**概**

**要**

**设**

**计**

**说**

**明**

**书**

**项目组长 魏诚**

**开发成员 黄纬纶**

**苏义豪**

**梅阳照**

**曾辰浩**

**项目名称 旅游攻略推荐系统**

目录

[1 引言 4](#_Toc512629842)

[1.1编写目的 4](#_Toc512629843)

[1.2 项目背景 4](#_Toc512629844)

[1．3定义 4](#_Toc512629845)

[2 总体设计方案 5](#_Toc512629846)

[2.1建设目标 5](#_Toc512629847)

[2.2运行环境 5](#_Toc512629848)

[2.3业务需求 5](#_Toc512629849)

[3功能模块设计 6](#_Toc512629850)

[3.1总体模块之间的关系： 6](#_Toc512629851)

[3.1 物理部署架构 6](#_Toc512629852)

[3.2系统的层次模型 7](#_Toc512629853)

[3.2.1 客户端(Client Tier) 8](#_Toc512629854)

[3.2.2 表现层(Presentation Tier) 8](#_Toc512629855)

[3.2.3 业务控制层(Usecase Controller Tier) 8](#_Toc512629856)

[3.2.4 业务服务层(Business Service Tier) 8](#_Toc512629857)

[3.2.5 集成层(Interface Tier) 8](#_Toc512629858)

[3.2.6 数据层(Resource Tier) 8](#_Toc512629859)

[3.3功能划分及处理流程 9](#_Toc512629860)

[3.4 设计原则 11](#_Toc512629861)

[3.5 体系架构 12](#_Toc512629862)

[4 数据结构设计 12](#_Toc512629863)

[4.1主要表结构 12](#_Toc512629864)

[4.2E-R图设计 12](#_Toc512629865)

[5 爬虫结构设计 13](#_Toc512629866)

[5.1爬虫语言及框架 13](#_Toc512629867)

[6 接口设计 13](#_Toc512629868)

[6.1用户接口 13](#_Toc512629869)

[6.2外部接口 14](#_Toc512629870)

[6.2.1、软件接口 14](#_Toc512629871)

[6.3内部接口 14](#_Toc512629872)

[7 运行设计 14](#_Toc512629873)

[7.1运行模块组合 14](#_Toc512629874)

[7.2运行控制 15](#_Toc512629875)

[7.3运行时间 15](#_Toc512629876)

[8 系统安全性设计 15](#_Toc512629877)

[8.1 数据库核心数据 16](#_Toc512629878)

[8.2 数据库面临的威胁 16](#_Toc512629879)

[8.3 数据库的安全策略 16](#_Toc512629880)

[9 系统异常处理设计 17](#_Toc512629881)

[9.1出错信息 17](#_Toc512629882)

[8.2补救措施 18](#_Toc512629883)

# 1 引言

## 1.1编写目的

 本节概要设计是为了开发在线系统功能设计的，主要面向系统分析员、程序 员、测试员、实施人员和最终用户。它对以后阶段的工作起指导作用。本文也是项目完成后系统验收的依据。

编写该文档的目的在于从总体设计的角度明确旅游攻略推荐系统的功能和 处理模式，服务于运营，管理的各个环节；明确用户对在管理系统的功能需求和 性能需求，并将这些需求用规范化的语言和规范化的结构完整、准确地表达清楚， 以此统一软件开发者和用户对该管理软件系统的理解和认识；可以有针对性的进行系统开发、测试、验收等各方面的工作，这是开发该旅游攻略推荐系统的基础。

## 1.2 项目背景

旅游出行已经成为现在人们节假日生活的一部分，也有各种各样的网站或地图为我们提供帮助，但我们的出行依旧有着各种各样的问题。

对于用户来说，在决定出行的时候经常是找到一起出行的人，准备好了旅游的花费，但是没有准备好要去哪，怎么去。在这种情况下，现在市面上的网站或者应用并没有一个很好的解决办法，而我们这个项目为这种用户提供了一个选择，用户可以提交他们尚未完善的出行想法，将用户所想看到的风景告诉我们，而我们的项目向用户反馈一个更加细致，更加完善的出行方案。其中用户可能会对我们的方案并不满意，但会让用户对出行有一个更加明确的认识，继而我们可以进一步完善用户的出行需求，给用户提供例如最省时间的方案或者最快到达的方案。

## 1．3定义

|  |  |  |
| --- | --- | --- |
| 术语缩写 | 术语全称 | 中文翻译(供参考) |
| CRUD | Create/Retrieve/Update/Delete | 增删改查(四种基本的数据操作) |
| BP | Business Process | 业务过程 |
| BO | Business Object | 业务对象 |
| VO | Value Object | 值对象 |
| MVC | Model/View/Controller | 模型/表示/控制模式 |
| DAO | Date Access Object | 数据访问对象 |

