**酒店预订管理系统EasyHotel**

**体系结构设计模型**

方雅婧、雷雨佳

2016-10-15

1. **体系结构模型**
   1. **逻辑视角**

酒店预订管理系统是典型的信息系统，MVC和分层是常见的两种选择，又因为开发人员不熟悉Java Swing以及RMI技术，对Web技术较了解，因此选择MVC作为系统的体系结构风格。MVC风格将系统分为3部分：模型Model、视图View、控制器Controller。模型封装应用数据，处理业务逻辑、视图负责呈现给用户的界面、控制器接受用户动作，并对应作出适当的处理。

Web MVC风格体系结构的逻辑视角和逻辑设计方案图1,2所示：

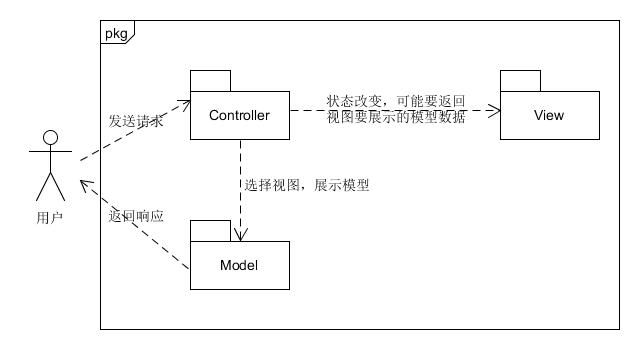


图 1参照体系结构风格的逻辑视角

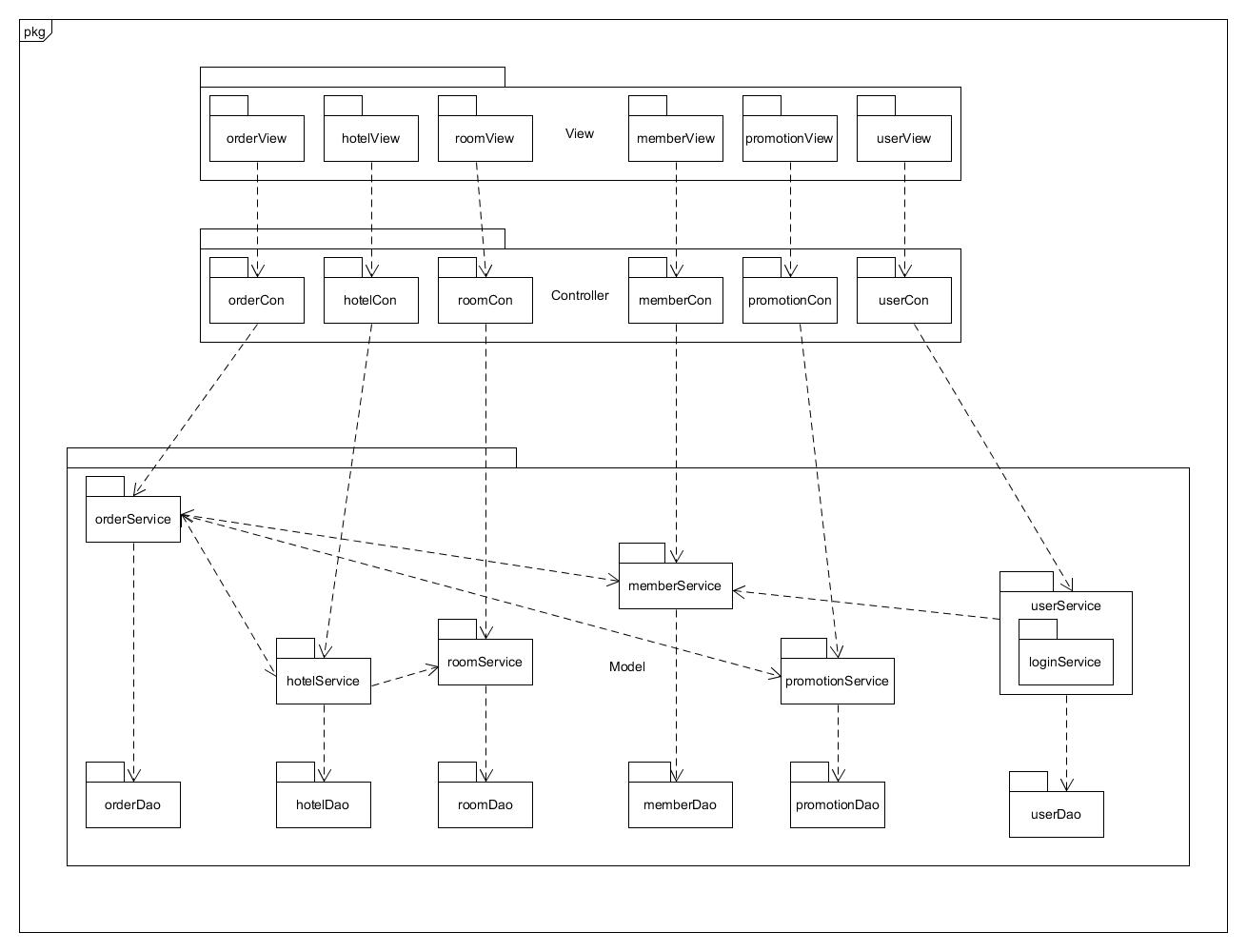


图2软件体系结构逻辑设计方案

* 1. **组合视角**
     1. **开发包图**

酒店预定管理系统的最终开发包图设计如下表所示：

表1 最终开发包设计

|  |  |
| --- | --- |
| 开发（物理）包 | 依赖的其他开发包 |
| mainView | orderView,hotelView,roomView,memberView,promotionView,userView,VO |
| orderView | dispatcherServlet,模板引擎类库包,VO |
| orderCon | orderService,VO |
| orderService |  |
| orderServiceImpl | orderService,memberServiceImpl,hotelServiceImpl,promotionServiceImpl,orderDao,VO |
| orderDao |  |
| orderDaoImpl | orderDao,databaseUtility,PO |
| hotelView | dispatcherServlet,模板引擎类库包,VO |
| hotelCon | hotelService,VO |
| hotelService |  |
| hotelServiceImpl | hotelService,roomServiceImpl,roomServiceImpl,hotelDao,VO |
| hotelDao |  |
| hotelDaoImpl | hotelDao, databaseUtility,PO |
| roomView | dispatcherServlet,模板引擎类库包,VO |
| roomCon | roomService.VO |
| roomService |  |
| roomServiceImpl | roomService,roomDao,VO |
| roomDao |  |
| roomDaoImpl | roomDao, databaseUtility,PO |
| memberView | dispatcherServlet,模板引擎类库包,VO |
| memberCon | memberService,VO |
| memberService |  |
| memberServiceImpl | memberService,memberDao,VO |
| memberDao |  |
| memberDaoImpl | memberDao, databaseUtility,PO |
| promotionView | dispatcherServlet,模板引擎类库包,VO |
| promotionCon | promotonService,VO |
| promotionService |  |
| promotionServiceImpl | promotionService,promotionDao,VO |
| promotionDao |  |
| promotionDaoImpl | promotionDao, databaseUtility,PO |
| userView | dispatcherServlet,模板引擎类库包,VO |
| userCon | userService,VO |
| userService |  |
| userServiceImpl | userService,memberServiceImpl,userDao,VO |
| userDao |  |
| userDaoImpl | userDao, databaseUtility,PO |
| VO |  |
| PO |  |
| databaseUtility | JDBC |
| 模板引擎类库包 |  |
| spring MVC框架 |  |

