lab7 体系结构文档

Table of Contents

lab7 体系结构文档
- 1. 修订记录
- 2. 引言
  - 2.1 编制目的
  - 2.2 参考资料
- 3. 产品概述
- 4. 逻辑视角
- 5. 组合视角
  - 5.1 开发包图
  - 5.2 运行时进程
  - 5.3 部署图
- 6. 接口视角
  - 6.1 模块化视图
  - 6.2 用户界面分解
    - 6.2.1 财务管理accountui
    - 6.2.2 经营管理bussinessui
    - 6.2.3 员工打卡 clockInui
    - 6.2.4 员工打卡情况 clockInui
    - 6.2.5 员工信息 hrManagementui
    - 6.2.6 工资单 salaryui
    - 6.2.7 促销策略 discountui
  - 6.3 网络层分解
  - 6.4 业务逻辑层分解
  - 6.5 数据层分解

1. 修订记录

姓名	修订版本	时间	修改内容
倪奕晨	v1.0.0	2022-7-1	创建并完成引言、产品概述、运行进程、部署图、模块视图
邓楚宸	v1.2.0	2022-7-7	完成网络层、逻辑层、数据层部分接口视角
何青云	v1.2.1	2022-7-7	补充查看销售明细、经营历程、经营情况的接口视角
倪奕晨	v1.3.0	2022-7-8	补充用户界面接口视角
邓楚宸	v1.4.0	2022-7-9	完成开发包图服务端部分
倪奕晨	v1.5.0	2022-7-9	完成逻辑包图，补充开发包图浏览器端部分
苏豪	v1.6.0	2022-7-9	补充审批工资单、审批赠品单、制定促销策略、制定年终奖、员工信息管理、员工打卡、查看员工打卡情况、制定工资单的接口视角
邓楚宸	v1.6.1	2022-7-10	修正接口视角内容

2. 引言

2.1 编制目的

本报告详细完成对ERP系统的概要设计，达到详细设计和开发的目的，同时实现和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而编写，是了解系统的导航。

2.2 参考资料

1）骆斌,丁二玉,刘钦.软件开发的技术基础[M].机械工业出版社, 2012.