# 2 总体设计方案

## 2.1建设目标

1.为用户完成出行路线的规划；

2.为用户完成最佳路线的选择（最短时间，最短路程）；

3.为用户出行提出合理的建议，积攒用户粘性。

## 2.2运行环境

我们的项目使用Windows操作系统，在数据库管理系统的使用上，我们最终选择了MySQL数据库来存储我们的重要数据，在其他支撑软件上，我们预计将会使用Dreamviewer，Hbuilder，MyEclipse等软件作为前台后台的支撑软件。

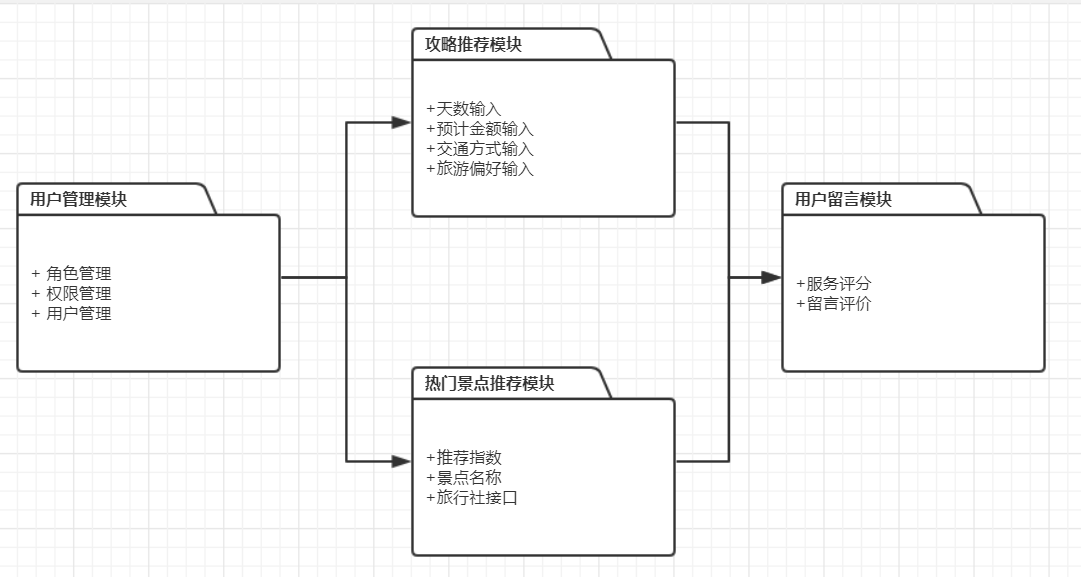
## 2.3业务需求

1.用户仅仅需要提供自己期望的信息和当前状况就可以得到一个即时的反馈，从而节省用户大量的时间和精力。

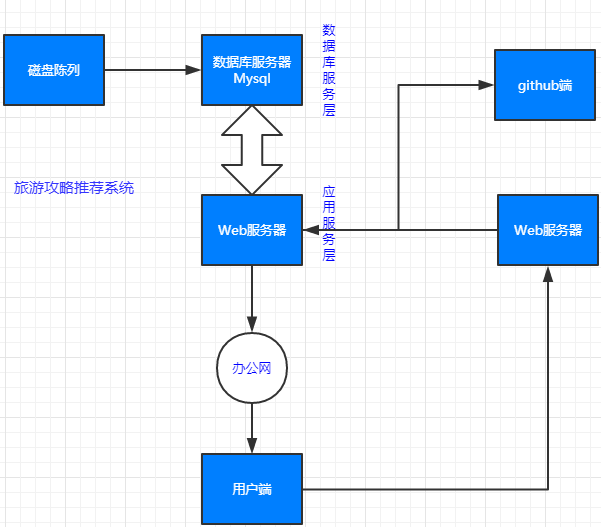
2.在之后更新的版本中，用户可以将自身情况和规划好的出行计划进行比对，在出行计划不符合自身预期时，用户可以及时地向软件客服反馈，在软件更新了第二版的出行计划之后继续根据自身需要选择继续要求更改或者选择该出行计划。

