目录

[更新历史 2](#_Toc464424859)

[一、 引言 4](#_Toc464424860)

[(一) 编制目的 4](#_Toc464424861)

[(二) 词汇表 4](#_Toc464424862)

[(三) 参考资料 5](#_Toc464424863)

[二、 产品描述 6](#_Toc464424864)

[三、 逻辑视角 7](#_Toc464424865)

[四、 组合视角 9](#_Toc464424866)

[(一) 开发包图 9](#_Toc464424867)

[(二) 运行时进程 14](#_Toc464424868)

[(三) 物理部署 15](#_Toc464424869)

[五、 接口视角 16](#_Toc464424870)

[(一) 模块的职责 16](#_Toc464424871)

[1. 客户端模块视图 16](#_Toc464424872)

[2. 服务器端模块视图 16](#_Toc464424873)

[3. 客户端各层的职责 16](#_Toc464424874)

[4. 服务器端各层的职责 17](#_Toc464424875)

[5. 各层联系 17](#_Toc464424876)

[(二) 用户界面层的分解 18](#_Toc464424877)

[1. 用户界面层模块的职责 19](#_Toc464424878)

[2. 用户界面模块的接口规范 19](#_Toc464424879)

[3. 用户界面模块设计原理 21](#_Toc464424880)

[(三) 业务逻辑层的分解 21](#_Toc464424881)

[1. 业务逻辑层模块的职责 21](#_Toc464424882)

[2. 业务逻辑模块的接口模范 22](#_Toc464424883)

[(四) 数据层的分解 42](#_Toc464424884)

[1. 数据层模块的职责 42](#_Toc464424885)

[2. 数据层模块的接口规范 42](#_Toc464424886)

[六、 信息视角 53](#_Toc464424887)

[(一) 数据持久化对象 53](#_Toc464424888)

[(二) 文件格式 54](#_Toc464424889)

# 更新历史

|  |  |  |  |
| --- | --- | --- | --- |
| **修改人员** | **修改日期** | **修改原因** | **版本号** |
| 冯俊杰 | 2016/10/4 | 最初草稿 |  |
| 全体成员 | 2016/10/10 | 新增逻辑分包 |  |
| 全体成员 | 2016/10/11 | 新增物理分包 |  |
| 冯俊杰 | 2016/10/12 | 修改物理分包 |  |
| 龚尘淼 | 2016/10/12 | 修改逻辑分包 |  |
| 龚尘淼 | 2016/10/12 | 新增promotion、hotel接口规范 |  |
| 董金玉 | 2016/10/12 | 新增member、user接口规范 |  |
| 冯俊杰 | 2016/10/12 | 新增logIn、credit、market接口规范 |  |
| 高源 | 2016/10/12 | 新增order接口规范 |  |
| 董金玉 | 2016/10/13 | member、user接口规范评审后第一次更新 |  |
| 冯俊杰 | 2016/10/13 | logIn、credit、market接口规范评审后第一次更新 |  |
| 高源 | 2016/10/13 | order接口规范评审后第一次更新 |  |
| 冯俊杰 | 2016/10/13 | credit、market接口规范评审后第二次更新 |  |
| 冯俊杰 | 2016/10/13 | 修改物理分包逻辑层依赖关系 |  |
| 龚尘淼 | 2016/10/14 | 修改逻辑图和用户界面跳转图 |  |
| 龚尘淼 | 2016/10/14 | promotion、hotel接口规范评审后第一次更新 |  |
| 冯俊杰 | 2016/10/14 | 修改逻辑分包 |  |
| 高源 | 2016/10/14 | 新增进程图 |  |
| 龚尘淼 | 2016/10/14 | promotion、hotel接口规范评审后第二次更新 |  |
| 董金玉 | 2016/10/14 | member、user接口规范评审后第二次更新 |  |
| 高源 | 2016/10/15 | order接口规范评审后第二次更新 |  |
| 冯俊杰 | 2016/10/15 | 整合、合并文档 | V0.1 |
| 龚尘淼 | 2016/10/15 | promotion、hotel接口规范评审后第三次更新 | V0.2 |
| 高源 | 2016/10/15 | 新增数据持久化对象PO | V0.3 |
| 高源 | 2016/10/16 | 修改更新数据持久化对象PO | V0.3.1 |
| 全体人员 | 2016/10/16 | 修改细节 | V0.4 |
| 高源 | 2016/10/16 | 排版 | V1.0 |
| 全体人员 | 2016/10/25 | 修改VO、PO | V1.1 |
| 全体人员 | 2017/1/1 | 最后修改接口规范 | V2.0 |
| 董金玉 | 2017/1/1 | 修改遗漏的用例图 | V2.1 |
| 冯俊杰 | 2017/1/2 | 根据实际数据库表项修改信息视角 | V2.2 |

# 引言

## 编制目的

本报告详细完成对互联网酒店预订系统的概要设计，达到指导详细设计和开发的目的，同时实现测试人员及用户的沟通。

本报告面向开发人员、测试人员及最终用户编写，是了解系统的导航。

## 词汇表

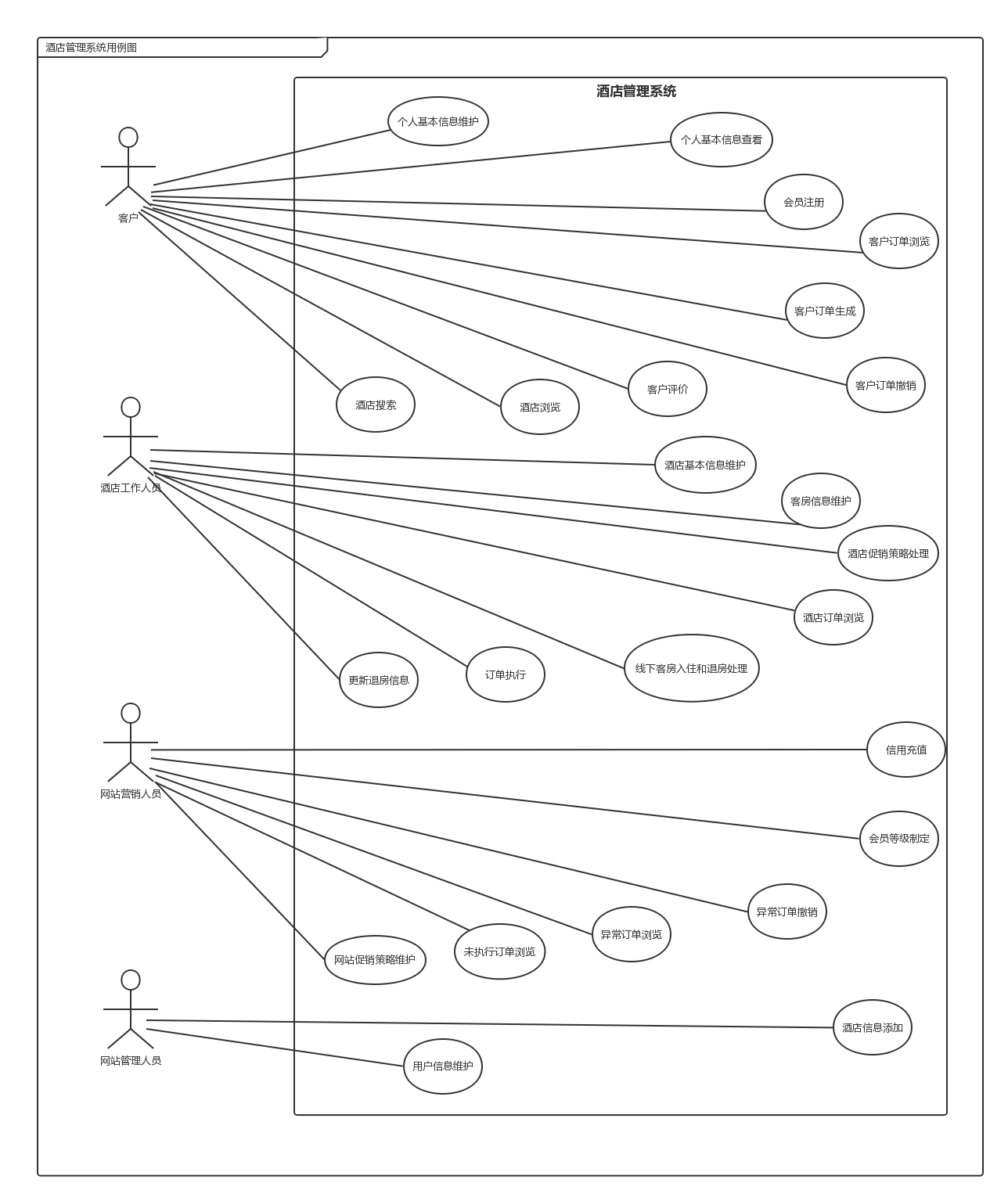
|  |  |
| --- | --- |
| **缩写或单词** | **解释** |
| HRS | 互联网酒店预订系统 |
| order | 订单 |
| hotel | 酒店 |
| guest | 客户 |
| hotelWorker | 酒店工作人员 |
| webMarketer | 网站营销人员 |
| webManager | 网站管理人员 |
| signUp | 注册 |
| logIn | 登录 |
| UI | 界面 |
| BL | 逻辑 |
| Data | 数据 |
| promotion | 促销策略 |
| credit | 信用值 |
| member | 会员 |
| market | 营销 |

## 参考资料

1. 互联网酒店预订系统用例文档 V1.6.3
2. 互联网酒店预订系统需求规格说明文档 V1.5.1
3. 丁二玉，刘钦.计算与软件工程（卷二）[M]机械工业出版2012：134—182
4. IEEE std 1471-2000

# 产品描述

参考《互联网酒店预订系统用例文档》和《互联网酒店预订系统需求规格说明文档》中对产品的概括描述。互联网酒店预订系统主要功能见用例图如下。



# 逻辑视角

互联网酒店预订系统中，选择了分层体系结构的风格，将系统分为3层（展示层、业务逻辑层、数据层）能够很好的示意整个高层抽象。展示层包括GUI页面的实现，业务逻辑层包含业务逻辑处理的实现，数据层负责数据的持久化和访问。分层体系结构的逻辑视角和逻辑设计方案如图1和图2所示。