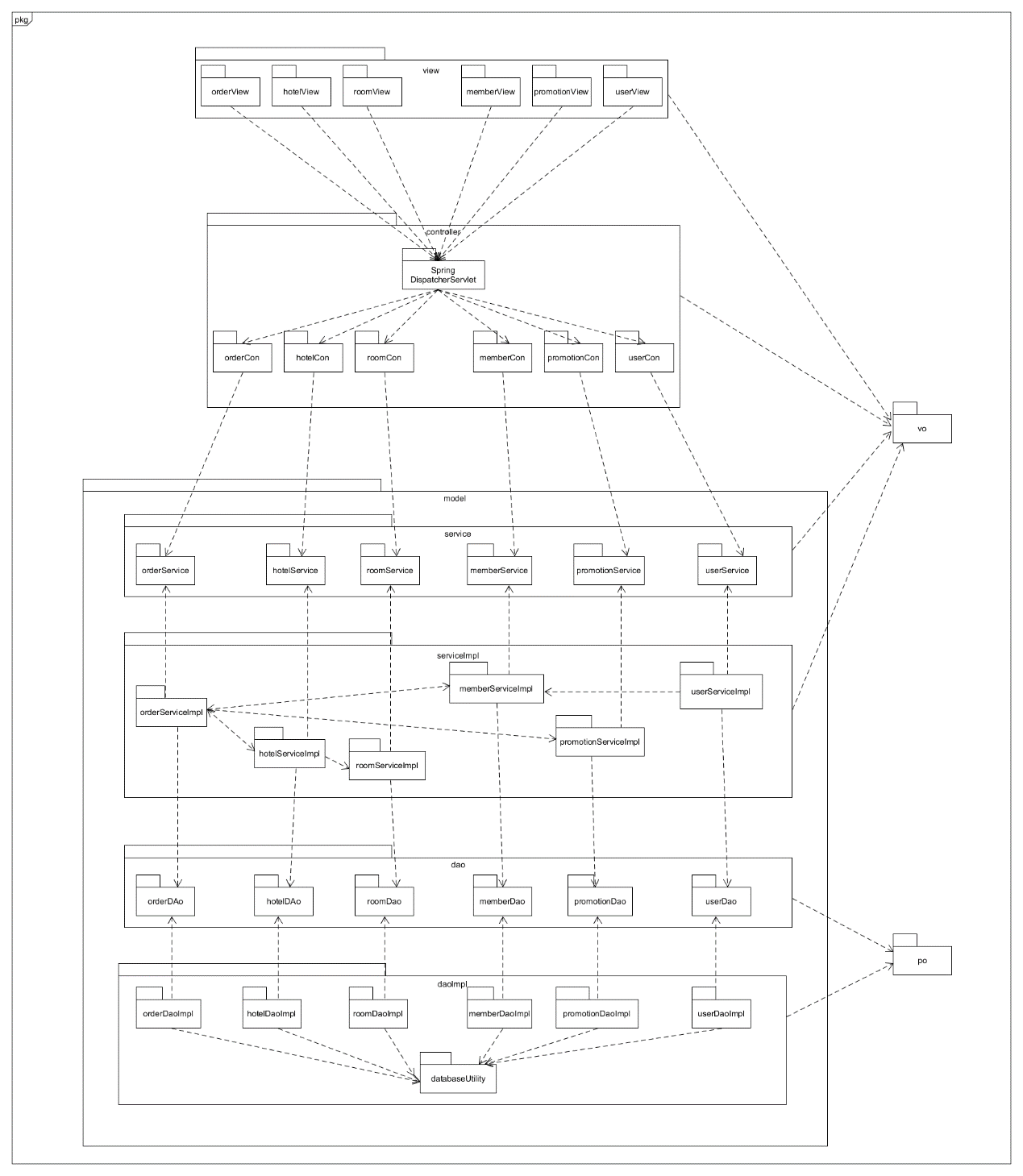


图3 开发包图

* + 1. **运行时进程**

在酒店预订管理系统中，会有多个服务端进程和一个客户端进程，其进程图如下图所示。客户端进程是在客户端上运行，服务器端进程在服务器端机器上运行。



图4 进程图

* + 1. **物理部署**

酒店预定管理系统采用Web技术开发，只需进行服务器端构件的部署。部署图如下图所示：

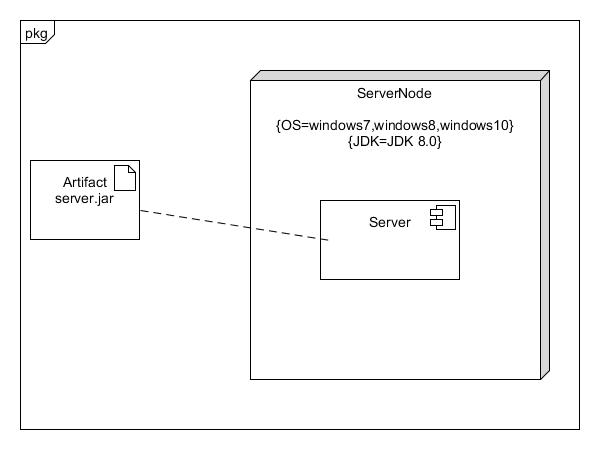


图5 部署图

* 1. **模块分解（接口视角）**

酒店预订管理系统的模块视图如下图所示：

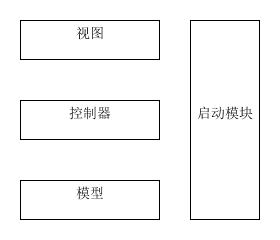
****

图6 模块视图

* + 1. **各模块的职责**

各模块的职责如下表所示：

表2 各模块的职责

|  |  |
| --- | --- |
| 模块 | 职责 |
| 启动模块 | 初始化网络通信机制，启动服务器进程 |
| 视图 | 基于Web浏览器的酒店预订管理系统用户界面 |
| 控制器 | 对用户请求进行响应，调用业务逻辑进行处理，返回对应的视图 |
| 模型 | 封装应用数据，处理业务逻辑 |

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。层之间的调用接口如下表所示

表3 层之间的调用接口

|  |  |  |
| --- | --- | --- |
| orderService | 服务调用方 | 服务提供方 |
| hotelService | 控制器层 | 模型层 |
| roomService |
| memberService |
| promotionService |
| userService |
| orderDao | 模型（业务逻辑）层 | 模型（数据访问）层 |
| hotelDao |
| roomDao |
| memberDao |
| promotionDao |
| userDao |

* + 1. **View视图层的分解**

1. 视图层的职责

View视图层主要由HTML文档以及一些资源文件（如css,js文件，图片等），负责用户界面的显示。