# 3功能模块设计

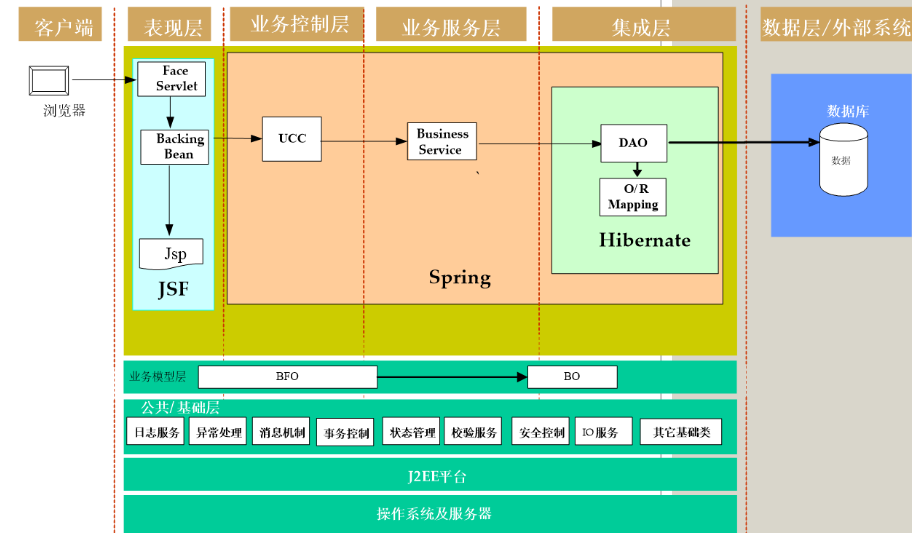
## 3.1总体模块之间的关系：



## 3.1 物理部署架构



## 3.2系统的层次模型



### 3.2.1 客户端(Client Tier)

客户端指的是访问应用的web浏览器终端，通过web浏览器来访问旅游攻略推荐系统。

### 3.2.2 表现层(Presentation Tier)

表现层接收客户端的 HTTP 请求，提供系统登陆，会话管理，访问控制，数据封装等功能。

### 3.2.3 业务控制层(Usecase Controller Tier)

对表示层发来的数据格式进行检查判断，根据不同的业务将数据分配到不同的业务处理服务进行处理。

### 3.2.4 业务服务层(Business Service Tier)

业务层是 J2EE 构架的核心层，它接收展示层分发的交易请求，完成业务逻辑的具体实现。对不同的业务数据进行处理，处理完成后，将处理结果返回表现层。

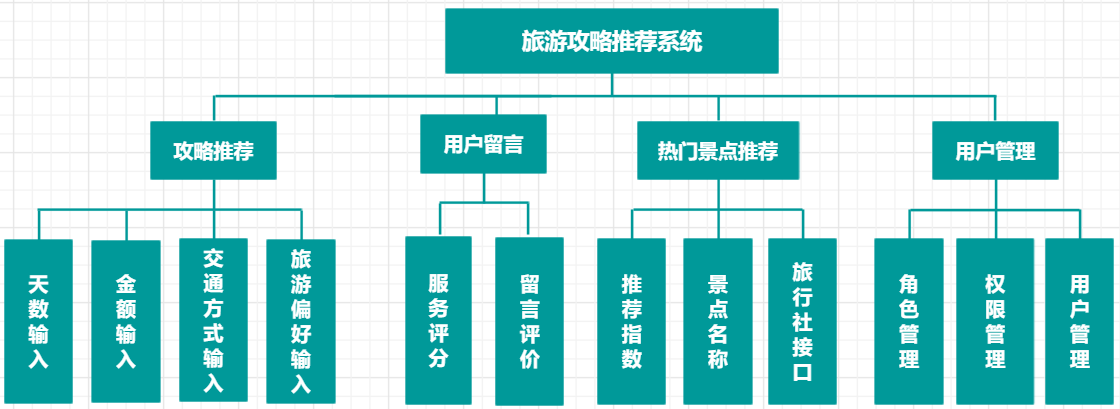
### 3.2.5 集成层(Interface Tier)

