物流管理系统要解决的问题主要有以下几个模块：分为基础模块、取派模块、中转流程管理和两个系统管理模块。系统管理模块：该模块有权限管理和用户管理两个功能。

登录页面

如下：



登录后首页，有菜单导航：基本功能、系统管理；

消息中心有公共栏、代办事宜、预警信息、系统反馈



系统管理功能：用户管理、角色管理、功能权限管理、流程定义管理、查看正在运行的任务

用户管理：实现系统用户的添加、查询和删除：

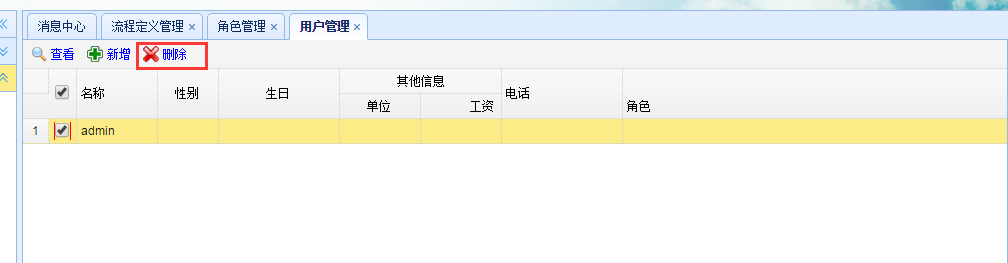
查询如图所示：



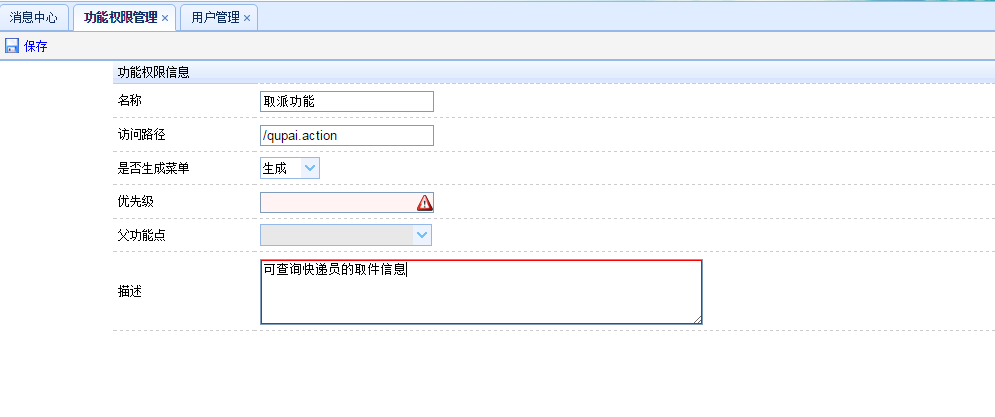
新增功能如图所示：



删除功能如图所示：



功能权限管理：为菜单添加功能权限，实现为用户赋予功能权限。如图所示：



角色管理：为用户添加角色权限



基础功能菜单包括：

基础数据模块、受理模块、调度模块、中转配送流程管理模块

基础数据模块包括：收派标准、取派员设置、区域设置、管理分区、管理定区/调试排班

收派标准主要功能是添加快递收费标准基础数据，提供了查询、增加、删除功能，具体如下图所示：



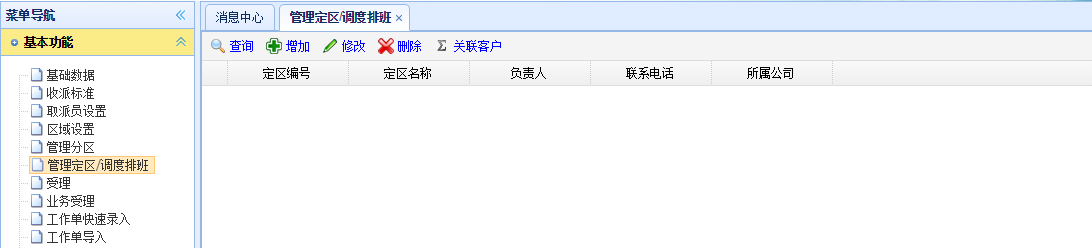
取派员设置添加快递员基本信息，其功能包括 查询、增加、作废、还原，具体页面如下图所示：



管理分区：实现地区域的划分，具体功能有查询、增加、修改、删除、还支持批量数据的导入和导出。页面如下图所示：



管理定区：实现为每个区域安排固定快递员，该模块还可以实现人为调班功能，操作按钮有查询、增加、修改、删除、关联客户，如下图所示：



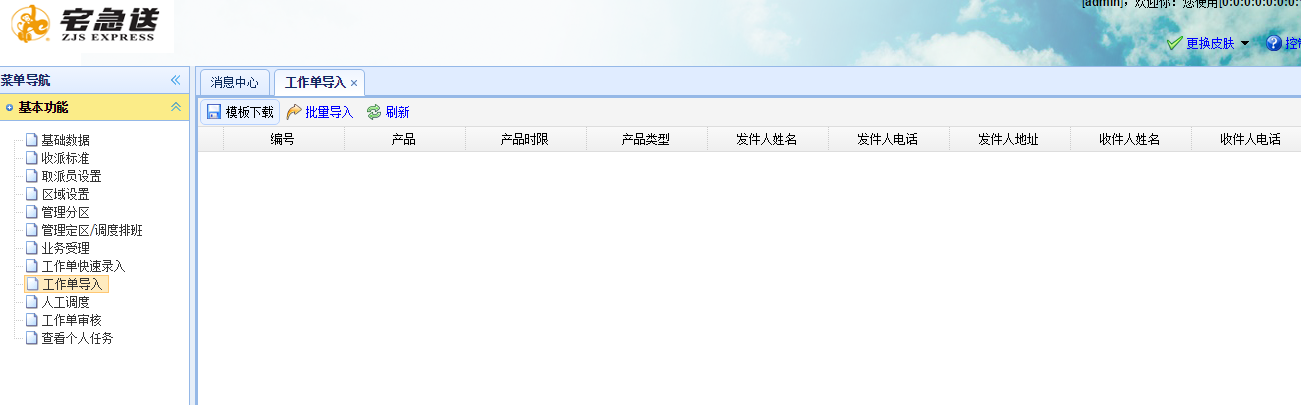
业务受理：实现客户信息的操作处理，具体有新增工单信息，更新工单信息，如下图所示：