1. 视图层的设计原理

视图层利用CSS+Div技术对页面进行布局，使用JavaScript制作页面效果，采用模板引擎技术（jsp或thymeleaf）动态的展示后台数据

* + 1. **Control控制器层的分解**

1. 控制器层的职责

表4 控制器层的职责

|  |  |
| --- | --- |
| 模块 | 职责 |
| orderCon | 负责执行对订单相关的请求的控制 |
| hotelCon | 负责执行对酒店相关的请求的控制 |
| roomCon | 负责执行对客房相关的请求的控制 |
| memberCon | 负责执行对会员相关的请求的控制 |
| promotionCon | 负责执行对促销相关的请求的控制 |
| userCon | 负责执行对用户相关的请求的控制 |

* + 1. **Model模型层的分解**

模型层可以细分为service业务逻辑层和dao数据访问层

1. 模型层的职责

表5 模型层的职责

|  |  |
| --- | --- |
| 模块 | 职责 |
| service业务逻辑层 | |
| orderService | 负责实现与订单信息管理相关的业务逻辑 |
| hotelService | 负责实现与酒店信息管理相关的业务逻辑 |
| roomService | 负责实现与客房信息管理相关的业务逻辑 |
| memberService | 负责实现与会员信息相关的业务逻辑 |
| promotionService | 负责实现与促销策略管理相关的业务逻辑 |
| userService | 负责实现与用户管理及登录相关的业务逻辑 |
| dao数据访问层 | |
| orderDao | 负责实现数据库中订单信息的增删改查等操作 |
| hotelDao | 负责实现数据库中酒店信息的增删改查等操作 |
| roomDao | 负责实现数据库中客房信息的增删改查等操作 |
| memberDao | 负责实现数据库中会员信息的增删改查等操作 |
| promotionDao | 负责实现数据库中促销策略信息的增删改查等操作 |
| userDao | 负责实现数据库中用户信息的增删改查等操作 |