集成层向业务层提供统一的内部和外部资源访问，为业务层的数据访问请求屏蔽不同的数据存储访问技术，以及与外部系统整合技术的差异性。

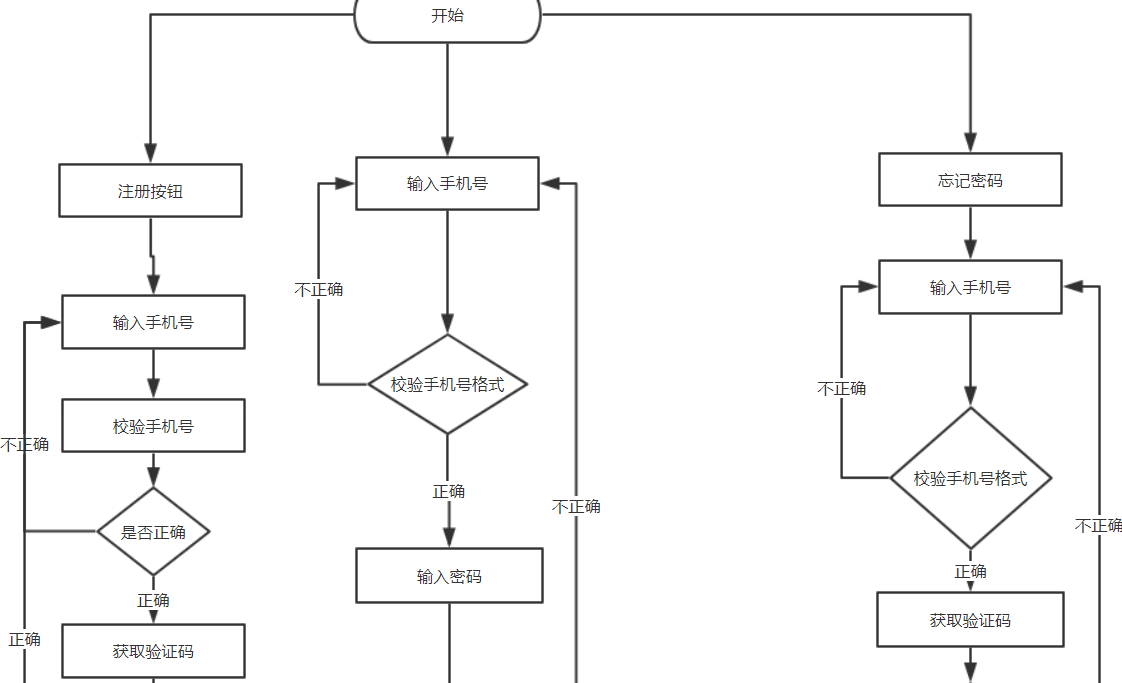
### 3.2.6 数据层(Resource Tier)