**图1：分层结构**



**图2：逻辑设计方案**

# 组合视角

## 开发包图

[软工2 166]与抽象的逻辑设计相比，实现物理设计要考虑更多的实现细节，这些细节有：

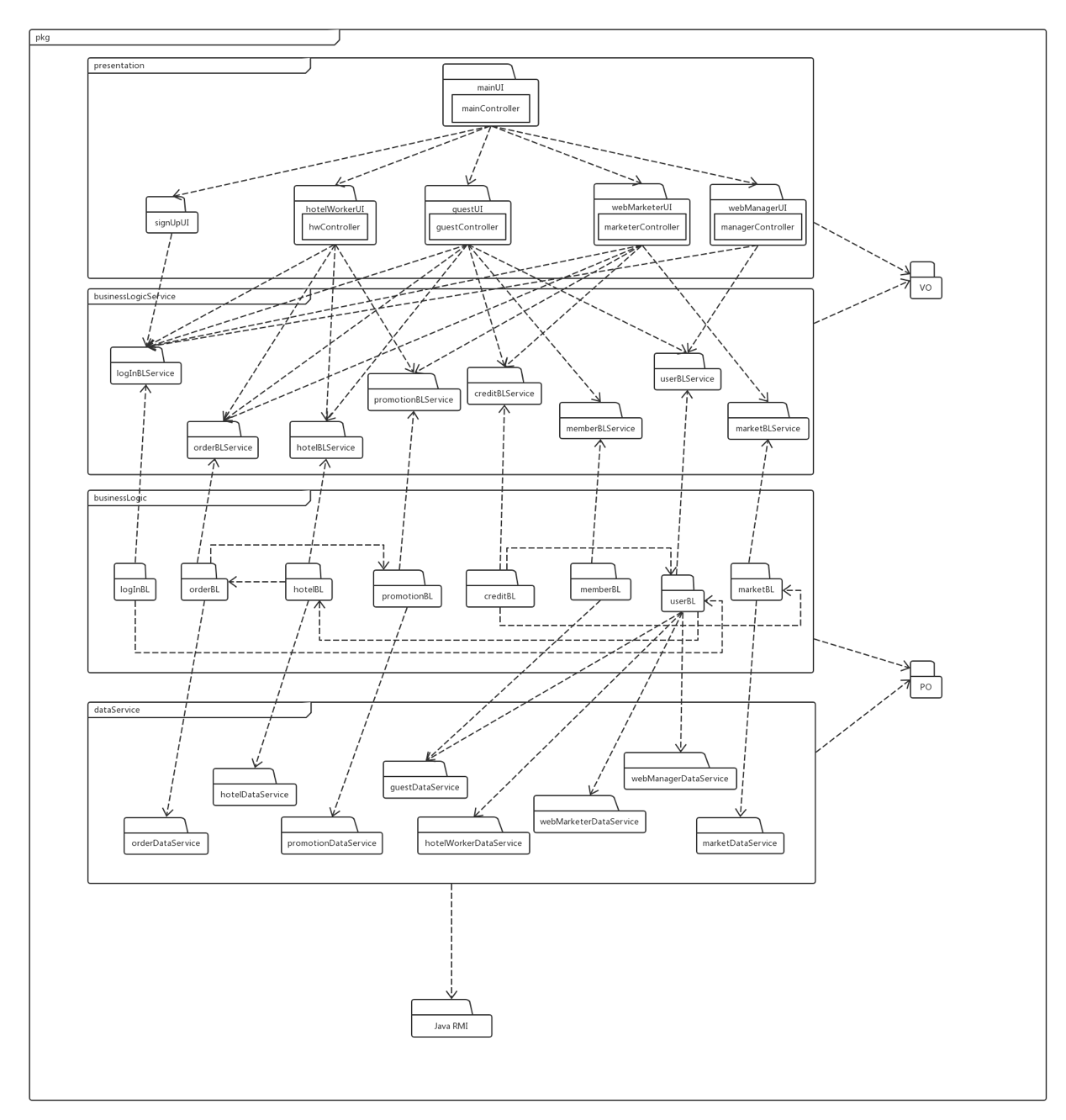
1. 所有的presentation层开发包都需要使用图形类型建立界面，都要依赖于图形界面类库包。
2. 在presentation层实现时，由mainUI包负责不同参与者在整个页面之间的跳转逻辑。其他各包负责各自页面自身的功能，并由各自的controller负责自身内部页面之间的跳转逻辑。
3. presentation层与businessLogic层被置于客户端，data层被置于服务器端，那么businessLogic层的开发包不可能依赖于data层的开发包。使用RMI技术，将data层开发包分解为置于客户端的dataService接口包和置于服务器端的data层开发包。这样一来，businessLogic层开发包依赖于dataService包，dataService包依赖于RMI类库包。
4. 所有的data层开发包都需要进行数据持久化（例如读写数据库、读写文件等），所以它们会有一些重复代码，可以将重复代码独立为新的开发包，然后所有的data层开发包都依赖于dataBaseUtility。dataBaseUtility会依赖于JDBC类库包。
5. 在分层风格的典型设计中，不希望高层直接依赖于低层，而是为低层建立接口包，实现依赖倒置原则，所以应该调整为：各presentation层开发包（调用）依赖于logic层接口包businessLogicService包，businessLogic层开发包也依赖于（实现了）businessLogic层接口包businessLogicService包。
6. 在分层风格的典型设计中，presentation层与logic层之间、logic层与data层之间可能会传递复杂数据对象，那么相邻两层都需要使用数据对象声明，所以需要将数据对象声明独立为开发包——VO包和PO包。其中VO包负责在presentation层与logic层之间传递，PO包负责在logic层与data层之间传递。
7. 使用依赖倒置原则消除包的循环依赖现象，将循环依赖变为单向依赖：
8. orderBL和hotelBL:将部分orderBL类抽象接口orderInfo置入hotelBL包，这样orderBL单向依赖于hotelBL（实现接口+调用）

8）在logic层中，初始化和业务逻辑层上下文的工作被分配到utility包中。

界面层依据参与者分为main、注册界面和4个参与者主界面；逻辑层主要依据逻辑分包的结果，并添加logInBL，负责用户的登录和客户的注册；数据层进行整合合并，依照参与者、订单、酒店的实际情况进行分类。

|  |  |
| --- | --- |
| 开发（物理）包 | 依赖的其他包 |
| mainUI | signUpUI, guestUI, hotelWorkerUI, webMarketerUI, webManagerUI, 界面类库包 |
| signUpUI | logInBLService, 界面类库包, VO |
| guestUI | logInBLService, orderBLService, hotelBLService, creditBLService, memberBLService, UserBLService, 界面类库包, VO |
| hotelWorkerUI | logInBLService, orderBLService, hotelBLService, promotionBLService, 界面类库包,VO |
| webMarketerUI | logInBLService, orderBLService, promotionBLService, creditBLService, marketBLService, 界面类库包,VO |
| webManagerUI | logInBLService, UserBLService, 界面类库包,VO |
| logInBLService | VO |
| orderBLService | VO |
| hotelBLService | VO |
| promotionBLService | VO |
| creditBLService | VO |
| memberBLService | VO |
| marketBLService | VO |
| userBLService | VO |
| logInBL | logInBLService, userBL, PO |
| orderBL | orderBLService, orderDataService, hotelBL, promotionBL, creditBL, userBL, PO |
| hotelBL | hotelBLService, hotelDataService, orderBL, promotionBL, PO |
| promotionBL | promotionBLService, promotionDataService, PO |
| creditBL | creditBLService, orderBL, userBL, marketBL, PO |
| memberBL | memberBLService, guestDataService, PO |
| marketBL | marketBLService, marketDataService, PO |
| userBL | userBLService, guestDataService, hotelWorkerDataService, webMarketerDataService, webManagerDataService, hotelBL, PO |
| orderDataService | Java RMI, PO |
| hotelDataService | Java RMI, PO |
| guestDataService | Java RMI, PO |
| hotelWorkerDataService | Java RMI, PO |
| webMarketerDataService | Java RMI, PO |
| webManagerDataService | Java RMI, PO |
| promotionDataService | Java RMI, PO |
| marketDataService | Java RMI, PO |
| orderData | orderDataService, PO |
| hotelData | hotelDataService, PO |
| guestData | guestDataService, PO |
| hotelWorkerData | hotelWorkerDataService, PO |
| webMarketerData | webMarketerDataService, PO |
| webManagerData | webManagerDataService, PO |
| promotionData | promotionDataService, PO |
| marketData | marketDataService, PO |
| VO |  |
| PO |  |
| dataBaseUtility | java.JDBC |
| 界面类库包 |  |
| JAVA RMI |  |
| java.swing,java.awt,java.2D |  |

下图为互联网酒店预订系统客户端开发包图：



下图为互联网酒店预订系统服务端开发包图：



## 运行时进程

在互联网酒店预订系统中，会有多个客户端进程和一个服务器端进程，其进程图如下图所示。结合部署图，客户端进程是在客户端机器上运行，服务器端进程是在服务器端机器上运行。数据库的自动更新（如过了订单的最晚入住时间会将订单置为异常）会在服务器端自动进行，作为另一个进程。



## 物理部署

互联网酒店预订系统中客户端构件是放在客户端机上，服务器端构件是放在服务器端机器上。在客户端节点上，还要部署RMIStub构件。由于Java RMI 构件属于JDK 1.8的一部分。所以，在系统JDK环境已经设置好的情况下，不需要独立部署。部署图如下图所示。



# 接口视角

## 模块的职责

### 客户端模块视图



### 服务器端模块视图



### 客户端各层的职责

|  |  |
| --- | --- |
| 层 | 职责 |
| 启动模块 | 负责初始化网络通信机制，启动用户界面 |
| 用户界面层 | 基于窗口的互联网预订系统客户端用户界面 |
| 业务逻辑层 | 对于用户界面的输入进行响应并进行业务处理逻辑 |
| 客户端网络模块 | 利用java RMI机制查找RMI服务 |

