结果相符
注册一个相同账号	注册失败	注册失败	期待结果与实际结果相符
输错密码	注册失败	注册失败	期待结果与实际结果相符
输错用户名	注册失败	注册失败	期待结果与实际结果相符
切换导游登录游客账号	注册失败	注册失败	期待结果与实际结果相符

界面图及测试结果如下。

首先正常注册一个游客账号，如图 5.1 所示

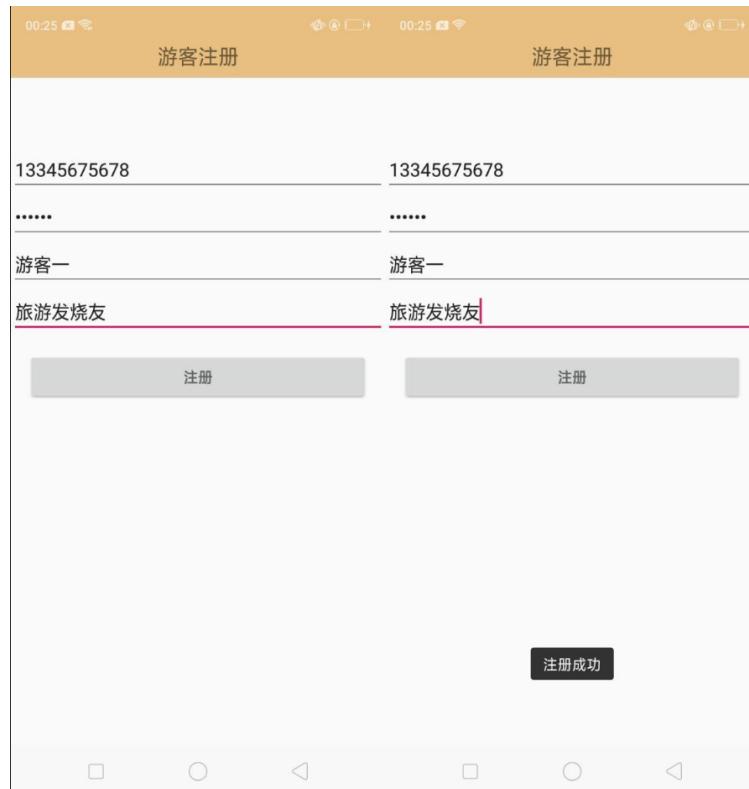


图 5.1 游客注册

接着尝试以相同账号进行注册，结果显示用户名已存在，如图 5.2 所示：



图 5.2 用户名已存在

然后在登陆时，分别尝试输错密码、输错用户名、错登导游，均返回错误提示。如图 5.3 所示：



图 5.3 登陆失败

最后注册两个导游账号，如图 5.4 及图 5.5 所示：



图 5.4 导游注册



图 5.5 导游注册

5.1.2 订单功能测试

订单功能界面测试用例，如表 5-2 所示：

表 5-2 订单功能测试用例表

输入	期待结果	实际出现结果	结果
搜索景点	搜索成功	搜索成功	期待结果与实际结果相符
单击跳转至下单	跳转成功	跳转成功	期待结果与实际结果相符
按需求完成下单	下单成功	下单成功	期待结果与实际结果相符
导游浏览订单	可以发现未接单订单	可以发现未接单订单	期待结果与实际结果相符
多导游接单显示导游列表	全部显示	全部显示	期待结果与实际结果相符
选取导游订单变为已接单	接单成功	接单成功	期待结果与实际结果相符
评价	评价成功	评价成功	期待结果与实际结果相符