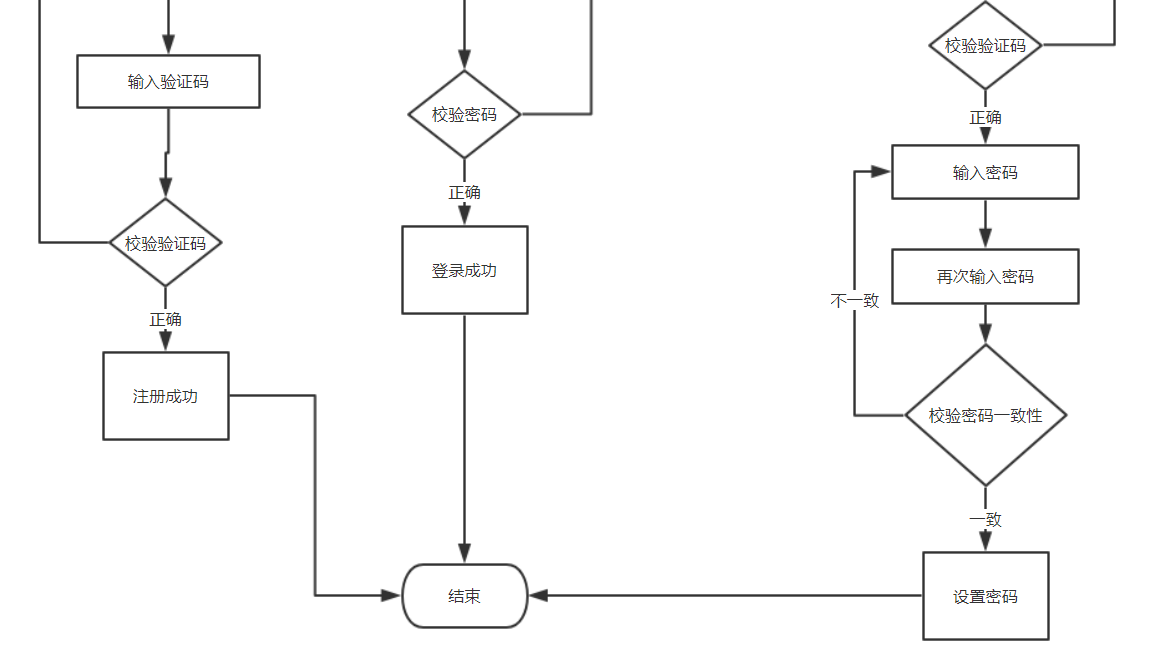
资源层主要指数据库、文件系统和外部。系统采用Mysql作为数据库系统。

## 3.3功能划分及处理流程



|  |  |  |  |
| --- | --- | --- | --- |
| 功能 | 目的 | 具体操作 | 操作结果 |
| 身份验证 | 输入管理员信息（账号和密码）、输入用户信息（登录名和密码） | 管理员身份验证：根据输入的账号在数据库中查询密码，若匹配则验证通过  用户信息验证：使用用户的登录名和密码在网上模拟登录，若网页返回200则验证通过 | 输入验证结果 |
| 推荐景点 | 用户目的地 | 在数据库中查找目的地是否存在，然后查询目的地的所有景点，返回查询结果 | 景点列表 |
| 查看目的景点票价 | 景点名称 | 在数据库中查找目的景点是否存在否存在，然后查询目的景点票价，返回查询结果 | 目的景点票价信息 |
| 查看景点附近酒店价格 | 景点名称 | 在数据库中查找目的景点是否存在否存在，然后查询目的景点附近酒店价格，返回查询结果 | 景点附近酒店价格 |
| 查看附近景点信息 | 景点名称 | 在数据库中查找目的景点是否存在否存在，然后查询目的景点附近景点信息，返回查询结果 | 附近景点信息 |
| 查看最近游客量 | 景点名称 | 在数据库中查找目的景点是否存在否存在，然后查询目的景点最近游客量，返回查询结果 | 最近游客量信息 |
| 查看乘车时间和票价 | 出发地和目的地和交通方式 | 在数据库中查找出发地和目的地是否存在否存在，然后查询目的地和出发地之间的车票信息中的乘车时间，返回查询结果 | 乘车时间和票价 |
| 查看推荐路线和推荐交通工具 | 出发地和目的地和路线信息 | 在数据库中查找出发地和目的地是否存在否存在，然后查询目的地和出发地之间的路线和输入的路线信息是否匹配，返回查询结果 | 推荐路线和交通工具 |
| 管理员维护数据 | 维护数据结果 | 将管理员输入的维护数据结果在数据库中进行更新 | 数据库更新成功 |
| 管理员录入信息结果 | 录入信息结果 | 将管理员输入的录入信息结果在数据库中进行增加 | 数据库更新成功 |





## 3.4 设计原则

1.坚持高起点、高标准，坚持实用性与先进性相统一的原则。

2.在系统建设中严格遵守相关标准，坚持标准化和一致性的原则。

3.实用性第一，兼顾先进性、可扩展性和兼容性的原则。

4.坚持适用性原则，采用成熟、完善的软件工程管理技术和手段保证系统的可操作性、易用性、扩展性和可管理性。

5.注重现实应用与兼顾长远利益的原则。

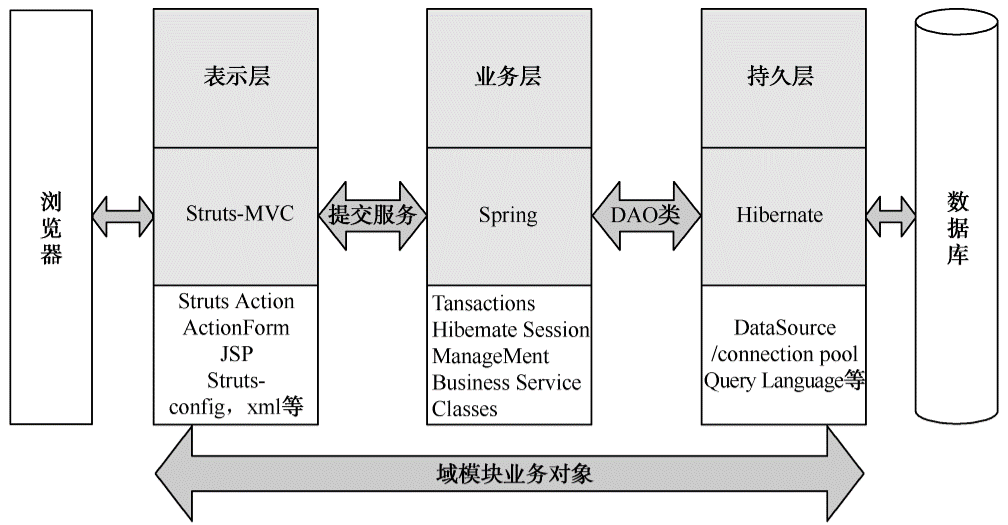
6.保证系统采用技术的成熟性和先进性。使用业界较为成熟的工程技术手段和先进的系统架构和中间件技术，以保证本系统具有成熟的品质和先进性。