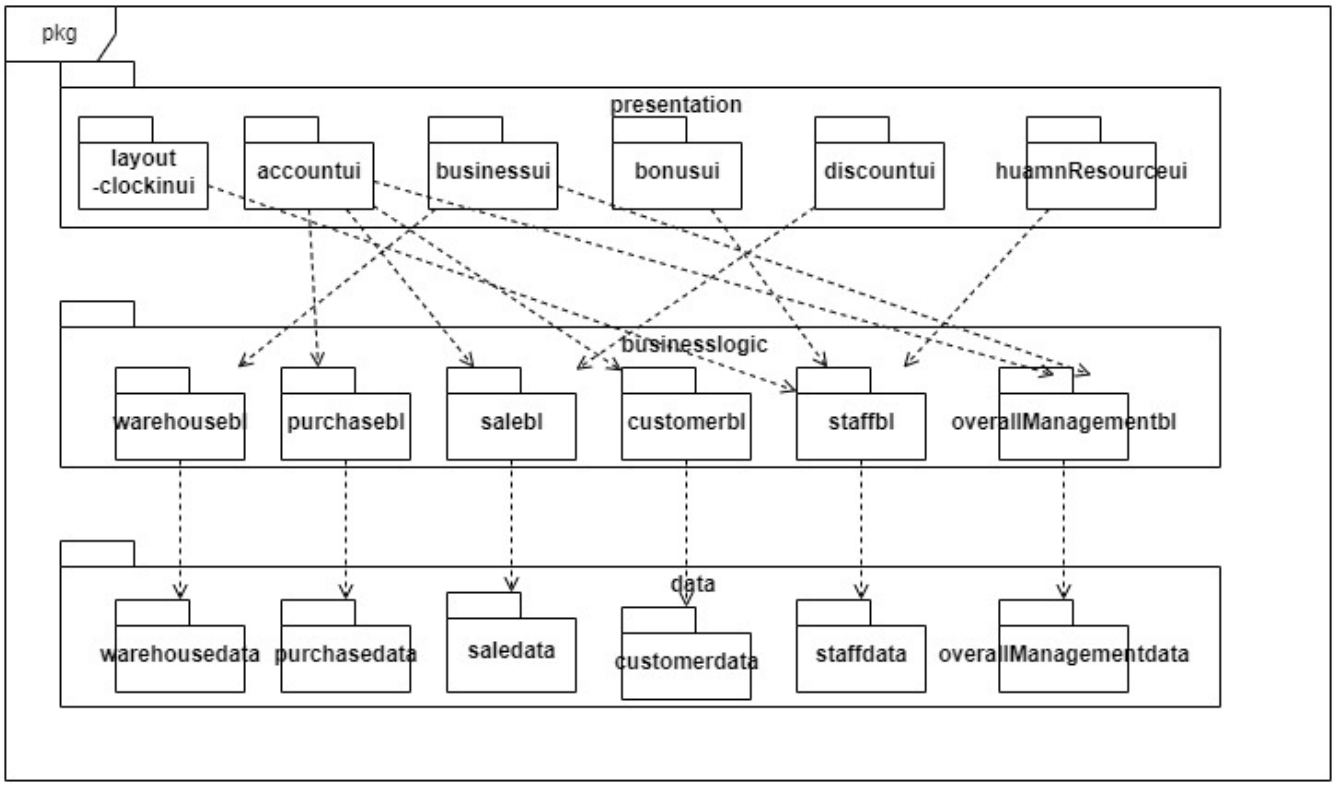
3. 产品概述

一民营企业专业从事灯具开关行业，是某著名开关品牌的南京地区总代理，主要在南京负责品牌的推广及项目的落地销售、分销的批发等工作，服务对象包括项目业主、施工单位、分销商、设计院、终端用户等。现公司规模扩大，企业业务量、办公场所、员工数都发生增长，希望可以帮助员工适应新的环境，提高工作效率和用户满意度。