### 服务器端各层的职责

|  |  |
| --- | --- |
| 层 | 职责 |
| 启动模块 | 负责初始化网络通信机制，启动用户界面 |
| 数据层 | 负责数据的持久化及数据访问接口 |
| 服务器端网络模块 | 利用Java RMI机制开启RMI服务，注册RMI服务 |

### 各层联系

每一层只是使用下方直接接触的层。层与层之间仅仅是通过接口的调用来完成的。层之间调用的接口如下图所示。

|  |  |  |
| --- | --- | --- |
| **接口** | **服务调用方** | **服务提供方** |
| logInBLService  orderBLService  hotelBLService  promotionBLService  creditBLService  memberBLService  marketBLService  userBLService  sourceBLService | 客户端  展示层  （presentation） | 客户端  业务逻辑层  （businessLogic） |
| orderDataService  hotelDataService  guestDataService  hotelWorkerDataService  webMarketerDataService  webManagerDataService  promotionDataService  marketDataService  sourceDataService | 客户端  业务逻辑层  （businessLogic） | 服务器端  数据层  （data） |

## 用户界面层的分解

根据需求，系统存在37个用户界面（加粗为主界面）： **登录界面**、**注册界面**、**客户主界面**、个人基本信息界面、个人基本信息维护界面、会员注册界面、未评价订单界面、订单评价界面、个人订单概况列表界面、个人订单详情界面、个人订单撤销界面、商圈地址选定界面、酒店概况列表界面、订单生成界面、酒店搜索界面、酒店详情界面、会员注册界面、**酒店工作人员界面**、酒店信息界面、酒店信息维护界面、客房信息维护界面、酒店促销策略界面、订单列表界面、订单详情界面、剩余客房信息界面、线下客户入住和退房处理界面、**网站营销人员界面**、用户编号录入界面、信用充值界面、异常订单概况列表界面、异常订单详情界面、异常订单撤销界面、网站促销策略界面、会员等级制定界面、**网站管理人员界面**、会员信息界面、会员信息维护界面、酒店信息添加界面。

界面跳转如下图所示。



### 用户界面层模块的职责

|  |  |
| --- | --- |
| 模块 | 职责 |
| MainFrame | 界面frame，负责界面的显示和界面的跳转 |

### 用户界面模块的接口规范

1. **用户界面层模块的接口规范**

|  |  |  |
| --- | --- | --- |
| mainUI | 语法 | init(args:String[]) |
| 前置条件 | 无 |
| 后置条件 | 显示Frame |
| logInUI.logIn | 语法 | void logIn(userVO) |
| 前置条件 | 得到界面的用户名密码 |
| 后置条件 | 显示对应的界面 |
| logInUI.register | 语法 | userVO register(userVO) |
| 前置条件 | 从界面得到部分信息 |
| 后置条件 | 返回注册的用户名密码 |
| guestUI.guestInfo.view | 语法 | Void openGuestInfo(userID) |
| 前置条件 | 客户已经登陆 |
| 后置条件 | 显示客户基本信息 |
| guestUI.guestInfo.modify | 语法 | void modifyGuestInfo(userVO) |
| 前置条件 | 客户修改了正确的个人信息 |
| 后置条件 | 显示修改后的信息 |
| guestUI.order.view | 语法 | Void searchOrder(guestID) |
| 前置条件 | 客户已经登陆 |
| 后置条件 | 显示order的信息 |
| guestUI.order.undo | 语法 | void undo(orderID) |
| 前置条件 | 客户查看订单 |
| 后置条件 | 将未执行的订单撤销 |
| guestUI.order.create | 语法 | Void createOrder(orderVO) |
| 前置条件 | 填写相应的订单内容 |
| 后置条件 | 系统生成订单 |
| guestUI.searchHotel | 语法 | List<HotelVO> searchHotel( ) |
| 前置条件 | 客户登陆系统 |
| 后置条件 | 显示符合条件的酒店 |
| guestUI.viewHotel | 语法 | hotelVO openHotelInfo( hotel ID) |
| 前置条件 | 客户查找到想要的酒店 |
| 后置条件 | 显示对应酒店的信息 |
| guestUI.order.comment | 语法 | Void comment(orderVO) |
| 前置条件 | 客户查看已执行的订单 |
| 后置条件 | 保存对订单的评价 |
| guestUI.member.register | 语法 | Void commitMember(memberVO) |
| 前置条件 | 客户尚未注册会员 |
| 后置条件 | 更改客户的客户会员信息 |
| hotelWorkerUI.HotelInfo.  Modify | 语法 | void hotelInfoModify(hotelVO) |
| 前置条件 | 酒店工作人员登陆 |
| 后置条件 | 保存酒店维护后的信息 |
| hotelWorkerUI.RoomInfo.  Modify | 语法 | void roomInfoModify (list<roomVO>) |
| 前置条件 | 酒店工作人员登陆系统 |
| 后置条件 | 对房间的信息进行保存 |
| hotelWorkerUI.promotion.  update | 语法 | void updatePromotion（list<promotion>） |
| 前置条件 | 酒店工作人员登陆 |
| 后置条件 | 系统保存更改后的策略 |

|  |  |  |
| --- | --- | --- |
| hotelWorkerUI.order.  execute | 语法 | void executeOrder(orderVO) |
| 前置条件 | 酒店工作人员执行未执行的订单 |
| 后置条件 | 改变酒店状态并更新房间信息 |
| hotelWorkerUI.order.  checkout | 语法 | void checkout(orderVO) |
| 前置条件 | 酒店工作人员退房已执行订单 |
| 后置条件 | 改变酒店房间信息 |
| hotelWorkerUI.offline | 语法 | void offline(roomInfoVO) |
| 前置条件 | 线下有人入住 |
| 后置条件 | 改变酒店的客房信息 |
| hotelWorkerUI.order.  view | 语法 | list<orderVO> searchOrder( ) |
| 前置条件 | 酒店工作人员登录 |
| 后置条件 | 显示酒店的订单 |

|  |  |  |
| --- | --- | --- |
| webMarketer.promotion.  update | 语法 | void updatePromotion (list<promotionVO>) |
| 前置条件 | 网站营销人员登录 |
| 后置条件 | 保存更改后的网站营销策略 |
| webMarketer.  abnormalOrder.  undo | 语法 | list<orderVO> undoOrder (double persent ) |
| 前置条件 | 网站营销人员登录 |
| 后置条件 | 撤销异常订单，按百分比返还信用值 |
| webMarketerUI.charge | 语法 | void charge( guestID ) |
| 前置条件 | 网站营销人员登录 |
| 后置条件 | 给对应的客户充值信用值 |
| webMarketerUI.  memberForm  ulation.update | 语法 | void memberUpdate (list<memberVO>) |
| 前置条件 | 营销人员登录 |
| 后置条件 | 更新会员等级策略 |

|  |  |  |
| --- | --- | --- |
| webManager.guestInfo.  modify | 语法 | Void guestModify(guestVO) |
| 前置条件 | 管理人员找到需要更改的客户信息 |
| 后置条件 | 系统保存更改后的客户信息 |
| webManager.hotelWorInfo.  modify | 语法 | void hotelWorkerModify (hotelWorkerVO) |
| 前置条件 | 管理人员找到需要更改的酒店工作人员信息 |
| 后置条件 | 系统保存更改后的酒店工作人员信息 |
| webManager.marketerInfo.  modify | 语法 | void webMarketerModify (webMarketerVO) |
| 前置条件 | 管理人员找到需要更改的网站营销人员信息 |
| 后置条件 | 显示webMarketerPanel |
| WebManagerUI.addHotel | 语法 | void addHotel (hotelVO) |
| 前置条件 | 管理人员登录系统 |
| 后置条件 | 保存新添加的酒店 |

1. **用户界面层模块需要的服务接口**

|  |  |
| --- | --- |
| 服务名 | 服务 |
| businessLogicService.logInBLService | 登录界面和客户注册的业务逻辑接口 |
| businessLogicService.orderBLService | 负责处理与订单相关的接口 |
| businessLogicService.hotelBLService | 负责处理与酒店相关的接口 |
| businessLogicService.promotionBLService | 负责处理与促销策略相关的接口 |
| businessLogicService.creditBLService | 负责处理与信用值相关的接口 |
| businessLogicService.memberBLService | 负责处理与会员相关的接口 |
| businessLogicService.userBLService | 负责处理与所有用户信息相关的接口 |
| businessLogicService.marketBLService | 负责处理与会员等级制度信息相关的接口 |
| businessLogicService.source BLService | 负责处理与资源文件信息相关的接口 |

### 用户界面模块设计原理

用户界面拟用java的Javafx库来实现。

## 业务逻辑层的分解

业务逻辑层包括多个针对界面及业务逻辑处理对象。例如logIn对象负责处理所有用户（客户、酒店工作人员、网站营销人员、网站管理人员）的登录和客户的注册的业务逻辑。业务逻辑层的设计如下图所示：



### 业务逻辑层模块的职责

|  |  |
| --- | --- |
| 模块 | 职责 |
| logInBL | 负责提供所有用户的登录和客户的注册的服务 |
| orderBL | 负责处理与订单相关的服务 |
| hotelBL | 负责处理与酒店相关的服务 |
| promotionBL | 负责处理与促销策略相关的服务 |
| creditBL | 负责处理与信用值相关的服务 |
| memberBL | 负责处理与会员相关的服务 |
| userBL | 负责处理与所有用户信息相关的服务 |
| marketBL | 负责处理与会员等级制度信息相关的服务 |
| sourceBL | 负责处理与资源文件信息相关的服务 |

### 业务逻辑模块的接口模范

1. **logIn模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| logInBL.guestSignUp | 语法 | public GuestVO guestSignUp (UserVO guestVO) |
| 前置条件 | 客户信息符合规范要求 |
| 后置条件 | 系统返回是否成功添加客户；是则进入系统的客户主界面，否则提示添加失败 |
| logInBL.logIn | 语法 | public UserType logIn (String userID, String password) |
| 前置条件 | 用户输入的用户编号和密码符合规范 |
| 后置条件 | 用户被认证并进入其相应的主界面 |
| logInBL.logOut | 语法 | public void logOut (String webManager) |
| 前置条件 | 无 |
| 后置条件 | 系统退出，进入登录主界面 |
| 需要的接口（需接口） | | |
| 服务名 | **服务内容** | |
| userBL.add(UserVo newUserVo, UserType userType) | 新增该用户并持久化保存该用户的数据 | |
| userBL.getLogInfo(String userID, UserType userType) | 返回与该ID对应的user密码 | |