7.系统要兼顾未来需求，具有良好的扩充能力以及对外接口。

8.充分考虑建立系统的可移植性。

## 3.5 体系架构

本系统的体系架构如图所示：

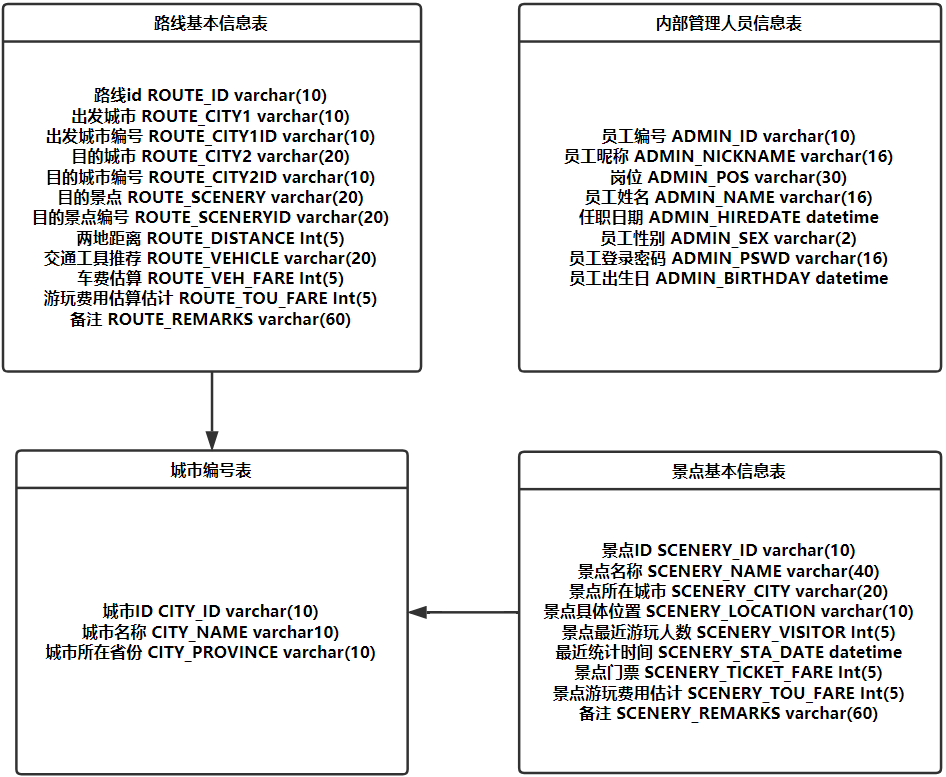


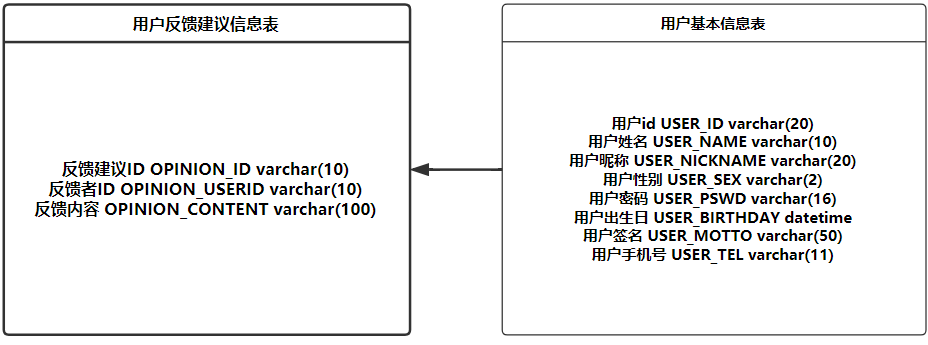
# 4 数据结构设计

## 4.1主要表结构

|  |  |  |
| --- | --- | --- |
| 序号 | 表 | 功能说明 |
| 1 | tour\_userinfo | 用户基本信息表 |
| 2 | tour\_admininfo | 内部管理人员信息表 |
| 3 | tour\_route | 路线基本信息表 |
| 4 | tour\_scenery | 景点基本信息表 |
| 5 | tour\_useropinion | 用户反馈建议信息表 |
| 6 | tour\_city | 城市编号表 |