界面图及测试结果如下。

首先搜索地点并单击景区跳转至下单界面，完成下单后查看订单详情，如图 5.6、图 5.7 及图 5.8 所示：



图 5.6 新建订单



图 5.7 新建订单



图 5.8 新建订单

接着登录导游账号查看订单列表，如图 5.9 所示：



图 5.9 浏览订单

操作两个导游账号共同接取刚刚新建的订单，如图 5.10 所示：

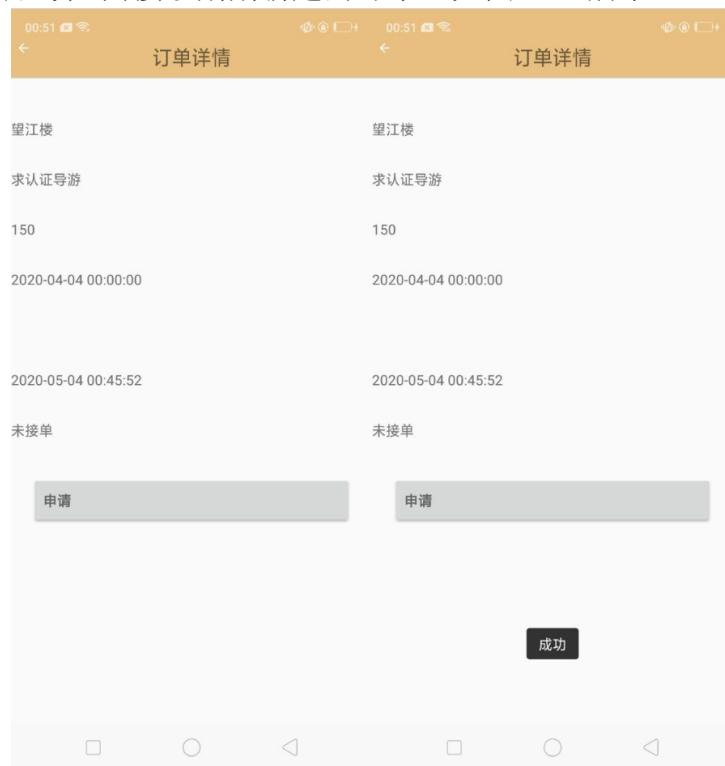


图 5.10 导游接单

然后再登录游客账号查看订单详情，这时显示有两个导游选择接单，如图 5.11 所示：



图 5.11 订单详情

接下来点击高亮字段进入导游详情页，并锁定接单导游，如图 5.12 所示：



图 5.12 选定导游

完成订单后我们再点进订单页，刚才的订单显示为已接单。再点进详情页，显示评分文本框，最后打分成功完成整个订单流程，如图 5.13 及图 5.14 所示：

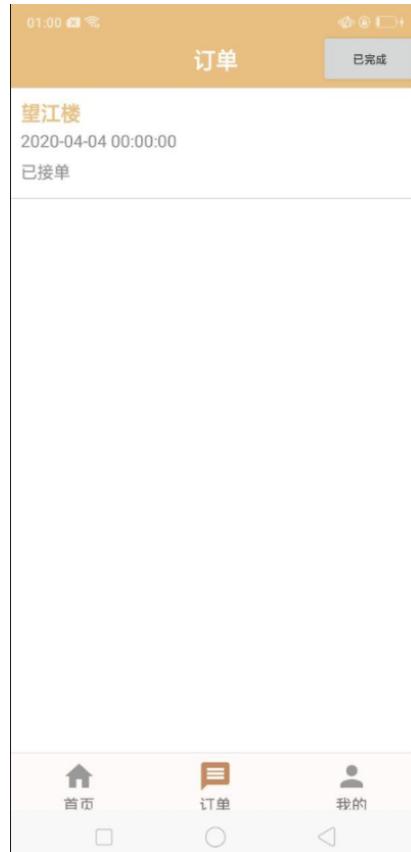


图 5.13 订单评价



图 5.14 订单评价

5.2 测试结果分析

本系统编码实现后陆续对各功能进行了严密的系统测试，最后的测试结果表明，本次设计开发的网约导游系统基本上可实现预期达到的效果，并且达到了实际应用的基本要求。

第 6 章 总 结

6.1 工作总结

结合着当下 Android 操作平台的流行趋势，“旅猫导游宝”APP 采用 Android 系统作为客户端进行开发设计，同时利用 Think PHP 框架进行后台的搭建，最终实现多用户的管理。该“旅猫导游宝”系统基本完成了需求分析中所有的功能。实现了游客用户和导游用户注册的功能以及用户登录的功能。游客用户查看景点，发布订单，查看订单，确认导游，评价订单；导游用户浏览订单，选择接单，等待接单等模块。由于开发经验很少，所以本次开发效率不高，开发周期较长，而且很多小的细节问题考虑的不是那么周全，并非完美的解决方案，所以仍需后续的不断学习，加强对于系统开发的知识以及理解，增加项目开发经验，提高编程能力，将本 APP 不断地完善和更新，使其不断趋于完善。