1. 模型层的接口规范

orderService包的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| orderService.getAllOrdersByCustomer | 语法 | public List<OrderVO> getAllOrdersByCustomer (String customer Id) |
| 前置条件 | 客户ID正确且符合规范 |
| 后置条件 | 根据客户id查找相应的用户，返回其所有订单 |
| orderService.getAllOrdersByHotel | 语法 | public List<OrderVO> getAllOrdersByHotel(String hotelId) |
| 前置条件 | 酒店ID正确且符合规范 |
| 后置条件 | 根据酒店id查账相应的酒店，返回其所有订单 |
| orderService.getAllCustomerOrdersByState | 语法 | public List<OrderVO> getAllCustomerOrdersByState (String customerId,String state) |
| 前置条件 | 客户ID及订单状态正确且符合规范 |
| 后置条件 | 根据客户的id查找用户，返回其指定状态的订单 |
| orderService.getAllHotelOrdersByState | 语法 | public List<OrderVO> getAllHotelOrdersByState (String hotelId,String state) |
| 前置条件 | 酒店ID及订单状态正确且符合规范 |
| 后置条件 | 根据酒店的id查找酒店，返回其指定状态的订单 |
| orderService.getOrder | 语法 | public OrderVO getOrder(String orderId) |
| 前置条件 | 订单ID正确且符合规范 |
| 后置条件 | 根据订单id找到订单并将订单信息返回 |
| orderService.placeOrder | 语法 | public OpMessage placeOrder(OrderPO order,String memberId,String hotelId,String promotionId) |
| 前置条件 | 会员ID、酒店ID、促销ID正确且符合规范 |
| 后置条件 | 根据用户输入的内容生成订单，并将生成结果返回 |
| orderService.reVOkeOrder | 语法 | public OpMessage reVOkeOrder(String OrderId) |
| 前置条件 | 订单ID正确且符合规范 |
| 后置条件 | 根据订单id将此订单撤销，并返回撤销结果 |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| memberService.getMember | 根据会员id返回该会员 | |
| hotelService.getHotel | 根据酒店id返回该酒店 | |
| promotionService.getPromotion | 根据促销信息id返回该促销信息 | |
| orderDao.insert | 在数据库中增加该订单信息 | |
| orderDao.update | 在数据库中更新该订单信息 | |
| orderDao.delete | 在数据库中删除该订单信息 | |
| orderDao.getAllOrdersByCustomer | 根据会员id返回该会员所有订单 | |
| orderDao.getAllOrdersByHotel | 根据酒店id返回该酒店所有订单 | |
| orderDao.getAllCustomerOrdersByState | 根据会员id和订单状态返回该会员相关订单 | |
| orderDao.getAllHotelOrdersByState | 根据酒店id和订单状态返回该酒店相关订单 | |
| orderDao.getOrder | 根据订单id返回该订单信息 | |

hotelService包的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| hotelService.getHotel | 语法 | | public HotelVO getHotel(String hotelId) |
| 前置条件 | | 酒店ID正确且符合规则 |
| 后置条件 | | 根据酒店ID查找酒店，返回其详细信息 |
| hotelService.getAllHotels | 语法 | | public List<HotelVO>getAllHotels() |
| 前置条件 | | 点击查询所有酒店 |
| 后置条件 | | 返回包含所有酒店的列表 |
| hotelService.searchHotels | 语法 | | public List<HotelVO> searchHotels(SearchCondition con) |
| 前置条件 | | 搜索条件输入且选择正确 |
| 后置条件 | | 根据条件查询酒店，并返回包含符合条件的酒店列表 |
| hotelService.sortHotels | 语法 | | public List<HotelVO> sortHotels(List<HotelVO> hotels ,SortCondition con) |
| 前置条件 | | 选择排序关键字 |
| 后置条件 | | 根据排序关键字对酒店列表中的酒店排序，返回排序后的酒店列表 |
| hotelService.addHotel | 语法 | | public opMessage addHotel(HotelPO hotel) |
| 前置条件 | | 酒店对象正确 |
| 后置条件 | | 根据输入的酒店信息增加酒店，返回添加结果 |
| hotelService.deleteHotel | 语法 | | public opMessage deleteHotel(String hotelId) |
| 前置条件 | | 酒店id正确且符合格式 |
| 后置条件 | | 根据酒店id删除酒店，返回删除结果 |
| hotelService.ModifyHotel | 语法 | | public opMessage modifyHotel(HotelPO hotel) |
| 前置条件 | | 酒店对象正确 |
| 后置条件 | | 修改该酒店的信息，返回修改结果 |
| 需要的服务（需接口） | | | |
| 服务名 | | 服务 | |
| roomService.getAllRoomsByHotel | | 根据酒店id返回酒店房间列表 | |
| orderService.getAllHotelOrders | | 根据酒店id返回所有酒店订单 | |
| orderService.getAllHotelOrdersByState | | 根据酒店id及订单状态返回酒店相关订单 | |
| hotelDao.insert | | 在数据库中插入新的酒店信息 | |
| hotelDao.update | | 在数据库中更新指定酒店的酒店信息 | |
| hotelDao.delete | | 根据酒店id在数据库中删除该酒店信息 | |
| hotelDao.getHotelById | | 根据酒店id查找酒店 | |
| hotelDao.getAllHotels | | 查找所有酒店 | |
| hotelDao.query | | 根据查询条件查找酒店 | |