该ERP系统就是为满足该公司新阶段业务发展要求而开发的，主要包括库存管理、销售管理、财务管理、人事管理和企业经营管理。

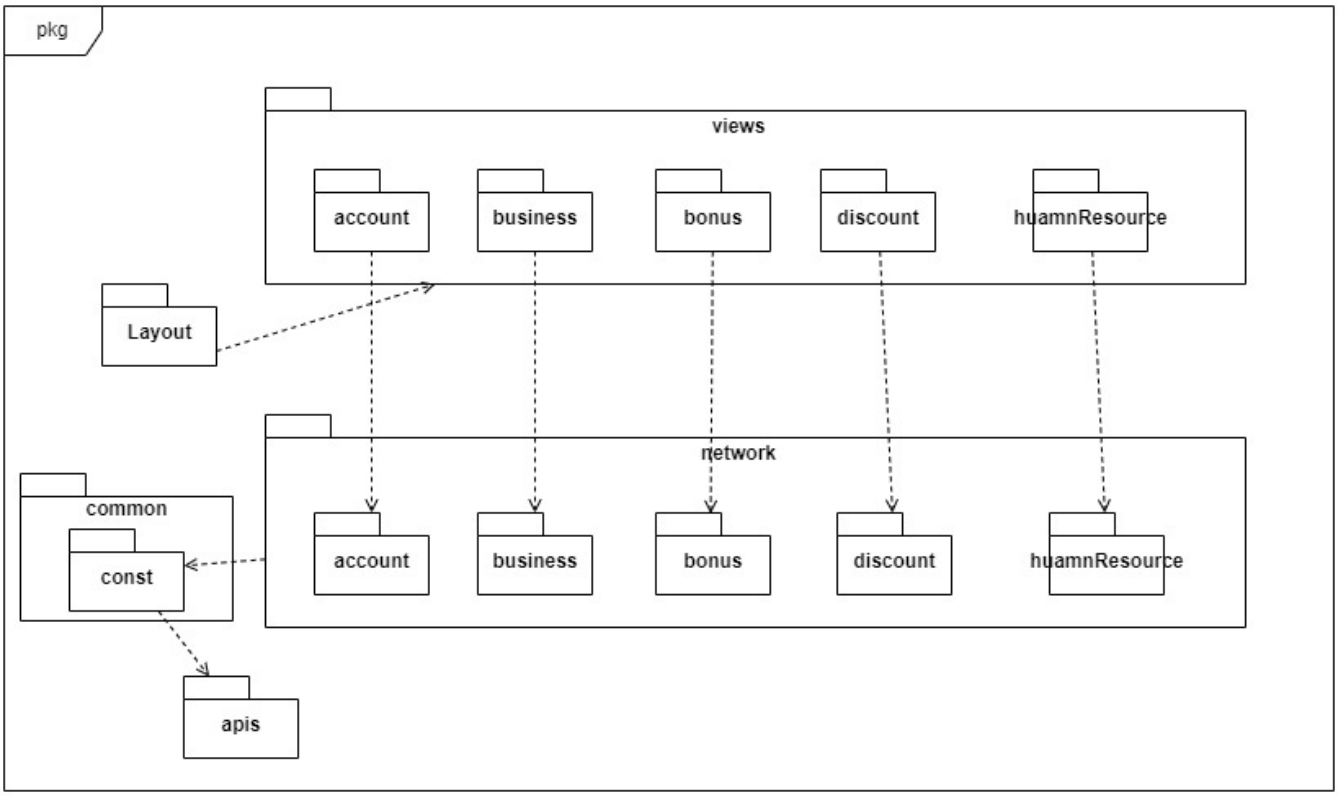
4. 逻辑视角

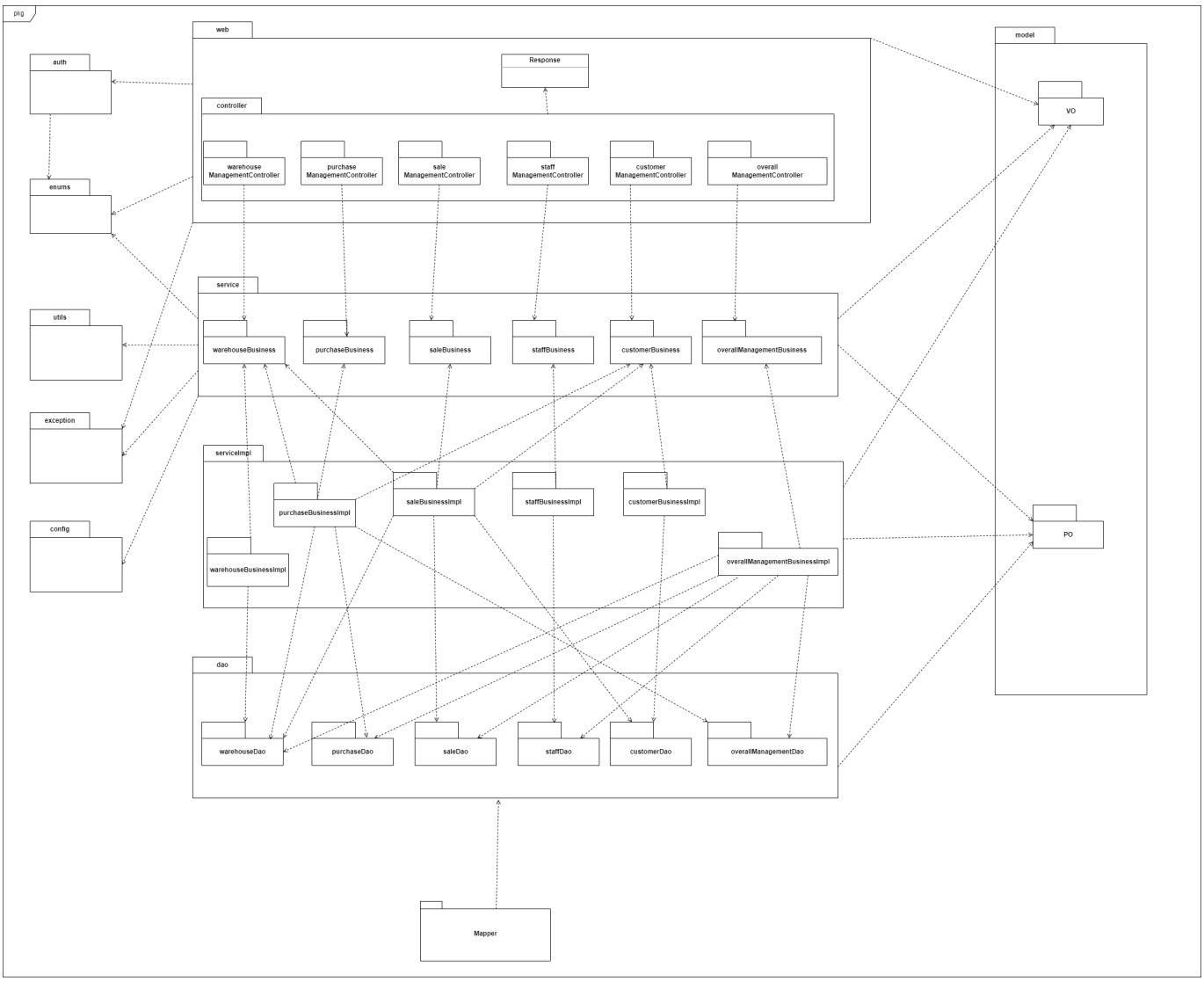
影院管理系统中，选择了分层体系结构风格，将系统分为3层（展示层、业务逻辑层、数据层）能够很好地示意整个高层抽象。展示层包含GUI页面的实现，业务逻辑层包含业务逻辑处理的实现，数据层负责数据的持久化和访问。