1. **order模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| orderBL.getAllOrderGenerals | 语法 | public Iterator<OrderGeneralVO> getAllOrderGenerals(String userID, UserType userType) |
| 前置条件 | 客户／酒店工作人员已经登录 |
| 后置条件 | 无 |
| orderBL.getSpecialOrderGenerals | 语法 | public Iterator<OrderGeneralVO> getSpecialOrderGenerals(String userID, UserType userType, OrderState orderState) |
| 前置条件 | 客户／酒店工作人员已经登录 |
| 后置条件 | 无 |
| orderBL.getOrderDetail | 语法 | public OrderVO getOrderDetail(String orderID) |
| 前置条件 | 客户／酒店工作人员／网站营销人员已经登录 |
| 后置条件 | 无 |
| orderBL.getEvaluations | 语法 | public Iterator<HotelEvaluationVO> getEvaluations(String hotelID) |
| 前置条件 | 客户／酒店工作人员已经登录 |
| 后置条件 | 无 |
| orderBL.getCalculatedPrice | 语法 | public int getCalculatedPrice (PreOrderVO preOrderVO) |
| 前置条件 | 客户已经登录，填写订单的内容符合规则且完整 |
| 后置条件 | 系统生成未执行订单并持久化增加该客户的订单数据 |
| orderBL.createOrder | 语法 | public ResultMessage createOrder(OrderVO orderVO) |
| 前置条件 | 客户已经登录，填写订单的内容符合规则且完整 |
| 后置条件 | 系统生成未执行订单并持久化增加该客户的订单数据 |
| orderBL.undoNormalOrder | 语法 | public ResultMessage undoNormalOrder(String orderID) |
| 前置条件 | 客户已经登录，订单状态为未执行 |
| 后置条件 | 系统更改状态为已执行订单并持久化保存该客户的订单数据 |
| orderBL.addEvaluation | 语法 | public ResultMessage addEvaluation(GuestEvaluationVO evaluationVO) |
| 前置条件 | 客户已经登录，订单状态为未执行 |
| 后置条件 | 系统更改状态为已执行订单并持久化保存该客户的订单数据 |
| orderBL.getAllGuestCommentOrderGeneral | 语法 | public Iterator<OrderGeneralVO> getAllGuestCommentOrderGeneral(String guestID, boolean hasCommented) |
| 前置条件 | 客户已经登录 |
| 后置条件 | 无 |
| orderBL.getMyOrdersOfThisHotel | 语法 | public Iterator<OrderGeneralVO> getMyOrdersOfThisHotel(String guestID, String hotelID) |
| 前置条件 | 客户已经登录 |
| 后置条件 | 无 |
| orderBL.updateCheckIn | 语法 | public ResultMessage updateCheckIn(CheckInVO checkInVO) |
| 前置条件 | 酒店工作人员已经登录 |
| 后置条件 | 系统更改此订单状态为已执行订单并持久化保存该客户的订单数据 |
| orderBL.updateCheckOut | 语法 | public ResultMessage updateCheckOut(CheckOutVO checkOutVO) |
| 前置条件 | 酒店工作人员已经登录 |
| 后置条件 | 系统更改此订单状态和酒店剩余房间数量，并持久化保存该客户的订单数据 |
| orderBL.getAllHotelCheckOutOrderGeneral | 语法 | public Iterator<OrderGeneralVO> getAllHotelCheckOutOrderGeneral(String hotelID, boolean hasCheckOut) |
| 前置条件 | 酒店工作人员已经登录 |
| 后置条件 | 无 |
| orderBL.undoAbnormalOrder | 语法 | public ResultMessage undoAbnormalOrder(String orderID, double percent) |
| 前置条件 | 网站营销人员已登录，订单状态为异常 |
| 后置条件 | 系统更改订单状态为已撤销订单并持久化保存该客户的订单数据 |
| orderBL.getAllAbnormalOrderGeneral | 语法 | public List<OrderGeneralVO> getAllAbnormalOrderGeneral(LocalDate date) |
| 前置条件 | 网站营销人员已经登录 |
| 后置条件 | 无 |
| orderBL.getAllUnexecutedOrderGeneral | 语法 | public List<OrderGeneralVO> getAllUnexecutedOrderGeneral(LocalDate date) |
| 前置条件 | 网站营销人员已经登录 |
| 后置条件 | 无 |
| orderBL.getBookedHotel | 语法 | public List<String> getBookedHotels(String guestID) |
| 前置条件 | 无 |
| 后置条件 | 无 |
| orderBL.getOrderState | 语法 | public OrderState getOrderState(String guestID, String hotelID) |
| 前置条件 | 无 |
| 后置条件 | 无 |
| 需要的接口（需接口） | | |
| 服务名 | **服务内容** | |
| orderDataService.getAllGuestOrderGeneral (String guestID) | 根据客户编号查找客户所有订单概况的持久化对象 | |
| orderDataService.getAllHotelOrderGeneral (String hotelID) | 根据酒店编号查找酒店所有订单概况的持久化对象 | |
| orderDataService.getOrderDetail(String orderID) | 根据订单编号查找订单详情的持久化对象 | |
| orderDataService.getEvaluations(String hotelID) | 根据酒店编号查找此酒店评价信息的持久化对象 | |
| orderDataService.createOrder(orderPO OrderPO) | 根据提供的po增加订单的持久化对象 | |
| orderDataService.undoNormalOrder(String orderID) | 根据订单编号修改单一订单持久化对象 | |
| orderDataService.addEvaluation(GuestEvaluationPO guestEvaluationPO) | 修改单一持久化对象订单的评价信息 | |
| orderDataService.updateCheckIn(CheckInPO checkInPO) | 修改单一持久化对象订单的入住信息 | |
| orderDataService.updateCheckOut(CheckOutPO checkOutPO) | 修改单一持久化对象订单的退房信息 | |
| orderDataService.UndoAbnormalOrder(String orderID, double percent) | 根据订单编号和撤销比例修改单一订单持久化对象 | |
| orderDataService.getAllAbnormalOrderGeneral (LocalDate date) | 根据时间查找持久化对象订单的集合 | |
| orderDataService.getAllUnexecutedOrderGeneral (LocalDate date) | 根据时间查找持久化对象订单的集合 | |
| creditBL.addCreditRecord(CreditVO creditVO) | 新增信用记录 | |
| hotelBL.checkIn(String hotelID, RoomType roomType, int roomNum) | 修改酒店剩余房间信息 | |
| hotelBL.checkOut(String hotelID, RoomType roomType, int roomNum) | 修改酒店剩余房间信息 | |
| hotelBL.updateRemainRoomNumForUndoOrder(String hotelID, RoomType roomType, int roomNum) | 更新订单撤销后的酒店剩余房间数 | |
| hotelBL.scoreUpdate(String hotelID,double score) | 更新酒店评分 | |
| promotionBL.getDiscountInSpan(PreOrderVO preOrder) | 根据现有订单状况获取此订单的折扣 | |
| userBL.getSingle(String userID) | 获取客户的单一持久化对象 | |