工作单快速录入：为了方便操作员快速录入信息，可使用“新增一行”，实现快速录入，如下图所示：



工作单导入：实现批量数据的导入，如图所示：



人工调度：实现疑难件人工介入处理，修改流程。如下图所示：



工作单审核：实现快件审核，如图所示：



查看个人任务：实现个人任务的管理，如图所示：



主要实现代码附录

基础 action类

/\*\*

\* 公共 抽象Action （实现代码复用）

\*

\* **@author** seawind

\*

\*/

**public** **abstract** **class** BaseAction **extends** ActionSupport {

@Resource(name = "userService")

**protected** UserService userService;

@Resource(name = "standardService")

**protected** StandardService standardService;

@Resource(name = "staffService")

**protected** StaffService staffService;

@Resource(name = "regionService")

**protected** RegionService regionService;

@Resource(name = "subareaService")

**protected** SubareaService subareaService;

@Resource(name = "decidedZoneService")

**protected** DecidedZoneService decidedZoneService;

@Resource(name = "customerService")

**protected** CustomerService customerService;

@Resource(name = "noticeBillService")

**protected** NoticeBillService noticeBillService;

@Resource(name = "workOrderManageService")

**protected** WorkOrderManageService workOrderManageService;

@Resource(name = "functionService")

**protected** FunctionService functionService;

@Resource(name = "roleService")

**protected** RoleService roleService;

@Resource(name = "processEngine")

**protected** ProcessEngine processEngine;

@Resource(name = "bosTaskService")

**protected** BosTaskService bosTaskService;

// 属性驱动，封装分页参数

**protected** **int** page;

**protected** **int** rows;

**public** **void** setPage(**int** page) {

**this**.page = page;

}

**public** **void** setRows(**int** rows) {

**this**.rows = rows;

}

// 封装PageRequestBean

**public** PageRequestBean initPageRequestBean(DetachedCriteria detachedCriteria) {

// 将分页查询参数 ，封装 PageRequestBean 对象中

PageRequestBean pageRequestBean = **new** PageRequestBean();

pageRequestBean.setPage(page);

pageRequestBean.setRows(rows);

pageRequestBean.setDetachedCriteria(detachedCriteria);

**return** pageRequestBean;

}

}

基础baseService类

/\*\*

\* 业务层 抽象 （实现业务层代码复用）

\*

\* **@author** seawind

\*

\*/

**public** **abstract** **class** BaseService {

@Resource(name = "userDAO")

**protected** GenericDAO<User> userDAO;

@Resource(name = "standardDAO")

**protected** GenericDAO<Standard> standardDAO;

@Resource(name = "staffDAO")

**protected** GenericDAO<Staff> staffDAO;

@Resource(name = "regionDAO")

**protected** GenericDAO<Region> regionDAO;

@Resource(name = "subareaDAO")

**protected** GenericDAO<Subarea> subareaDAO;

@Resource(name = "decidedZoneDAO")

**protected** GenericDAO<DecidedZone> decidedZoneDAO;

@Resource(name = "noticeBillDAO")

**protected** GenericDAO<NoticeBill> noticeBillDAO;

@Resource(name = "workBillDAO")

**protected** GenericDAO<WorkBill> workBillDAO;

@Resource(name = "customerService")

**protected** CustomerService customerService;

@Resource(name = "workOrderManageDAO")

**protected** GenericDAO<WorkOrderManage> workOrderManageDAO;

@Resource(name = "functionDAO")

**protected** GenericDAO<Function> funtionDAO;

@Resource(name = "roleDAO")

**protected** GenericDAO<Role> roleDAO;

@Resource(name = "processEngine")

**protected** ProcessEngine processEngine;

@Resource(name = "zhongZhuanInfoDAO")

**protected** GenericDAO<ZhongZhuanInfo> zhongZhuanInfoDAO;

@Resource(name = "transferInfoDAO")

**protected** GenericDAO<TransferInfo> transferInfoDAO;

@Resource(name = "inStoreDAO")

**protected** GenericDAO<InStore> inStoreDAO;

@Resource(name = "outStoreDAO")

**protected** GenericDAO<OutStore> outStoreDAO;

@Resource(name = "receiveGoodsInfoDAO")

**protected** GenericDAO<ReceiveGoodsInfo> receiveGoodsInfoDAO;

// 分页通用代码

**public** <T> PageResponseBean pageQuery(PageRequestBean pageRequestBean, GenericDAO<T> dao) {

PageResponseBean pageResponseBean = **new** PageResponseBean();

// 查询当前页显示数据

**int** firstResult = (pageRequestBean.getPage() - 1) \* pageRequestBean.getRows(); // 　从哪条开始

**int** maxResults = pageRequestBean.getRows(); // 返回记录数

List<T> data = dao.pageQuery(pageRequestBean.getDetachedCriteria(), firstResult, maxResults);

pageResponseBean.setRows(data);

// 满足当前条件，记录总条数

**long** total = dao.findTotalCount(pageRequestBean.getDetachedCriteria());

pageResponseBean.setTotal(total);

**return** pageResponseBean;

}

}