5. 组合视角

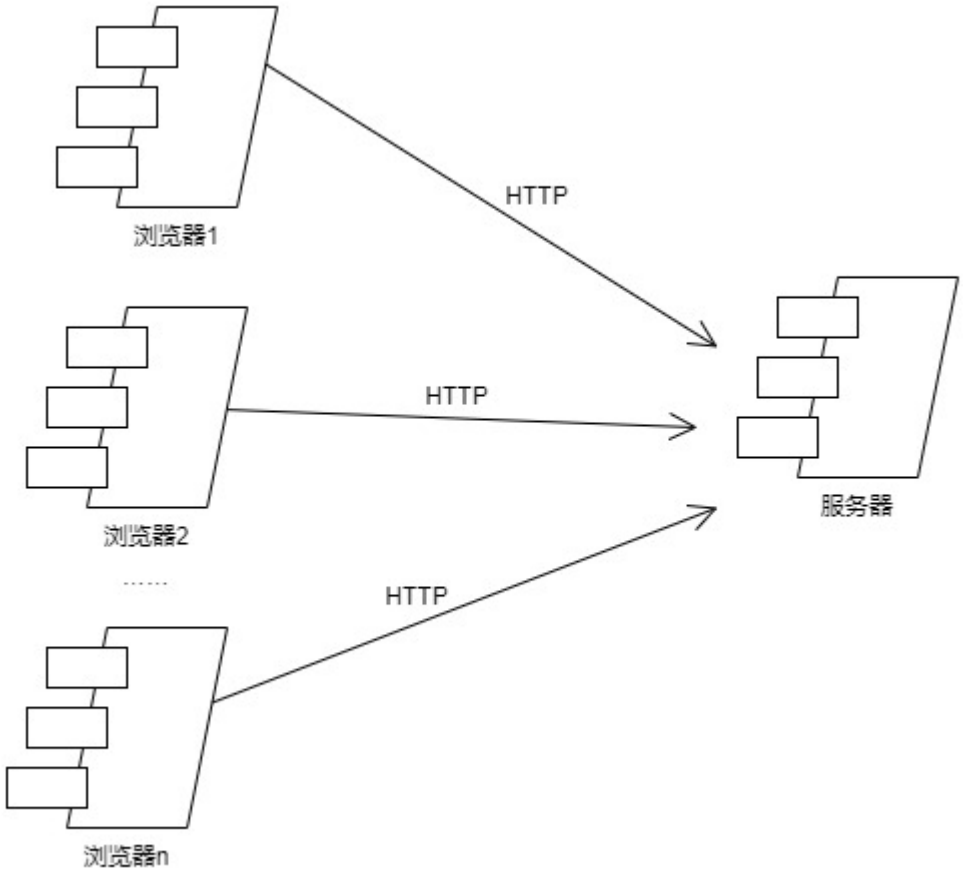
5.1 开发包图


pkg
views
account
business
bonus
discount
huamnResource
Layout
network
common
const
account
business
bonus
discount
huamnResource
apis