1. **hotel模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| hotel.getHotelInfo | 语法 | Public HotelVO getHotelInfo (String hotelID); |
| 前置条件 | 已知酒店编号 |
| 后置条件 | 返回编号对应酒店的基本信息 |
| hotel.updateHotelInfo | 语法 | Public ResultMessage updateHotelInfo (HotelVO hotelVO) |
| 前置条件 | 已知需要更新的酒店基本信息 |
| 后置条件 | 系统更新该酒店基本信息，返回是否更新成功 |
| hotel.addHotel | 语法 | Public ResultMessage addHotel (HotelVO hotelVO) |
| 前置条件 | 已知需要添加的酒店基本信息 |
| 后置条件 | 系统添加一条酒店基本信息，返回是否添加成功 |
| hotel.checkInOffline | 语法 | Public ResultMessage checkInOffline (String hotelID,RoomType RoomType,int roomNum) |
| 前置条件 | 线下入住，已知酒店编号，房间类型，房间数量 |
| 后置条件 | 更新该酒店该房型的剩余房间数量，返回是否更新成功 |
| hotel.checkOutOffline | 语法 | public ResultMessage checkOutOffline (String hotelID,RoomType RoomType,int roomNum) |
| 前置条件 | 线下退房，已知酒店编号，房间类型，房间数量 |
| 后置条件 | 更新该酒店该房型的剩余房间数量，返回是否更新成功 |
| hotel.getHotelRoomInfo | 语法 | public Iterator<RoomInfoVO> getHotelRoomInfo (String hotelID) |
| 前置条件 | 已知酒店编号 |
| 后置条件 | 返回该酒店的客房信息集合 |
| hotel.addRoomType | 语法 | public ResultMessage addRoomType(RoomInfoVO roomInfoVO) |
| 前置条件 | 已知一条需要添加的客房信息 |
| 后置条件 | 系统添加一条客房信息，返回是否添加成功 |
| hotel.updateHotelInfo | 语法 | public ResultMessage updateHotelRoomInfo (RoomInfoVO roomInfoVO) |
| 前置条件 | 已知一条需要修改的客房信息 |
| 后置条件 | 系统更新该条客房信息，返回是否更新成功 |
| hotel.getRemainRoomNum | 语法 | public int getRemainRoomNum(String hotelID,RoomType roomType) |
| 前置条件 | 已知酒店编号、房间类型 |
| 后置条件 | 系统返回该酒店该房间类型的剩余房间数量 |
| hotel. getOriginPrice | 语法 | public int getOriginPrice(String hotelID, RoomType roomType) |
| 前置条件 | 已知酒店编号、房间类型 |
| 后置条件 | 返回该酒店该房型的原始价格 |
| hotel.getHotels | 语法 | public Iterator<HotelVO> getHotels(String city,String circle) |
| 前置条件 | 已知城市、商圈名称 |
| 后置条件 | 返回该城市商圈的酒店集合 |
| hotel.sortHotels | 语法 | public Iterator<HotelVO> sortHotels(SortStrategy sortStategy) |
| 前置条件 | 已知排序策略 |
| 后置条件 | 返回按该排序策略排好的酒店集合 |
| hotel.searchHotels | 语法 | public Iterator<HotelVO> searchHotels(List<SearchCriteriaType> searchCriteriaTypes,SearchCriteriaVO vo) |
| 前置条件 | 已知搜索标准类型和搜索标准信息 |
| 后置条件 | 返回经搜索标准筛选后的酒店集合 |
| hotel.scoreUpdate | 语法 | public ResultMessage scoreUpdate(String hotelID,double score) |
| 前置条件 | 已知酒店编号、评分 |
| 后置条件 | 更新酒店评分 |
| hotel.getHotelAddress | 语法 | public Address getHotelAddress(String hotelID) |
| 前置条件 | 已知酒店编号 |
| 后置条件 | 返回酒店的地址信息 |
| hotel.checkIn | 语法 | public ResultMessage checkIn(String hotelID, RoomType roomType, int roomNum) |
| 前置条件 | 线上入住，已知酒店编号、房间类型、房间数量 |
| 后置条件 | 更新该酒店该房型的剩余房间数量 |
| hotel.checkOut | 语法 | public ResultMessage checkOut(String hotelID, RoomType roomType, int roomNum) |
| 前置条件 | 线上退房，已知酒店编号、房间类型、房间数量 |
| 后置条件 | 更新该酒店该房型的剩余房间数量 |
| hotel.updateRemainRoomNumForUndoOrder | 语法 | public ResultMessage updateRemainRoomNumForUndoOrder(String hotelID, RoomType roomType, int roomNum) |
| 前置条件 | 客户撤销订单，已知酒店编号、房间类型、房间数量 |
| 后置条件 | 更新该酒店该房型的剩余房间数量 |
| hotel.getHotelPO | 语法 | public HotelPO getHotelPO(String hotelID) |
| 前置条件 | 已知酒店编号 |
| 后置条件 | 返回酒店信息载体 |
| 需要的接口（需接口） | | |
| 服务名 | **服务内容** | |
| hotelDataService.getHotelInfo (String hotelID) | 返回对应该酒店编号的酒店基本信息 | |
| hotelDataService.getHotels(String city, String circle) | 返回对应该城市、商圈的所有酒店列表 | |
| hotelDataService.updateHotelInfo(HotelPO hotelPO) | 更新数据库里的一条酒店基本信息 | |
| hotelDataService.addHotelInfo(HotelPO hotelPO) | 向数据库里添加一条酒店基本信息 | |
| hotelDataService.getRoomInfo(String hotelID) | 返回对应该酒店的客房信息列表 | |
| hotelDataService.updateRoomInfo(RoomInfoPO roomInfoPO) | 更新数据库里一条客房信息 | |
| hotelDataService.addRoomInfo(RoomInfoPO roomInfoPO) | 向数据库里添加一条客房信息 | |
| hotelDataService.add(HotelInfoPO hotelInfoPO) | 在数据库里添加一条就酒店的信息 | |
| promotion.getPromotions(String userID) | 依据酒店工作人员编号得到数据库里当前酒店的所有策略信息列表 | |
| order.getOrderState([String](eclipse-javadoc:%E2%98%82=ClientSystem/src%5C/main%5C/java%3CbusinessLogic.orderBL%7BOrder.java%E2%98%83Order~getOrderState~QString;~QString;%E2%98%82java.lang.String) guestID, [String](eclipse-javadoc:%E2%98%82=ClientSystem/src%5C/main%5C/java%3CbusinessLogic.orderBL%7BOrder.java%E2%98%83Order~getOrderState~QString;~QString;%E2%98%82java.lang.String) hotelID) | 根据客户编号、酒店编号，返回该客户在该酒店的订单状态 | |
| order.getBookedHotels(String guestID) | 根据客户编号，返回客户所有预定过酒店的编号列表 | |

1. **promotion模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| promotion.getHotelFixedPromotions | 语法 | Public Iterator<HotelFixedPromotionVO> getHotelFixedPromotions(String hotelID) |
| 前置条件 | 已知酒店编号 |
| 后置条件 | 系统返回该酒店的固定促销策略集合 |
| promotion.updateHotelFixedPromotion | 语法 | public ResultMessage updateHotelFixedPromotion(HotelFixedPromotionVO hotelFixedPromotionVO); |
| 前置条件 | 酒店工作人员修改一条固定策略折扣，请求修改 |
| 后置条件 | 系统更新并保存当前固定策略的折扣，并返回是否成功更新 |
| promotion. getHotelSpecialSpanPromotions | 语法 | public Iterator<SpecialSpanPromotionVO> getHotelSpecialSpanPromotions(String userID); |
| 前置条件 | 已知酒店编号 |
| 后置条件 | 系统返回酒店特定期间促销策略集合 |
| promotion.getWebSpecialSpanPromotions | 语法 | public Iterator<SpecialSpanPromotionVO> getWebSpecialSpanPromotions(); |
| 前置条件 | 网站营销人员请求查看网站特定期间促销策略 |
| 后置条件 | 系统返回网站特定期间促销策略 |
| promotion. addHotelSpecialSpanPromotion | 语法 | public ResultMessage addHotelSpecialSpanPromotion(SpecialSpanPromotionVO specialSpanPromotionVO) |
| 前置条件 | 已知添加的酒店特定期间策略的信息 |
| 后置条件 | 系统添加并保存一条特定期间策略，并返回是否成功添加 |
| promotion.addWebSpecialSpanPromotion | 语法 | public ResultMessage addWebSpecialSpanPromotion(SpecialSpanPromotionVO specialSpanPromotionVO) |
| 前置条件 | 已知添加的网站特定期间策略的信息 |
| 后置条件 | 系统添加并保存一条特定期间策略，并返回是否成功添加 |
| promotion.updateHotelSpecialSpanPromotion | 语法 | public ResultMessage updateHotelSpecialSpanPromotion(SpecialSpanPromotionVO specialSpanPromotionVO) |
| 前置条件 | 已知修改的酒店特定期间策略的信息 |
| 后置条件 | 系统更新并保存该条特定期间策略的信息，返回是否成功更新 |
| promotion. updateWebSpecialSpanPromotion | 语法 | public ResultMessage updateWebSpecialSpanPromotion(SpecialSpanPromotionVO specialSpanPromotionVO) |
| 前置条件 | 已知修改的网站特定期间策略的信息 |
| 后置条件 | 系统更新并保存该条特定期间策略的信息，返回是否成功更新 |
| promotion.deleteHotelSpecialSpanPromotion | 语法 | public ResultMessage deleteHotelSpecialSpanPromotion(String userID,String promotionName) |
| 前置条件 | 已知酒店编号和被删除的特定期间策略的名称 |
| 后置条件 | 系统删除该特定期间策略，返回是否删除成功 |
| promotion.deleteWebSpecialSpanPromotion | 语法 | public ResultMessage deleteWebSpecialSpanPromotion(String promotionName) |
| 前置条件 | 已知被删除的网站特定期间策略的名称 |
| 后置条件 | 系统删除该条特定期间策略，返回是否删除成功 |
| promotion.getSpecialCirclePromotions | 语法 | public Iterator<AddressVO> getSpecialCirclePromotions(String city) |
| 前置条件 | 已知城市名称 |
| 后置条件 | 系统返回该城市的特定商圈vip促销策略集合 |
| promotion.updateSpecialCirclePromotions | 语法 | public ResultMessage updateSpecialCirclePromotions(AddressVO addressVO) |
| 前置条件 | 已知修改的特定商圈折扣信息 |
| 后置条件 | 系统更新并保存该条修改的特定商圈折扣信息，返回是否成功修改 |
| promotion. getSpecialCirclePromotion | 语法 | public double getSpecialCirclePromotion(String city, String circle) |
| 前置条件 | 已知城市和商圈的名称 |
| 后置条件 | 系统返回该城市商圈的vip折扣 |
| promotion.getDiscountInSpan | 语法 | Public Iterator<Double> getDiscountInSpan(PreOrderVO preOrder) |
| 前置条件 | 已知预创建订单的信息 |
| 后置条件 | 返回在该订单的生效期间的所有天的折扣集合 |
| 需要的接口（需接口） | | |
| 服务名 | **服务内容** | |
| promotionDataService. getHotelFixedPromotion(String hotelWorkerID) | 得到数据库中对应该酒店管理的固定策略列表 | |
| promotionDataService. updateHotelFixedPromotion(HotelFixedPromotionPO hotelFixedPromotionPO) | 更新数据库中一条酒店固定策略，返回是否更新成功 | |
| promotionDataService.getHotelSpecialSpanPromotion(String hotelID) | 得到数据库中对应该酒店管理的特定期间促销策略列表 | |
| promotionDataService. getWebSpecialSpanPromotion() | 得到数据库中对应网站的特定期间促销策略策略列表 | |
| promotionDataService. getHotelFixedPromotion(String hotelWorkerID) | 得到数据库中对应该酒店管理的固定策略列表 | |
| promotionDataService. addSpecialSpanPromotion(SpecialSpanPromotionPO specialSpanPromotionPO) | 向数据库中添加一条特定期间促销策略，返回是否添加成功 | |
| promotionDataService. updateSpecialSpanPromotion(SpecialSpanPromotionPO specialSpanPromotionPO) | 更新数据库中一条特定期间促销策略列表，返回是否更新成功 | |
| promotionDataService. deleteSpecialSpanPromotion(String userID,String promotionName) | 删除数据库中一条特定期间促销策略，返回是否删除成功 | |
| promotionDataService. getSpecialCirclePromotion(String city) | 返回对应该城市的vip特定商圈促销策略列表 | |
| promotionDataService. updateSepecialCirclePromotion(AddressPO addressPO) | 更新数据库中一条vip特定商圈促销策略，返回是否更新成功 | |
| promotionDataService. getSpecialCircleDiscount(String city, String cycle) | 返回指定城市、商圈的vip特定商圈折扣 | |
| member.isMember([String](eclipse-javadoc:%E2%98%82=ClientSystem/src%5C/main%5C/java%3CbusinessLogic.memberBL%7BMember.java%E2%98%83Member~isMember~QString;~QMemberType;%E2%98%82java.lang.String) userID, [MemberType](eclipse-javadoc:%E2%98%82=ClientSystem/src%5C/main%5C/java%3CbusinessLogic.memberBL%7BMember.java%E2%98%83Member~isMember~QString;~QMemberType;%E2%98%82utilities.enums.MemberType) memberType) | 判断该用户是否为指定会员类型 | |
| member.getMemberInfo([String](eclipse-javadoc:%E2%98%82=ClientSystem/src%5C/main%5C/java%3CbusinessLogic.memberBL%7BMember.java%E2%98%83Member~getMemberInfo~QString;%E2%98%82java.lang.String) userID) | 根据用户编号，返回该用户的普通会员信息 | |
| market.getMemberDiscout([String](eclipse-javadoc:%E2%98%82=ClientSystem/src%5C/main%5C/java%3CbusinessLogic.marketBL%7BMarket.java%E2%98%83Market~getMemberDiscout~QString;%E2%98%82java.lang.String) userID) | 根据用户编号，返回该用户的会员等级折扣 | |
| hotel.getHotelAddress([String](eclipse-javadoc:%E2%98%82=ClientSystem/src%5C/main%5C/java%3CbusinessLogic.hotelBL.hotel%7BHotel.java%E2%98%83Hotel~getHotelAddress~QString;%E2%98%82java.lang.String) hotelID) | 根据酒店编号，返回酒店的地址信息 | |