同时针对本系统的开发背景，希望本系统的开发构想能够成为一个真正解决“黑导”“恶导”以及解放被旅行社捆绑压榨的导游等社会问题的 APP，使游客能按价格得到称心的服务，让导游提升工作效率、工作信心和工作成就感，享受到真正的工作成果，从而最终达到缓解导游与游客这两个社会群体的矛盾，为这个社会的和谐与稳定做出应有的贡献。

6.2 展望

本文中虽已实现了对于订单发布及抢单接单等核心功能，但是由于开发周期较为紧凑其开发者也是初次进行开发实践，所以其实很多地方仍然可以进行仔细推敲、修改和完善：

首先是在界面上的美观程度，因为系统为自己开发，并且开发重点基本都落在逻辑代码的设计上，故而 UI 界面其实有很大的改善空间，如设计 LOGO，增加界面背景，设计浮窗，增加广告位等等。

其次在功能上，首先是用户注册现阶段还是简单的输入电话号码直接进行注册，下一步其实可以申请手机验证短信的 API 接口，通过该接口达到对用户手机号码验证的功能。其次，导游认证也是可以通过 API 接口使用户直接通过输入导游证号进行查询，系统自动对返回的结果进行处理。

参考文献

- [1] 李崇, 可娜.基于共享经济的网约导游平台服务.经济发展研究, 2018(9):144
- [2] 张豪锋, 杨绪辉.基于微博的移动办公实例研究[J].继续教育研究, 2014(03)
- [3] Chandra, A., GPS Locator an Application for Location Tracking and Sharing Using GPS for Java Enabled Handhelds[C]. 2011 International Conference on Computational Intelligence and Communication Networks (CICN), 2011:406-410.
- [4] 王红崧, 周海宴.基于百度地图 API 的旅游地理信息系统开发[J].现代计算机, 2010(8):60-63.
- [5] 徐潇潇, 谢林柏, 彭力, 等.基于 WIFI 信号强度特征的室内定位系统设计[J].计算机工程, 2015, 41(4):87-91.
- [6] 代敏.基于 Android 平台下手机定位程序的设计及实现 [J].计算机数字工程, 2012, 40(4):143-145.
- [7] 李崇, 可娜.基于共享经济的网约导游平台服务.经济发展研究, 2018(9):144
- [8] 王璟.基于 Android 的健身助手 APP 设计与实现.吉林大学, 2018
- [9] 刘新, 翁锡全, 林文弢.基于智能手机健身 App 自我监控干预日常体力活动的实验研究 [A]. 2015 第十届全国体育科学大会论文摘要汇编 (二) [C]. 2015:18
- [10] 黄冠瑞.基于 Android 的智能外卖配送系统的实现.昆明理工大学, 2016
- [11] 李宁.Android/OPhone 开发完全讲义[M].中国水利水电出版社, 2010
- [12] 王楠.基于安卓的网上商城手机 APP 设计与实现.吉林大学, 2018
- [13] 王楠.基于安卓的网上商城手机 APP 设计与实现.吉林大学, 2018
- [14] 王璟.基于 Android 的健身助手 APP 设计与实现.吉林大学, 2018
- [15] 王楠.基于安卓的网上商城手机 APP 设计与实现.吉林大学, 2018

致 谢

当写到现在这段话时我真的百感交集。写下这段致谢语也就意味着我真的就要毕业，离开这所我生活了四年的学校。说实话在这里学到的不仅仅有知识，还有对于我人生能够一生受益的东西，比如待人处事的能力，自我学习的能力以及重塑我对于人生，对家庭，对社会的观念，这也是我能顺利完成学习任务，顺利完成毕业论文撰写的核心原因。同时这篇论文的完成也离不开老师对我的帮助和指导，故而在这毕业之际，我谨向我的导师表示衷心的感谢！

在完成毕业设计期间，从最开始的拟定研究方向，老师就对仍然处于迷茫的我进行了点拨，接着在毕业设计阶段，老师也时常对我进行知识上的帮助，开发过程的指导以及进度上督促，这让以前几乎没有任何开发经验的我获益良多。感谢我的导师谢盈谢老师，不仅在学业上对我帮助良多，并且老师也不断在开发过程中对我进行鼓励和指导，使我变得更加沉稳以及充满信心。

此外，我还要对我的父母，帮助过我的朋友表示感谢，是他们的关心和帮助才让我能这么安心和顺利的完成大学的学业生涯。