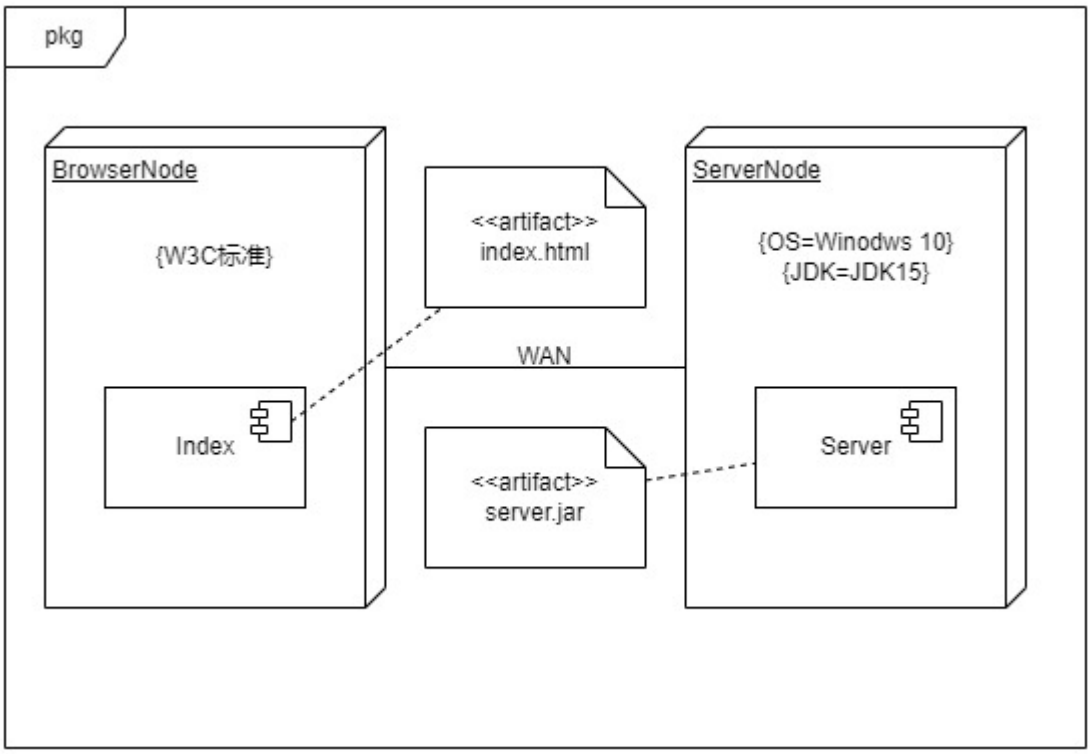
5.2 运行时进程

在ERP系统中，会有多个浏览器端进程和一个服务器端进程，其进程图如下所示。结合部署图，浏览器端进程是在用户浏览器上运行，服务器端进程是在服务器机器上运行。



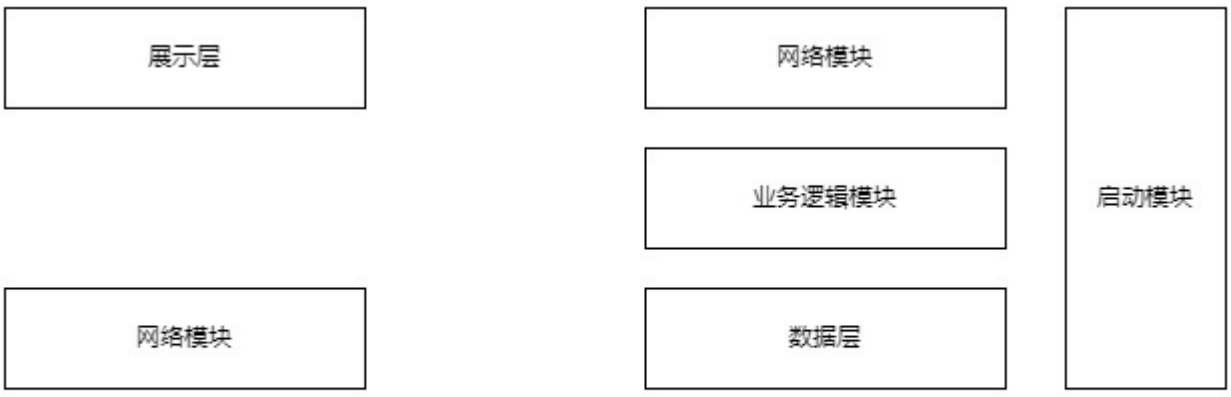
5.3 部署图

ERP系统中浏览器构建是放在浏览器端的机器上，服务器构建是放在服务器机器上。在浏览器端要部署Index构件。



6. 接口视角

6.1 模块化视图



浏览器端模块视图

服务器端模块视图

层	职责
展示层	基于网络的ERP系统浏览器端用户界面
浏览器端网络模块	利用HTTP协议访问服务器端服务
服务器端网路模块	利用HTTP协议接受浏览器端数据以及返回数据
业务逻辑模块	对浏览器端的请求进行响应并执行业务处理逻辑
数据层	负责数据的持久化及数据访问接口
启动模块	初始化数据库

接口	服务调用方	服务提供方
customerBusiness overallManagementBusiness purchaseManagementBusiness saleBusiness staffBusiness warehouseBusiness	网络模块	业务逻辑层
customerDao overallManagementDao purchaseDao saleDao staffDao warehouseDao	业务逻辑层	数据层

6.2 用户界面分解

根据需求，系统存在13个用户界面：收付款管理、账户管理、查看销售明细表、查看经营情况表、查看经营历程表、审批工资单、审批赠品单、制定促销策略、制定年终奖、员工信息管理、员工打卡、查看员工打卡情况、制定工资单

6.2.1 财务管理accountui

职责：账户管理&收付款管理页面的显示

接口	path
getALLAccount()	/api/company-bank-account-management/find-all-account
getParticularAccount()	/api/company-bank-account-management/find-company-bank-account
deleteAccount()	/api/company-bank-account-management/delete-company-bank-account
createAccount()	/api/company-bank-account-management/create-company-bank-account
changeAccount()	/api/company-bank-account-management/change-company-bank-account
createPay()	/api/payable/payable-sheet-make
getAllCustomer()	/api/customer/show-all-customer-id
createReceive()	/api/receivable/receivable-sheet-make

6.2.2 经营管理bussinessui

职责：查看经营历程表、经营情况表、销售明细表

接口	path