## 4.2 E-R图设计





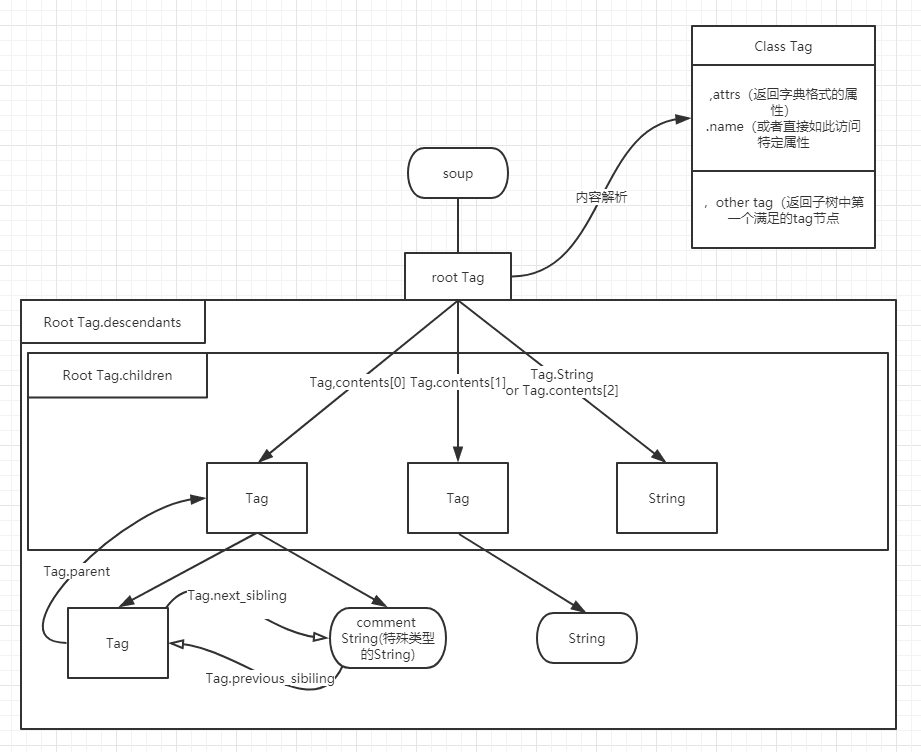
# 5 爬虫结构设计

## 5.1爬虫语言及方法库

    网络爬虫采用了在爬虫方面比较热门的python语言，使用了其中的requests和Beautifulsoup库，requests是访问网页的基础，而Beautifulsoup则是解析网页的重要工具。  
    BeautifulSoup是具有针对性，使用便捷的，能从HTML或XML中提取数据的Python库，将requests网页返回的网页内容解析成一个树形，以标签和文本为基本类型，一个标签包含其他标签，被解析为一个标签是另一个标签的子树。那么，最终整个HTML文档被解析为一棵树的形式。  
BeautifulSoup包括四种基对象类型Tag，NavigableString，BeautifulSoup，Comment。

目前暂时的表结构为：景点或城市名称，景点评价或推荐

通过从携程网攻略里各大城市或者景点的网址来爬取相关内容



# 6 接口设计

## 6.1用户接口

在用户界面部分，根据需求分析的结果，用户需要一个用户友善界面。在界面设计上，应做到简单明了，易于操作，并且要注意到界面的布局，应突出的显示重点以及出错信息。外观上也要做到合理化，考虑到用户多对WINDOW 风格较熟悉，应尽量向这一方向靠拢。在设计语言上，已决定使用 JAVA 进行编程，在界面上可使用JAVA所提供的可视化组件，向WINDOWS 风格靠近。 其中服务器程序界面要做到操作简单，易于管理。

总的来说，系统的用户界面应作到可靠性、简单性、易学习和使用。

## 6.2外部接口

### 6.2.1、软件接口

服务器程序可使用Java 提供的对 MySQL 的接口，进行对数据库的所有访问。

服务器程序上可使用MySQL 的对数据库的备分命令，以做到对数据的保存。

在网络软件接口方面，使用一种无差错的传输协议，采用滑动窗口方式对数据进行网络传输及接收。

## 6.3内部接口

⮚账户信息管理模块。对其他模块提供对用户账户信息的查询接口。

⮚出行信息管理模块。对其他模块提供对用户出行信息查询的接口，包括按用户名查询、按目的地查询和按出行时间查询三种。

⮚反馈信息管理模块。对其他模块提供对反馈信息查询的接口，包括按用户名查询、按出行时间查询等方法。

# 7 运行设计

## 7.1运行模块组合

客户机在用户登录时启动接收数据模块和账户信息管理模块、通过这两个模块之间的调用，读取并比对用户输入的用户名和密码来完成用户登录这一功能。在用户输入出行信息时，调用接收数据模块、账户信息管理模块、出行信息管理模块，在得到充分的数据之后，调用网络传输模块，将数据通过网络送到服务器，通过服务器传输给数据库。在接收到来自服务器的返回信息之后，调用数据输出模块，对信息进行处理，产生相应的输出。

服务器在接收到来自客户机的数据之后，调用数据处理/查询模块对数据库进行访问，在完成账户信息和相应出行信息的对比后，将数据存储至数据库.之后调用数据输出模块，向客户机返回信息，引导客户机完成信息提示。

## 7.2运行控制

运行控制将会考虑接口之间以及模块之间的相互传递的关系，以及客户机和服务器、浏览器之间数据传输和模块运行之间的次序，这影响到软件的性能，其中包括运行速度、相应速度等多个方面。

网络传输方面，客户机向数据库发送信息之后，将等待下一次来自用户的账户或者出行信息、或者是服务器的确认信号。在完成接收之后，反馈给对应服务器或者用户接口。服务器在接到数据后发送确认信号，先通过数据处理/查询模块对数据进行特殊处理，之后使用相同模块访问数据库后，将返回信息送回客户机，并等待确认。