roomService包的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| roomService.getRoom | 语法 | public RoomVO getRoom(String roomId) |
| 前置条件 | 房间id正确且符合标准 |
| 后置条件 | 根据房间id查找房间，并将该房间信息返回 |
| roomService.get getAllRoomsByHotel | 语法 | public List<RoomVO> getAllRoomsByHotel (String HotelId) |
| 前置条件 | 酒店id正确且符合标准 |
| 后置条件 | 根据酒店id查找所有房间，并返回包含所有房间的列表 |
| roomService.deleteRoom | 语法 | public opMessage deleteRoom(String roomId) |
| 前置条件 | 房间id正确且符合标准 |
| 后置条件 | 根据房间id删除房间，并返回删除结果 |
| roomService.modifyRoom | 语法 | public opMessage modifyRoom(RoomPO room) |
| 前置条件 | 房间对象正确 |
| 后置条件 | 修改指定房间的信息 |
| roomService.addRoom | 语法 | public opMessage addRoom(RoomPO room) |
| 前置条件 | 房间对象正确 |
| 后置条件 | 根据信息增加房间信息，返回添加结果 |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| roomDao.insert | 在数据库中插入一条房间信息 | |
| roomDao.update | 在数据库中修改一条房间信息 | |
| roomDao.delete | 在数据库中删除一条房间信息 | |
| roomDao.getRoomlById | 根据房间id在数据库中查找该房间 | |
| roomDao.getAllRoomsByHotel | 根据酒店id在数据库中查找该酒店所有的房间 | |

memberService包的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| memberService.searchMembers | 语法 | public List<MemberVO> searchMember(SearchCondition con) |
| 前置条件 | 搜索条件正确 |
| 后置条件 | 根据搜索条件查找用户，返回包括相关用户的列表 |
| memberService.getMemberByName | 语法 | public List<MemberVO> searchMember(String key) |
| 前置条件 | 姓名正确且符合格式 |
| 后置条件 | 根据姓名查找用户，返回用户信息 |
| memberService.getMemberById | 语法 | public MemberVO getMember(String memberId) |
| 前置条件 | 会员id正确且符合标准 |
| 后置条件 | 根据会员id查找用户，返回用户信息 |
| memberService.getMembers | 语法 | public List<MemberVO> getMembers() |
| 前置条件 | 点击查询所有用户 |
| 后置条件 | 查找系统中所有用户，返回包括所有用户的列表 |
| memberService.addMember | 语法 | public opMessage addMember(MemberPO member) |
| 前置条件 | 用户对象正确 |
| 后置条件 | 添加用户信息，返回添加结果 |
| memberService.modifyMember | 语法 | public opMessage modifyMember(MemberPO member) |
| 前置条件 | 用户对象正确 |
| 后置条件 | 修改指定用户信息，返回修改结果 |
| memberService.deleteMember | 语法 | public opMessage deleteMember(String memberId) |
| 前置条件 | 用户id正确且符合标准 |
| 后置条件 | 根据用户id删除该用户信息，返回删除结果 |
| 需要的服务（需接口） | | |
| 服务名 | 服务 | |
| orderService.getAllOrdersByCustomer | 根据用户id返回该用户所有订单 | |
| memberDao.insert | 在数据库中插入一条用户信息 | |
| memberDao.update | 在数据库中更新一条用户信息 | |
| memberDao.delete | 在数据库中删除一条用户信息 | |
| memberDao.getAllMembers | 在数据库中查找所有用户 | |
| memberDao.query | 根据搜索条件在数据可中查找相关用户 | |
| memberDao.getMemberByName | 根据姓名查找用户 | |
| memberDao.getMemberById | 根据用户id查找用户 | |

promotionService包的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| promotionService.getPromotion | | 语法 | public promotionVO getPromotion (String promotionId) |
| 前置条件 | 促销信息id正确且符合标准 |
| 后置条件 | 根据促销信息id查找促销信息并返回 |
| promotionService.getAllPromotionsByHotel | | 语法 | public List<PromotionVO> getHotelPromotions(String hotelId) |
| 前置条件 | 酒店id正确且符合格式 |
| 后置条件 | 根据酒店id查找所有促销信息，并返回包含所有促销信息的列表 |
| promotionService.getAllSitePromotions | | 语法 | public List<PromotionVO> getSitePromotions() |
| 前置条件 | 点击搜索网站促销信息 |
| 后置条件 | 返回包含所有网站促销信息的列表 |
| promotionService.addPromotion | | 语法 | public opMessage addPromotion (PromotionPO promotion) |
| 前置条件 | 促销信息对象正确 |
| 后置条件 | 添加促销，并返回添加结果 |
| promotionService.modifyPromotion | | 语法 | public opMessage modifyPromotion (PromotionPO promotion) |
| 前置条件 | 促销信息对象正确 |
| 后置条件 | 添加促销，并返回添加结果 |
| promotionService.deletePromotion | | 语法 | public opMessage deletePromotion (String promotionId) |
| 前置条件 | 促销信息id正确且符合标准 |
| 后置条件 | 根据促销信息id删除该促销，返回删除结果 |
| 需要的服务（需接口） | | | |
| 服务名 | 服务 | | |
| promotionDao.getPromotionById | 根据促销信息id查找促销信息 | | |
| promotionDao.insert | 在数据库中增加一条促销信息 | | |
| promotionDao.update | 在数据库中修改一条促销信息 | | |
| promotionDao.delete | 在数据库中删除一条促销信息 | | |
| promotionDao.getAllSitePromotions | 返回所有网站促销信息 | | |
| promotionDao.getAllPromotionsByHotel | 根据酒店id查找所有促销信息 | | |