getSalesAccount()	/api/sale-status/sale-status-show
getAllUsers()	/api/user/find-all-user
getBusinessCondition()	/api/business-condition/business-condition-show
getBusinessProcess()	/api/business-process/business-process-show

6.2.3 员工打卡 clockInui

职责：员工每日打卡，检验员工是否打卡

接口	path
FIND_STAFFID	/api/attendance/findStaffIdByName
ATTENDANCE_RECORD_REGISTER	/api/attendance/register
THIS_DAY_ATTENDANCE	/api/attendance/thisDayAttendance

6.2.4 **员工打卡情况** clockInui

职责：查询员工打卡情况，查看某段时间所有员工打卡情况

接口	path
FIND_ALL_ATTENDANCE_RECORD	/api/attendance/findAll
FIND_SOMEBODY_ATTENDANCE_RECORD	/api/attendance/findStaffIdByName

6.2.5 **员工信息** hrManagementui

职责：添加、编辑员工信息，编辑薪资发放方式

接口	path
MAKE_STAFF_ACCOUNT	/api/staff/staff-account-make
SHOW_STAFF_ACCOUNT	/api/staff/staff-account-show
STAFF_ACCOUNT_CHANGE	/api/staff/staff-account-change

6.2.6 **工资单** salaryui

职责：制作工资单、工资单审批

接口	path
MAKE_SALARY_SHEET	/api/salary/sheet-make
SHOW_SHEET_BY_STATE	/api/salary/sheet-show
SECOND_APPROVAL	/api/salary/second-approval
MAKE_SALARY_RULE	/api/salary/salaryRuleChange
SHOW_PREV_SALARY	/api/salary/prev-salary
MAKE_ANNUAL_BONUS	/api/salary/set-annual-bonus
SHOW_ALL_SHEET	/api/salary/show-all-sheet
GET_ANNUAL_AMOUNT	/api/salary/get-annual-bonus

6.2.7 **促销策略** discountui

职责：促销策略制定以及显示、二级审批赠品单

接口	path
ADDDISCOUNT	/api/promotion/discount
ADDGIFT	/api/promotion/gift
ADDVOUCHER	/api/promotion/voucher
GET_PROMOTION	/api/promotion/getPromotion
DISCOUNTLIST_APPROVAL	/api/promotion/approve