## 7.3运行时间

软件的运行时间是一个软件是否优秀的一个重要的评判标准，我们的项目预计要求达到在保持吞吐量的同时尽量保持尽可能少的响应时间，最长响应时间应该不超过一秒而平均响应时间和服务器的响应时间相关，要求不超过1000ms。其次，系统应该拥有足够用户使用的容量，预期是当众多用户同时使用的情况能够保持系统的高效运行。在面对降级模式时，我们的项目应该要有一份良好的适用性，在系统降级的情况下，应该能同时保有良好的运行速度和响应时间，在内存占用上，我们通过减少部分额外功能的内存占用来保证核心功能的使用，使得系统的运行时间保持在一个正常的范围。

# 8 系统安全性设计

在系统建设中，保障数据库的安全是一个非常重要的组成部分。数据库中存放着大量的重要信息，还有一些宏观的决策数据库。这些信息一旦遭到攻击或丢失将会给个人、单位以及国家带来不可挽回的损失。

## 8.1 数据库核心数据

我们首先需要确定有哪些数据是我们软件功能的核心：

1. 用户所输入的账户信息。

2. 用户所输入的出行相关信息。

3. 经过数据处理之后的出行相关信息。

4. 用户对我们所给出行方案的反馈信息。

5. 经过数据库分析处理后所给出的出行方案。

## 8.2 数据库面临的威胁

我们在使用数据库时，经常会遇到以下几种威胁：

1. 由于数据库管理员的不当操作引发的突发性意外；
2. 由于早期数据库暴露导致的外部恶意撞库；
3. 由于黑客入侵导致的数据库信息泄露；
4. 由于病毒所导致的数据库信息泄露。

这几种情况无疑都会导致十分严重的后果，而我们在构筑设计数据库时就应当考虑到这几种情况的发生，并提早做好预防措施或者补救手段。

## 8.3 数据库的安全策略

数据库安全策略是涉及信息安全的高级指导方针，这些策略根据用户需要、安装环境、建立规则和法律等方面的限制来制定。

数据库系统的基本安全性策略主要是一些基本性安全的问题，如访问控制、伪装数据的排除、用户的认证、可靠性，这些问题是整个安全性问题的基本问题。数据库的安全策略主要包含以下几个方面：

1、保证数据库存在安全

数据库是建立在主机硬件、操作系统和网络上的系统，因此要保证数据库安全，首先应该确保数据库存在安全。数据库管理员应当在每一次数据更新之后完成对整个软件数据库的保存，以防止因为管理员的不当操作或者突发性断电等情况导致数据内容的丢失。而数据库管理员应当在发生突发性情况的短时间内完成对数据库备份的读取和二次使用，以保证软件正常功能的尽快恢复。

2、保证数据库使用安全

数据库使用安全是指数据库的完整性、保密性和可用性。其中，完整性是整个软件得以正常使用的关键，是数据库设计时的重中之重，需要保证从各个表到数据库整体都保有良好的完整性。保密性是数据库安全最重要的一环，数据库可能因为各种各样的情况导致信息泄露，这就需要在数据库设计时，要求良好的保密性来使得面对黑客和病毒的入侵能够及时完成数据的再加密。在面对撞库时，要求数据库管理员能够及时的完善数据库加密算法以挽回损失。可用性保证了各种用户都能相同的使用数据库内的数据，这与保密性相悖，但却是数据库十分重要的一环。

# 9 系统异常处理设计

## 9.1出错信息

|  |  |  |
| --- | --- | --- |
| 出错名称 | 系统输出信息 | 处理方法 |
| 用户名输入错误 | “您的账户信息不存在，请确认！” | 返回登录页面 |
| 前三次密码错误 | “账号或密码错误，请重新输入！” | 返回登录页面 |
| 密码错数次数超过三次 | “账号或密码错误，请重新输入！”输出验证码 | 返回登录页面，用户需要输入正确验证码登录 |
| 验证码输入错误 | “验证码输入错误，请重新输入！” | 返回登录界面，验证码刷新 |
| 不允许为空的输入框输入空字符 | “请输入xxx（对应输入框内容）！” | 返回原输入页面 |
| 不允许为空的选择框不选择 | “该选项不允许为空！” | 返回原选择页面 |
| 输入金额不是数字或者为负数 | “您输入的金额不符合格式，请重新输入！” | 返回原输入页面 |
| 输入金额超出范围 | “您输入的金额过高，请重新输入！” | 返回原输入页面 |
| 系统故障 | “服务器维护中！暂停服务！” | 立即启用备用机，恢复故障 |

## 8.2补救措施

数据库在某一次更新之后自动备份，在系统产生故障的同时可能产生数据损毁，直接使用备份数据库和备用机保证大部分用户的正常使用。如果是在数据更新时产生的系统故障，从而导致用户数据的丢失，我们在数据进入数据库之前会将少量数据通过过滤器存储在服务器本地，可以预防这种突发性的系统错误。