userService包的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| userService.searchUsers | | 语法 | public List<UserVO> searchUsers(SearchCondition con) |
| 前置条件 | 搜索条件正确 |
| 后置条件 | 根据搜索条件查找相关用户，返回包括相关用户的列表 |
| userService.getUser | | 语法 | public UserVO getUser (String userId) |
| 前置条件 | 用户id正确且符合格式 |
| 后置条件 | 根据用户id查找用户，并返回其信息 |
| userrService.getAllUsers | | 语法 | public List<UserVO> getUsers() |
| 前置条件 | 点击查找所有用户 |
| 后置条件 | 查找系统内所有用户，返回包含所有用户的列表 |
| userService.addUser | | 语法 | public opMessage addUser (UserPO user) |
| 前置条件 | 用户对象正确 |
| 后置条件 | 添加一条用户信息，返回添加结果 |
| userService.modifyUser | | 语法 | public opMessage modifyUser (UserPO user) |
| 前置条件 | 用户对象正确 |
| 后置条件 | 修改指定用户信息，返回修改结果 |
| userService.deleteUser | | 语法 | public opMessage deleteUser (String userId) |
| 前置条件 | 用户id正确且符合格式 |
| 后置条件 | 根据用户id删除该用户，返回删除结果 |
| userService.memberLogin | | 语法 | public opMessage memberLogin(MemberVO member) |
| 前置条件 | 会员对象正确 |
| 后置条件 | 返回登录结果 |
| userService.userLogin | | 语法 | public opMessage userLogin(UserVO user) |
| 前置条件 | 用户对象正确 |
| 后置条件 | 返回登录结果 |
| userService.memberRegister | | 语法 | public opMessage memberRegister(MemberVO member) |
| 前置条件 | 会员对象正确 |
| 后置条件 | 返回注册结果 |
| 需要的服务（需接口） | | | |
| 服务名 | 服务 | | |
| memberService.getMember | 根据会员id查找会员信息 | | |
| userDao.insert | 在数据库中添加一条用户信息 | | |
| userDao.delete | 在数据库中删除一条用户信息 | | |
| userDao.update | 在数据库中更新一条用户信息 | | |
| userDao.getUser | 根据用户id查找用户信息 | | |
| userDao.getAllUsers | 查找数据库中所有用户 | | |
| userDao.query | 根据搜索条件查找相关用户 | | |

orderDao包的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的服务（供接口） | | | |
| orderDao.insert | | 语法 | public opMessage insert(OrderPO order) |
| 前置条件 | 同样的订单记录在数据库中不存在 |
| 后置条件 | 在数据库中插入一条订单记录 |
| orderDao.delete | | 语法 | public opMessage delete(String OrderId) |
| 前置条件 | 该id指向的订单记录在数据库要存在 |
| 后置条件 | 删除该id指向的订单记录 |
| orderDao.update | | 语法 | public opMessage update(OrderPO order) |
| 前置条件 | 同样的订单记录在数据库中要存在 |
| 后置条件 | 更新这条订单记录 |
| orderDao.getOrderById | | 语法 | public OrderPO getOrderById(String orderId) |
| 前置条件 | 该id指向的订单记录在数据库要存在 |
| 后置条件 | 返回该id指向的订单记录的信息 |
| orderDao.getAllOrdersByCustomer | 语法 | | public getAllOrdersbyCustomer(String customerId) |
| 前置条件 | | 该Id指向的客户记录在数据库中要存在 |
| 后置条件 | | 根据客户id查找他所有的订单记录并返回 |
| orderDao.getAllOrdersByHotel | 语法 | | public getAllOrdersByHotel(String hotelId) |
| 前置条件 | | 该id指向的酒店记录在数据库中要存在 |
| 后置条件 | | 根据酒店id查找其所有的订单记录并返回 |
| orderDao.getAllCustomerOrdersByState | 语法 | | public getAllCustomerOrdersByState(String state,String customrId) |
| 前置条件 | | 状态字符串要正确且该Id指向的客户记录在数据库中要存在 |
| 后置条件 | | 返回该客户所有指定状态的订单记录 |
| orderDao.getAllHotelOrdersByState | | 语法 | public getAllHotelOrdersByState(String state,String hotelId) |
| 前置条件 | 状态字符串要正确且该Id指向的酒店记录在数据库中要存在 |
| 后置条件 | 返回该酒店所有指定状态的订单记录 |