6.3 网络层分解

负责前后端的交互

具体职责&接口规范如下

账户管理
web.controller.overallManagementController.CompanyBankAccountController

供接口:

```
@RequestMapping(path = "/company-bank-account-management")


@PostMapping(value = "/create-company-bank-account")
    public Response createBankAccount(@RequestBody CompanyBankAccountVO
companyBankAccountVO);


@PostMapping(value = "/delete-company-bank-account")
    public Response deleteBankAccount(@RequestBody String id);


@PostMapping(value = "/change-company-bank-account-name")
    public Response changeBankAccountName(@RequestBody CompanyBankAccountVO
companyBankAccountVO);


@GetMapping(value = "/find-company-bank-account")
    public Response findByBankAccountName(@RequestParam(value =
"companyBankAccountName") String companyBankAccountName);


 @GetMapping(value = "/find-all-account")
    public Response findAllBankAccount();
```

需接口:
来自service.overallManagementBusiness.CompanyBankAccountService的

```
createCompanyBankAccount(CompanyBankAccountVO vo);
deleteCompanyBankAccount(String id);
changeCompanyBankAccountName(CompanyBankAccountVO vo);
getCompanyBankAccountByName(String companyBankAccountName);
getAllCompanyBankAccount();
```

制订促销策略(包括查看、审批库存赠送单)
web.controller.saleManagementController.PromotionController

供接口：

```
@RequestMapping(path = "/promotion")

@PostMapping(value = "/voucher")
    public Response addVoucher(@RequestBody PromotionPO promotionPO)

@PostMapping(value = "/discount")
    public Response addDiscount(@RequestBody PromotionPO promotionPO);

@PostMapping(value = "/gift")
    public Response addGift(@RequestBody PromotionPO promotionPO);

@GetMapping(value = "/getGiftSheetById")
    public Response getGiftSheetById(@RequestParam("GiftSheetId") String id);

@GetMapping(value = "/approve")
    public Response approve(@RequestParam("GiftSheetId") String id,
                            @RequestParam("state")GiftSheetState state);

@GetMapping(value = "/getGiftSheetByState")
    public Response getGiftSheetById(@RequestParam("state")GiftSheetState state)
```

需接口：

来自service.saleManagementBusiness.PromotionStrategyService的

```
addVoucher(PromotionPO po);
addDiscount(PromotionPO po);
addGift(PromotionPO po);
```

来自service.saleManagementBusiness.GiftService的

```
approval(String id,GiftSheetState state);
getGiftSheetByState(GiftSheetState state);
```

制订、查看、审批收款单
web.controller.saleManagementController.ReceivableController

供接口：

```
@RequestMapping(path = "/receivable")

@PostMapping(value = "/receivable-sheet-make")
    public Response makeReceipt(UserVO userVO, @RequestBody ReceivableSheetVO
receivableSheetVO)


@GetMapping(value = "/second-approval")
    public Response secondApproval(@RequestParam("receivableSheetId") String
receivableSheetId,
                                   @RequestParam("state") ReceivableSheetState state)

@GetMapping(value = "/find-all-receivable-sheet")
    public Response findAllReceivable()
```

需接口：

来自service.purchaseManagementBusiness.PayableService的

```
makePayableSheet(UserVO vo, PayableSheetVO vo2);
approval(String payableSheetId,PayableSheetState state);
getAllPayableSheet();
```

制订、查看、审批付款单
web.controller.purchaseManagementController.PayableController

供接口：

```
@RequestMapping(path = "/payable")


@PostMapping(value = "/payable-sheet-make")
    public Response makePaymentOrder(UserVO userVO, @RequestBody PayableSheetVO
payableSheetVO)


@GetMapping(value = "/second-approval")
    public Response secondApproval(@RequestParam("payableSheetId") String
payableSheetId,
                                   @RequestParam("state") PayableSheetState state)


 @GetMapping(value = "/find-all-payable-sheet")
    public Response findAllPayable();
```

需接口：

来自service.saleManagementBusiness.ReceivableService的

```
makeReceivableSheet(UserVO vo, ReceivableSheetVO vo2);
approval(String receivableSheetId,ReceivableSheetState state);
getAllReceivableSheet();
```

制订薪酬规则、制订、审批、查看工资单、制订年终奖
web.controller.staffManagementController.SalaryController