1. **credit模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| creditBL.charge | 语法 | public double charge(String guestID, double creditNum) |
| 前置条件 | 网站营销人员被识别和授权，并线下已知客户编号和欲充值的的信用值 |
| 后置条件 | 系统为客户信用值充值 |
| creditBL.getAllCreditDetail | 语法 | public Iterator<CreditVO> getAllCreditDetail(final String guestID) |
| 前置条件 | 客户已被识别和授权 |
| 后置条件 | 无 |
| creditBL.getCreditOfOneOrder | 语法 | public Iterator<CreditVO> getCreditOfOneOrder(String orderID) |
| 前置条件 | 无 |
| 后置条件 | 无 |
| creditBL.addCreditRecord | 语法 | public ResultMessage addCreditRecord(CreditVO creditVO) |
| 前置条件 | 用户已被识别和授权 |
| 后置条件 | 在信用中添加此条记录 |
| 需要的接口（需接口） | | |
| 服务名 | **服务内容** | |
| creditDataService.getAllCreditDetail(final String guestID) | 根据客户编号获得单个客户的所有信用记录的持久化对象 | |
| creditDataService.getCreditOfOneOrder(String orderID) | 获得一个订单的所有信用记录持久化对象 | |
| creditDataService.addCredit(CreditVO creditVO) | 为客户添加信用记录 | |
| userBL.modifyCredit(String guestID, double creditNum) | 修改客户的信用信息 | |
| UserBL.getSingle(String userID) | 根据客户编号查看客户持久化对象 | |

1. **member模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| Memeber.add | 语法 | Public ResultMessage add(MemberVO memberVO) |
| 前置条件 | 会员信息符合规则且齐全 |
| 后置条件 | 系统根据已指定会员类型持久化为该客户增加会员数据 |
| Member.modify | 语法 | Public ResultMessage modify(MemberVO memberVO) |
| 前置条件 | 会员信息符合规则且齐全 |
| 后置条件 | 系统修改该会员并持久化保存该客户的会员数据 |
| Member.getMemberInfo | 语法 | Public MemberVO getMemberInfo(String userID,MemberType memberType) |
| 前置条件 | userID.Length=对应用户类型的ID长度 |
| 后置条件 | 系统返回与该userID对应的指定会员类型 |
| Member.isMember | 语法 | public boolean isMember(String userID, MemberType memberType) |
| 前置条件 | userID.Length=对应用户类型的ID长度 |
| 后置条件 | 返回该userID的用户是否为会员 |
| Member.getMemberType | 语法 | public MemberType getMemberType(String userID, MemberType memberType) |
| 前置条件 | userID.Length=对应用户类型的ID长度 |
| 后置条件 | 系统返回该userID的会员类型 |
| 需要的接口（需接口） | | |
| 服务名 | **服务内容** | |
| GuestDataService.ModifyMember(MemberPO memberPO) | 根据提供的memberPO修改单一持久化客户对象的信用值 | |
| GuestDataService.  getSingle (String userID) | 根据ID查找客户持久化对象 | |

1. **user模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| User.getAll | 语法 | Public List<UserVO> getAll(UserType userType) |
| 前置条件 | 无 |
| 后置条件 | 系统返回指定用户类型的所有用户信息 |
| User.getSingle | 语法 | Public UserVO getSingle(String userID，UserType userType) |
| 前置条件 | userID.Length=对应用户类型的ID长度 |
| 后置条件 | 系统返回与该ID对应的user信息 |
| User.add | 语法 | Public ResultMessage add(UserVO newUserVO) |
| 前置条件 | 输入的userID不存在，password和所有信息符合规范 |
| 后置条件 | 系统新增该用户并持久化保存该用户的数据 |
| User.addHotel | 语法 | Public ResultMessage addHotel(HotelVO newHotelVO, String hotelID) |
| 前置条件 | 输入的hotelID不存在,所有酒店信息符合规范 |
| 后置条件 | 系统新增该酒店并持久化保存该酒店的数据 |
| User.modify | 语法 | Public ResultMessage modify(UserVO userVO) |
| 前置条件 | 用户信息符合规则且齐全 |
| 后置条件 | 系统修改该用户信息并持久化保存该用户数据 |
| User.modifyCredit | 语法 | Public ResultMessage modifyCredit (float creditNum) |
| 前置条件 | 网站营销人员为客户信用充值 |
| 后置条件 | 系统修改该客户信用值并持久化保存该用户数据 |
| User.getLoginInfo | 语法 | Public ResultMessage getLoginInfo(String userID, UserType userType) |
| 前置条件 | userID.Length=对应用户类型的ID长度 |
| 后置条件 | 系统返回与该用户ID对应的密码  如果ID不存在，则系统返回该ID对应用户不存在 |
|  | | |
| 需要的接口（需接口） | | |
| 服务名 | **服务内容** | |
| guestDataService.  add(GuestPO newGuestPO) | 根据提供的newGuestPO增加客户持久化对象 | |
| guestDataService.  getSingle (String guestID) | 根据userID查找客户持久化对象 | |
| guestDataService.modify  (GuestPO guestPO) | 修改单一客户持久化对象 | |
| guestDataService.getAll() | 获取所有客户持久化对象 | |
| hotelWorkerDataService.  add(HotelWorkerPO newHotelWorkerPO) | 根据提供的newHotelWorkerPO增加酒店工作人员持久化对象 | |
| hotelWorkerDataService.  getSingle (String hotelWorkerID) | 根据userID查找酒店工作人员持久化对象 | |
| hotelWorkerDataService.modify  (HotelPO hotelPO) | 修改单一酒店工作人员持久化对象 | |
| hotelWorkerDataService.getAll() | 获取所有酒店工作人员持久化对象 | |
| webMarketerDataService.  add(WebMarketerPO newWebMarketerPO) | 根据提供的newWebMarkterPO增加网站营销人员持久化对象 | |
| webMarkterDataService.  getSingle (String webMarketerID) | 根据userID查找网站营销人员持久化对象 | |
| webMarkterDataService.modify(WebMarketerPO webMarketerPO) | 修改单一网站营销人员持久化对象 | |
| webMarketerDataService.getAll() | 获取所有网站营销人员持久化对象 | |
| webManagerDataService.  add(WebManagerPO newWebManagerPO) | 根据提供的newWebManagerPO增加网站管理人员持久化对象 | |
| webManagerDataService.  getSingle (String webManagerID) | 根据userID查找网站管理人员持久化对象 | |
| webManagerDataService.modify  (WebManagerPO webManagerPO) | 修改单一网站管理人员持久化对象 | |
| webManagerDataService.getAll() | 获取所有网站管理人员持久化对象 | |
| Hotel.add | 新增该酒店并持久化保存该酒店的数据 | |

1. **market模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| market.getMemberFormulation | 语法 | Public MarketVO getMemberFormulation () |
| 前置条件 | 无 |
| 后置条件 | 返回现有的会员等级制度信息 |
| market.setMemberFormulation | 语法 | Public ResultMessage setMemberFormulation (MarketVO marketVO) |
| 前置条件 | 网站营销人员输入会员等级制度 |
| 后置条件 | 系统记录此会员等级制度信息 |
| 需要的接口（需接口） | | |
| 服务名 | **服务内容** | |
| marketDataService.getMemberFormulation | 返回现有的会员等级制度信息 | |
| marketDataService.setMemberFormulation(MarketVO marketVO) | 修改会员等级制度信息持久化对象，系统返回是否成功记录 | |

1. **source模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| source.getHotelFixedPromotions | 语法 | public Iterator<String> getCities() |
| 前置条件 |  |
| 后置条件 | 返回系统支持的所有城市集合 |
| source.getCircles | 语法 | public Iterator<String> getCircles(String city) |
| 前置条件 | 已知城市名称 |
| 后置条件 | 返回该城市的所有商圈集合 |
| source.getLevels | 语法 | public Iterator<String> getLevels() |
| 前置条件 |  |
| 后置条件 | 返回系统支持的所有酒店等级的集合 |
| source.getRoomTypes | 语法 | public Iterator<String> getRoomTypes() |
| 前置条件 |  |
| 后置条件 | 获取系统支持的所有酒店房型 |
| source.getMaxGuestNumEachOrder | 语法 | public int getMaxGuestNumEachOrder() |
| 前置条件 |  |
| 后置条件 | 返回系统支持的单个订单允许的最大人数 |
| source.getMaxRoomNumEachOrder | 语法 | public int getMaxRoomNumEachOrder() |
| 前置条件 |  |
| 后置条件 | 返回系统支持的单个订单允许预订的最大房间数 |
| 需要的接口（需接口） | | |
| 服务名 | **服务内容** | |
| sourceDataService. getCities() | 获取该系统覆盖的所有城市列表 | |
| sourceDataService.getCircles(String city) | 根据选中的城市，获取该城市内，系统支持的所有商圈列表 | |
| sourceDataService.getLevels() | 获取该系统支持的所有酒店的星级列表 | |
| sourceDataService.getRoomTypes() | 获取该系统支持的所有房间类型列表 | |
| sourceDataService.getMaxGuestNumEachOrder() | 获取系统允许的每一订单最大人数 | |
| sourceDataService.getMaxRoomNumEachOrder() | 获取系统允许的每一订单最大房间数 | |