hotelDao包的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| hotelDao.insert | 语法 | public opMessage insert (hotelPO hotel) |
| 前置条件 | 同样的酒店记录在数据库中不存在 |
| 后置条件 | 向数据库中插入一条酒店记录 |
| hotelDao. query | 语法 | public List<HotelPO> query(QueryCondition con) |
| 前置条件 | 查询条件完整且规范 |
| 后置条件 | 返回符合查询条件的酒店酒店记录列表 |
| hotelDao. getAllHotels | 语法 | public List<HotelPO> getAllHotels() |
| 前置条件 |  |
| 后置条件 | 返回所有酒店记录 |
| hotelDao. getHotelById | 语法 | public getHotelById(String hoteld) |
| 前置条件 | 输入的酒店id在数据库中要存在 |
| 后置条件 | 返回该Id指向的所有酒店记录 |
| hotelDao.delete | 语法 | public opMessage delete (String hotelId) |
| 前置条件 | 该Id指向的酒店记录在数据库中要存在 |
| 后置条件 | 删除该酒店记录 |
| hotelDao.update | 语法 | public opMessage update(hotelPO hotel) |
| 前置条件 | 同样的酒店记录在数据库中要存在 |
| 后置条件 | 更新这条记录 |

roomDao包的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| roomDao.insert | 语法 | public opMessage insert(RoomPO room) |
| 前置条件 | 同样的房间记录在数据库中不存在 |
| 后置条件 | 向数据库中插入一条房间记录 |
| roomDao.delete | 语法 | public opMessage delete(String roomId) |
| 前置条件 | 该id指向的客房记录在数据库中不存在 |
| 后置条件 | 删除该id指向的客房记录 |
| roomDao.update | 语法 | public opMessage update(RoomPO room) |
| 前置条件 | 同样的客房记录在数据库中要存在 |
| 后置条件 | 更新这条记录 |
| roomDao.getRoomById | 语法 | public RoomPO getRoomById(String roomId) |
| 前置条件 | 该ID指向的客房记录在数据库中要存在 |
| 后置条件 | 返回这条客房记录 |
| roomDao.getallRoomsByHotel() | 语法 | public List<RoomPO> getAllRoomsByHotel(String hotelId) |
| 前置条件 | 该id指向的酒店记录在数据库中要存在 |
| 后置条件 | 返回该酒店的所有客房记录 |

memberDao包的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| memberDao.insert | 语法 | public opMessage insert(memberPO member) |
| 前置条件 | 同样的会员记录在数据库中不存在 |
| 后置条件 | 向数据库插入一条会员记录 |
| memberDao.delete | 语法 | public opMessage delete(String memberId) |
| 前置条件 | 该Id指向的会员记录在数据库中要存在 |
| 后置条件 | 删除这条记录 |
| memberDao.update | 语法 | public opMessage update(memberPO member) |
| 前置条件 | 同样的会员记录在数据库中要存在 |
| 后置条件 | 更新这条记录 |
| memberDao.query | 语法 | public List<MemberPO> query(QueryCondition con) |
| 前置条件 | 查询条件完整且规范 |
| 后置条件 | 返回符合查询条件的会员记录集合 |
| memberDao.getMemberById | 语法 | public MemberPO getMemberById(String MemberId) |
| 前置条件 | 该Id指向的会员记录在数据库中要存在 |
| 后置条件 | 返回这条记录 |
| memberDao.getMemberByName | 语法 | public MemberPO getMemberByName(String name) |
| 前置条件 | 会员名称正确 |
| 后置条件 | 返回这条会员记录 |
| memberDao.getAllMembers | 语法 | public List<MemberPO> getAllMembers() |
| 前置条件 |  |
| 后置条件 | 返回所有的会员记录列表 |

promotionDao包的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| promotionDao.insert | 语法 | public opMessage insert(PromotionVO promotion) |
| 前置条件 | 同样促销策略在数据库中不存在 |
| 后置条件 | 插入这条促销策略 |
| promotionDao.delete | 语法 | public opMessage delete(String promotionId) |
| 前置条件 | 该id指向的促销策略在数据库中要存在 |
| 后置条件 | 删除这条促销策略记录 |
| promotionDao.update | 语法 | public opMessage update(PromotionVO promotion) |
| 前置条件 | 同样的促销策略记录在数据库中要存在 |
| 后置条件 | 更新这条记录 |
| promotionDao.getPromotionById | 语法 | public PromotionVO getPromotionById(String id) |
| 前置条件 | 该id指向的促销策略记录在数据库中要存在 |
| 后置条件 | 返回这条记录 |
| promotionDao.getAllPromotionsByHotel | 语法 | public List<PromotionVO> getAllPromotionsByHotel(String HotelId) |
| 前置条件 | 该Id指向的酒店在数据库中要存在 |
| 后置条件 | 返回该酒店的所有促销策略列表 |
| promotionDao.getAllSitePromotions() | 语法 | public List<PromotionVO> getAllSitePromotions() |
| 前置条件 |  |
| 后置条件 | 返回网站所有促销策略的列表 |

userDao包的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的服务（供接口） | | |
| userDao.insert | 语法 | public opMessage insert(UserPO user) |
| 前置条件 | 同样的用户记录在数据库中不存在 |
| 后置条件 | 向数据库中插入一条用户记录 |
| userDao.delete | 语法 | public opMessage delete(String userId) |
| 前置条件 | 该id指向的用户记录在数据库中要存在 |
| 后置条件 | 删除这条记录 |
| userDao.update | 语法 | public opMessage update(UserPO user) |
| 前置条件 | 同样的用户记录在数据库中要存在 |
| 后置条件 | 更新这条记录 |
| userDao.getUserById | 语法 | public UserPO getUserById(String userId) |
| 前置条件 | 用户id完整且正确 |
| 后置条件 | 返回该id指向的用户记录 |
| userDao.getAllUsers | 语法 | public List<UserPO> getAllUsers() |
| 前置条件 |  |
| 后置条件 | 返回所有用户记录列表 |
| userDao.query | 语法 | public List<UserPO> query(QueryConditon con) |
| 前置条件 | 查询条件完整且正确 |
| 后置条件 | 返回符合条件的所有用户记录列表 |