## 数据层的分解

数据层主要给业务逻辑层提供数据访问服务，包括对于持久化数据的增、删、改、查。例如，AccountBusinessService的服务主要由AccountDataService提供。由于持久化数据的保存可能存在多种形式：Txt文件、序列化文件、数据库等，所示抽象了数据服务。以AccountDataService为例，数据层模块的具体描述下图所示：

### 数据层模块的职责

|  |  |
| --- | --- |
| 模块 | 职责 |
| orderData | 负责保存订单信息，进行增、改、查的操作 |
| hotelData | 负责保存酒店信息，进行增、改、查的操作 |
| promotionData | 负责保存促销策略信息，进行增、删、改、查的操作 |
| marketData | 负责保存会员等级制度信息，进行改、查的操作 |
| guestData | 负责保存客户信息，进行增、改、查的操作 |
| hotelWorkerData | 负责保存酒店工作人员信息，进行增、改、查的操作 |
| webMarketerData | 负责保存网站营销人员信息，进行改、查的操作 |
| webManagerData | 负责保存网站管理人员信息，进行改、查的操作 |

### 数据层模块的接口规范

1. **orderData模块的接口规范**

|  |  |  |
| --- | --- | --- |
| orderDataService.createOrder | 语法 | public ResultMessage createOrder (OrderPO order) |
| 前置条件 | 客户已被认证和授权 |
| 后置条件 | 系统生成未执行订单并持久化增加该客户的订单数据 |
| orderDataService.executeOrder | 语法 | public ResultMessage executeOrder (String orderID) |
| 前置条件 | 酒店工作人员已被认证和授权 |
| 后置条件 | 系统更改状态为已执行订单并持久化保存该客户的订单数据 |
| orderDataService.undoAbnormalOrder | 语法 | public ResultMessage undoAbnormalOrder (String orderID, double percent) |
| 前置条件 | 网站营销人员已被认证和授权 |
| 后置条件 | 系统更改状态为已撤销订单、恢复一定比例的信用值并持久化保存该客户的订单数据 |
| orderDataService.undoNormalOrder | 语法 | public ResultMessage undoNormalOrder (String orderID) |
| 前置条件 | 客户已被认证和授权，订单状态为未执行 |
| 后置条件 | 系统更改状态为已撤销订单并持久化保存该客户的订单数据 |
| orderDataService.getOrderDetails | 语法 | public OrderPO getOrderDetail(String orderID) |
| 前置条件 | 用户已被认证和授权 |
| 后置条件 | 无 |
| orderDataService.getAllGuestOrderGeneral | 语法 | public List<OrderGeneralPO> getAllGuestOrderGeneral (String guestID) |
| 前置条件 | 客户已被认证和授权 |
| 后置条件 | 无 |
| orderDataService.getAllHotelOrderGeneral | 语法 | public List<OrderGeneralPO> getAllHotelOrderGeneral (String orderID) |
| 前置条件 | 酒店工作人员已被认证和授权 |
| 后置条件 | 无 |
| orderDataService.getAllAbnormalOrderGeneral | 语法 | public List<OrderGeneralPO> getAllAbnormalOrderGeneral (LocalDate date) |
| 前置条件 | 网站营销人员已被认证和授权 |
| 后置条件 | 无 |
| orderDataService.getAllUnexecutedOrderGeneral | 语法 | public List<OrderGeneralPO> getAllUnexecutedOrderGeneral (LocalDate date) |
| 前置条件 | 网站营销人员已被认证和授权 |
| 后置条件 | 无 |
| orderDataService.updateCheckIn | 语法 | ResultMessage updateCheckIn(CheckInPO checkInPO) |
| 前置条件 | 网站营销人员已被认证和授权 |
| 后置条件 | 无 |
| orderDataService.updateCheckOut | 语法 | ResultMessage updateCheckOut(CheckOutPO checkOutPO) |
| 前置条件 | 网站营销人员已被认证和授权 |
| 后置条件 | 无 |
| orderDataService.addEvaluation | 语法 | ResultMessage addEvaluation(GuestEvaluationPO guestEvaluationPO) |
| 前置条件 | 网站营销人员已被认证和授权 |
| 后置条件 | 无 |
| orderDataService.getEvaluations | 语法 | List<HotelEvaluationPO> getEvaluations(String hotelID) |
| 前置条件 | 网站营销人员已被认证和授权 |
| 后置条件 | 无 |
| orderDataService.getBookedHotels | 语法 | List<String> getBookedHotels(String guestID) |
| 前置条件 | 网站营销人员已被认证和授权 |
| 后置条件 | 无 |
| 需要的接口（需接口） | | |
| 服务名 | **服务内容** | |
| dataHelper.add(OrderPO orderPO) | 新增订单的持久化对象 | |
| dataHelper.setState(String orderID, OrderState state) | 根据订单编号修改订单持久化对象的状态 | |
| dataHelper.getSingleOrder(String orderID) | 根据客户编号查找客户单个订单的持久化对象 | |
| dataHelper.getAllOrderOfGuest(String guestID) | 根据客户编号查找客户的所有订单持久化对象 | |
| dataHelper.getAllOrderOfHotel(String hotelID) | 根据酒店编号查找酒店的所有订单持久化对象 | |
| dataHelper.getUnexecuted() | 查找所有未执行订单的持久化对象 | |
| dataHelper.getAbnormal() | 查找所有异常订单的持久化对象 | |
| dataHelper.setCheckIn(String orderID, String roomNumber, LocalDateTime checkInTime, LocalDateTime expectLeaveTime) | 根据订单编号修改单个订单持久化对象的入住信息 | |
| dataHelper.setCheckOut(String orderID, LocalDateTime checkOutTime) | 根据订单编号修改单个订单持久化对象的退房信息 | |
| dataHelper.setEvaluation(String orderID, double score, String comment) | 根据订单编号修改单个订单持久化对象的评论信息 | |
| dataHelper.setHasCommentBool(String orderID) | 根据订单编号修改单个订单持久化对象的评价特征值 | |

1. **hotelData模块的接口规范**

|  |  |  |
| --- | --- | --- |
| hotelData.getHotelInfo | 语法 | public HotelPO getHotelInfo (String hotelID) |
| 前置条件 | 无 |
| 后置条件 | 获取数据库中对应该hotelID的一条酒店信息 |
| hotelDat.updateHotelInfo | 语法 | public ResultMessage updateHotelInfo(HotelPO hotelPO) |
| 前置条件 | 无 |
| 后置条件 | 更新数据库中一条酒店基本信息 |
| hotelData.addHotelInfo | 语法 | public ResultMessage addHotelInfo(HotelPO hotelPO) |
| 前置条件 | 无 |
| 后置条件 | 向数据库中添加一条酒店基本信息 |
| hotelData.getHotels | 语法 | public List<HotelPO> getHotels(String city, String circle) |
| 前置条件 | 无 |
| 后置条件 | 获取酒店数据库中所有位于对应城市、商圈的酒店集合 |
| hotelData.getRoomInfo | 语法 | public List<RoomInfoPO> getRoomInfo(String hotelID) |
| 前置条件 | 无 |
| 后置条件 | 得到数据库中对应酒店编号酒店的客房信息集合 |
| hotelData.updateRoomInfo | 语法 | public ResultMessage updateRoomInfo(RoomInfoPO roomInfoPO) |
| 前置条件 | 无 |
| 后置条件 | 更新数据库中一条客房信息 |
| hotelData.addRoomInfo | 语法 | public ResultMessage addRoomInfo(RoomInfoPO roomInfoPO) |
| 前置条件 | 无 |
| 后置条件 | 向数据库中添加一条客房信息 |
| hotelData.deleteRoomInfo | 语法 | public ResultMessage deleteRoomInfo(String hotelID, RoomType roomType) |
| 前置条件 | 无 |
| 后置条件 | 删除一条酒店客房信息 |
| hotelData.getRoomInfo | 语法 | public List<RoomInfoPO> getRoomInfo(String hotelID) |
| 前置条件 | 无 |
| 后置条件 | 得到数据库中对应酒店编号酒店的客房信息集合 |
| hotelData.getHotels | 语法 | public List<HotelPO> getHotels(String city, String circle) |
| 前置条件 | 无 |
| 后置条件 | 获取酒店数据库中所有位于对应城市、商圈的酒店集合 |
| hotelData.getRoomInfo | 语法 | public List<RoomInfoPO> getRoomInfo(String hotelID) |
| 前置条件 | 无 |
| 后置条件 | 得到数据库中对应酒店编号酒店的客房信息集合 |