* 1. **界面模块的设计**

系统共有42个界面，分别为:

系统主界面、登录界面

1. 客户**：**

客户主界面

个人信息管理主界面、个人信息显示界面、信用记录显示界面、个人信息修改界面

订单管理主界面、订单详细信息显示界面、预定过的酒店显示界面

订单生成界面

酒店评价界面

会员注册界面

酒店搜索界面、酒店详细信息界面

1. 酒店工作人员**：**

酒店工作人员主界面

酒店信息显示界面、酒店信息修改界面

客房信息显示界面、客房信息修改界面

酒店促销策略显示界面、酒店促销策略修改界面

客户入住界面

客户退房界面

浏览订单界面、订单详细信息显示界面

1. 网站营销人员**：**

网站营销人员主界面

制定促销策略主界面、双十一促销界面、VIP会员折扣界面、会员等级制定界面

异常订单浏览界面、异常订单详细信息显示界面、异常订单撤销界面

信用充值界面

1. 网站管理人员**：**

网站管理人员主界面

用户查询界面、用户信息显示界面、用户信息修改界面、网站营销人员添加界面、酒店添加界面、酒店工作人员添加界面

界面跳转整体设计如下图所示：

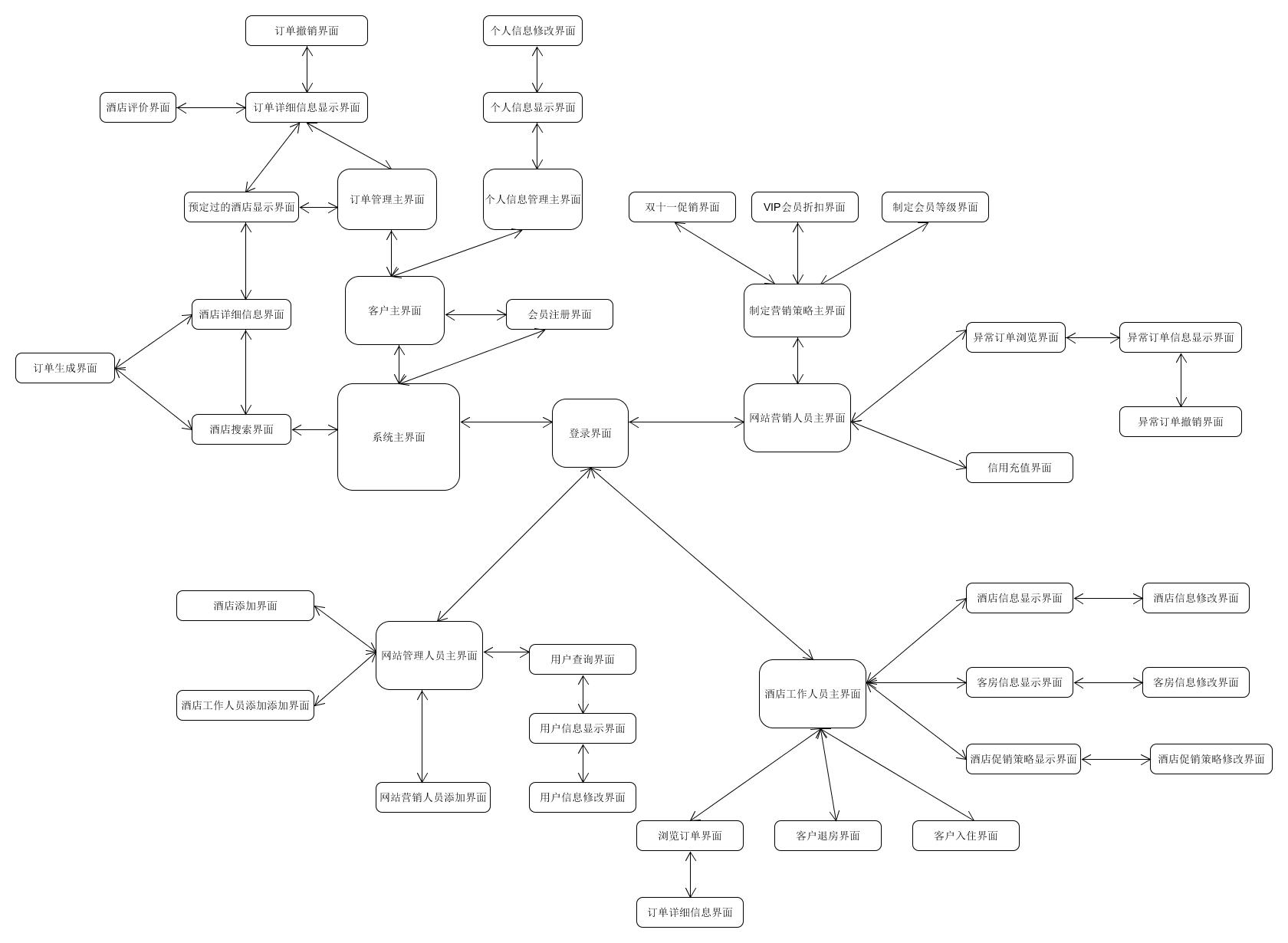


图7 界面跳转整体设计