1. **promotionData模块的接口规范**

|  |  |  |
| --- | --- | --- |
| promotionData.getHotelFixedPromotion | 语法 | public List<HotelFixedPromotionPO> getHotelFixedPromotion(String hotelWorkerID) |
| 前置条件 | 无 |
| 后置条件 | 获取数据库中对应酒店编号的固定策略集合 |
| promotionDat.updateHotelFixedPromotion | 语法 | public ResultMessage updateHotelFixedPromotion(HotelFixedPromotionPO hotelFixedPromotionPO) |
| 前置条件 | 无 |
| 后置条件 | 更新数据库中一条酒店基本信息 |
| promotionData.getHotelSpecialSpanPromotion | 语法 | public List<SpecialSpanPromotionPO> getHotelSpecialSpanPromotion(String hotelID) |
| 前置条件 | 无 |
| 后置条件 | 获取数据库中对应酒店编号的特定期间策略集合 |
| promotionData.getWebSpecialSpanPromotion | 语法 | public List<SpecialSpanPromotionPO> getWebSpecialSpanPromotion() |
| 前置条件 | 无 |
| 后置条件 | 获取数据库中网站特定期间策略的集合 |
| promotionData.addSpecialSpanPromotion | 语法 | public ResultMessage addSpecialSpanPromotion(SpecialSpanPromotionPO specialSpanPromotionPO) |
| 前置条件 | 无 |
| 后置条件 | 向数据库中添加一条特定期间策略 |
| promotionData.updateSpecialSpanPromotion | 语法 | public ResultMessage updateSpecialSpanPromotion(SpecialSpanPromotionPO specialSpanPromotionPO) |
| 前置条件 | 无 |
| 后置条件 | 更新数据库中一条特定期间策略 |
| promotionData.deleteSpecialSpanPromotion | 语法 | public ResultMessage deleteSpecialSpanPromotion(String userID,String promotionName) |
| 前置条件 | 无 |
| 后置条件 | 删除数据库中一条对应酒店的对应特定期间策略 |
| promotionData.getSpecialCirclePromotion | 语法 | public List<AddressPO> getSpecialCirclePromotion(String city) |
| 前置条件 | 无 |
| 后置条件 | 获取数据库中对应城市的vip特定商圈策略集合 |
| promotionData.updateSepecialCirclePromotion | 语法 | public ResultMessage updateSepecialCirclePromotion(AddressPO addressPO) |
| 前置条件 | 无 |
| 后置条件 | 更新数据库中一条vip特定商圈策略 |
| promotionData.getSpecialCircleDiscount | 语法 | public double getSpecialCircleDiscount(String city, String cycle) |
| 前置条件 | 无 |
| 后置条件 | 获取特定对应商圈的折扣 |

1. **marketData模块的接口规范**

|  |  |  |
| --- | --- | --- |
| marketData.getMemberFormulation | 语法 | Public MarketPO getMemberFormulation() |
| 前置条件 | 无 |
| 后置条件 | 返回现有的会员等级制度信息 |
| marketData.setMemberFormulation | 语法 | Public MessageResult setMemberFormulation (HotelInfoPO hotelInfoPO) |
| 前置条件 | 网站营销人员输入会员等级制度信息 |
| 后置条件 | 系统记录此会员等级制度信息 |

1. **guestData模块的接口规范**

|  |  |  |
| --- | --- | --- |
| guestData.modifyMember | 语法 | Public ResultMessage modifyMember(MemberPO memberPO) |
| 前置条件 | 无 |
| 后置条件 | 根据提供的memberPO修改单一持久化客户对象的信用值 |
| guestData.getSingleGuest | 语法 | Public ResultMessage getSingle Guest(String guestID) |
| 前置条件 | guestID.Length=客户的ID长度10 |
| 后置条件 | 系统返回与该ID对应的客户信息 |
| guestData.add | 语法 | Public ResultMessage add(GuestPO newGuestPO) |
| 前置条件 | 输入的guestID不存在，password和所有客户信息符合规范 |
| 后置条件 | 系统新增该客户并持久化保存该客户的数据 |
| guestData.modify | 语法 | Public ResultMessage modify  (GuestPO guestPO) |
| 前置条件 | 客户信息符合规则且齐全 |
| 后置条件 | 系统修改该用户信息并持久化保存该用户数据 |
| guestData.getAllGuest | 语法 | Public List<GuestPO> getAllGuest() |
| 前置条件 | 无 |
| 后置条件 | 系统返回所有客户信息 |

1. **hotelWorker模块的接口规范**

|  |  |  |
| --- | --- | --- |
| hotelWorkerData.getSingleHotelWorker | 语法 | Public ResultMessage getSingleHotelWorker(String hotelWorkerID) |
| 前置条件 | hotelWorkerID.Length=酒店工作人员的ID长度8 |
| 后置条件 | 系统返回与该ID对应的酒店工作人员信息 |
| hotelWorkerData.add | 语法 | Public ResultMessage add(HotelWorkerPO newHotelWorkerPO) |
| 前置条件 | 输入的hotelWorkerID不存在，password和所有酒店工作人员信息符合规范 |
| 后置条件 | 系统新增该酒店工作人员并持久化保存该酒店工作人员的数据 |
| hotelWorkerData.modify | 语法 | Public ResultMessage modify  (HotelWorkerPO hotelWorkerPO) |
| 前置条件 | 酒店工作人员信息符合规则且齐全 |
| 后置条件 | 系统修改该酒店工作人员信息并持久化保存该酒店工作人员数据 |
| hotelWorkerData.getAllHotelWorker | 语法 | Public List<HotelWorkerPO> getAllHotelWorker() |
| 前置条件 | 无 |
| 后置条件 | 返回所有酒店工作人员的信息 |

1. **webMarketer模块的接口规范**

|  |  |  |
| --- | --- | --- |
| webMarketerData.getSingleWebMarketer | 语法 | Public ResultMessage getSingleWebMarketer(String webMarketID) |
| 前置条件 | webMarketID.Length=网站营销人员的ID长度6 |
| 后置条件 | 系统返回与该ID对应的网站营销人员信息 |
| webMarketerData.add | 语法 | Public ResultMessage add(WebMarketerPO newWebMarketerPO) |
| 前置条件 | 输入的webMarketerID不存在，password和所有网站营销人员信息符合规范 |
| 后置条件 | 系统新增该网站营销人员并持久化保存该网站营销人员的数据 |
| webMarketerData.modify | 语法 | Public ResultMessage modify  (WebMarketerPO webMarketerPO) |
| 前置条件 | 网站营销人员信息符合规则且齐全 |
| 后置条件 | 系统修改该网站营销人员信息并持久化保存该网站营销人员数据 |
| webMarketerData.getAllWebMareketer | 语法 | Public List<WebMarketerPO> getAllWebMarketer() |
| 前置条件 | 无 |
| 后置条件 | 返回所有网站营销人员信息 |

1. **webManager模块的接口规范**

|  |  |  |
| --- | --- | --- |
| webManagerData.getSingleWebManager | 语法 | Public ResultMessage getSingleWebManager (String webManagerID) |
| 前置条件 | webManagerID.Length=网站管理人员的ID长度4 |
| 后置条件 | 系统返回与该ID对应的网站管理人员信息 |
| webManagerData.add | 语法 | Public ResultMessage add(WebManagerPO newWebManagerPO) |
| 前置条件 | 输入的webManagerID不存在，password和所有网站管理人员信息符合规范 |
| 后置条件 | 系统新增该网站管理人员并持久化保存该网站管理人员的数据 |
| webManagerData.modify | 语法 | Public ResultMessage modify  (WebManagerPO webManagerPO) |
| 前置条件 | 网站管理人员信息符合规则且齐全 |
| 后置条件 | 系统修改该网站管理人员信息并持久化保存该网站管理人员数据 |
| webManagerData.getAllWebManager | 语法 | Public List<WebManagerPO> getAllWebManager() |
| 前置条件 | 无 |
| 后置条件 | 返回所有网站管理人员信息 |

1. **creditData模块的接口规范**

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| creditDataService.getAllCreditDetail | 语法 | public List<CreditPO> getAllCreditDetail(String guestID) |
| 前置条件 | 客户已被认证和授权 |
| 后置条件 | 无 |
| creditDataService.addCredit | 语法 | public ResultMessage addCredit(CreditPO creditPO) |
| 前置条件 | 无 |
| 后置条件 | 添加一条信用记录 |
| 需要的接口（需接口） | | |
| 服务名 | **服务内容** | |
| dataHelper.getAllCreditDetail(String guestID) | 获取该客户的所有订单记录 | |
| dataHelper.addCredit(CreditPO creditPO) | 新增信用记录 | |

# 信息视角

## 数据持久化对象

系统的PO类就是对应的相关的实体类，如下所示。

|  |  |
| --- | --- |
| 类名 | 包含的属性 |
| MemberPO | 客户编号、生日（普通会员）、企业名称（企业会员） |
| GuestPO | 客户编号、生日（普通会员）、企业名称（企业会员）、姓名、昵称、密码、联系方式，信用值 |
| CreditPO | 客户编号、信用记录（时间、订单号、动作、信用度变化、信用度结果） |
| HotelWorkerPO | 酒店／酒店工作人员编号、密码、所属酒店 |
| WebMarketPO | 网站营销人员编号、密码 |
| WebManagerPO | 网站管理人员编号、密码 |
|  |  |
| OrderGeneralPO | 订单编号、客户编号、酒店编号、酒店名、地址、价格、入住日期和退房日期、订单状态 |
| OrderPO | 订单编号、客户编号、酒店编号、酒店名、地址、价格、生成时间、入住时间和退房时间、最晚订单执行时间（注：客户填写）、订单状态、房间类型、房间数、预计入住人数、入住人姓名、联系方式、特别要求（如有无儿童）、房间号、预计离开时间（注：酒店工作人员填写）、实际离开时间（注：酒店工作人员填写） |
| CheckInPO | 订单编号、房间号，入住时间、预计离开时间 |
| CheckOutPO | 订单编号、实际离开时间 |
|  |  |
| HotelGeneralPO | 酒店编号、酒店名称、所属城市商圈、星级、评分、最低房价 |
| HotelPO | 酒店编号、酒店编号、酒店名称、所属城市商圈、详细地址、星级、评分评价、简介、设施服务 |
| RoomInfoPO | 酒店编号、类型、数量、原始价格 |
| RemainRoomInfoPO | 酒店编号、类型、数量、原始价格 |
|  |  |
| MarketPO | 等级名称、等级信用度、等级优惠 |
|  |  |
| PromotionPO | 促销策略、折扣率、折扣酒店、开始日期和结束日期 |

## 文件格式

本系统采用数据库保存文件。

|  |  |  |
| --- | --- | --- |
| 所属包与表名 | 内容 | 字段设计 |
| address | 酒店城市商圈及其折扣 |  |
| credit | 信用记录 |  |
| guest | 客户信息 |  |
| hotel | 酒店信息 |  |
| hotelfixedpromotion | 酒店固定折扣（会员生日折扣、企业会员、三间及以上预订折扣） |  |
| hotelworker | 酒店工作人员信息 |  |
| market | 会员等级信息 |  |
| order | 订单信息 |  |
| roominfo | 酒店房间信息 |  |
| specialspanpromotion | 特定区间折扣（酒店／网站） |  |
| webmanager | 网站管理人员信息 |  |
| webmarketer | 网站营销信息 |  |