供接口：

```
@RequestMapping(path = "/salary")

@PostMapping(value = "/sheet-make")
    public Response makePurchaseOrder(UserVO userVO, @RequestBody SalarySheetVO
salarySheetVO)


@GetMapping(value = "/second-approval")
    public Response secondApproval(@RequestParam("SalarySheetId") String
salarySheetId,
                                   @RequestParam("state") SalarySheetState state)

@GetMapping(value = "/sheet-show")
    public Response showSheetByState(@RequestParam(value = "state", required = false)
SalarySheetState state)


@GetMapping(value = "/find-sheet")
    public Response findBySheetId(@RequestParam(value = "id") String id)


@GetMapping(value = "/salaryRuleChange")
    public Response makeSalaryRule(@RequestParam("staffId") String staffId,
                                   @RequestParam("rule")SalaryRule rule);

@GetMapping(value = "/prev-salary")
    public Response showPrevSalary(@RequestParam("staffId") String staffId)

@GetMapping(value = "/get-annual-bonus")
    public Response getAnnualAmount(@RequestParam("staffId") String staffId)

@GetMapping(value = "/set-annual-bonus")
    public Response makeAnnualAmount(@RequestParam("staffId") String staffId,
                                   @RequestParam("annualAmount")BigDecimal
annualAmount)
```

需接口：

来自service.staffManagementBusiness.SalaryService的

```
makeSalarySheet(UserVO vo, SalarySheetVO vo2);
approval(String salarySheetId,SalarySheetState state);
getSalarySheetByState(SalarySheetState state);
getSalarySheetById(String id);
```

```
makeSalaryRule(String staffId,SalaryRule rule);
checkPrevSalary(String staffId)
```
来自service.staffmanagementBusiness.AnnualBonus的
```
findAnnualAmountById(String staffId)
setAnnualAmount(String staffId, BigDecimal annualAmount)
```

查看销售明细表
web.controller.overallManagementController.SaleStatusController
供接口：
```
@RequestMapping(path = "/sale-status")
@GetMapping(value = "/sale-status-show")
    public Response showSaleStatus(@RequestParam("startTime") String startTime,
@RequestParam("endTime") String endTime,
                                   @RequestParam("name") String name,
@RequestParam("supplier") Integer supplier,
                                   @RequestParam("salesman") String salesman);
```

需接口：

来自service.overallManagementBusiness.SaleStatusService的

```
getSaleStatusSheet(startTime, endTime, name, supplier, salesman);
```

查看经营历程表
web.controller.overallManagementController.BusinessProcessController
供接口：

```
@RequestMapping(path = "/business-process")
@GetMapping(value = "/business-process-show")
    public Response showSaleStatus(@RequestParam("sheetType") String
sheetType,@RequestParam("startTime") String startTime,
                                   @RequestParam("endTime") String endTime,
@RequestParam("supplier") Integer supplier,
                                   @RequestParam("salesman") String salesman);
```

需接口：

来自service.overallManagementBusiness.BusinessProcessService的

```
findSaleSheet(startTime, endTime, supplier, salesman);
findSaleReturnsSheet(startTime, endTime, supplier, salesman);
findPurchaseSheet(startTime, endTime, supplier, salesman);
findPurchaseReturnsSheet(startTime, endTime, supplier, salesman);
findPayableSheet(startTime, endTime, supplier, salesman);
findReceivableSheet(startTime, endTime, supplier, salesman);
findSalarySheet(startTime, endTime, supplier, salesman);
```

查看经营情况表
web.controller.overallManagementController.BusinessConditionController
供接口：

```
@RequestMapping(path = "/business-condition")
@GetMapping(value = "/business-condition-show")
    public Response showSaleStatus(@RequestParam("startTime") String startTime,
@RequestParam("endTime") String endTime);
```

需接口：
来自service.overallManagementBusiness.BusinessConditionService的

```
getBusinessConditionSheet(startTime, endTime);
```

员工管理: 增加改变查询员工信息
web.controller.staffManagementController.StaffController
供接口：

```
@RequestMapping(path = "/staff")
@PostMapping(value = "/staff-account-make")
    public Response makeStaffAccount(@RequestBody StaffInformationVO
staffInformationVO);


@PostMapping(value = "/staff-account-change")
    public Response changeStaffAccountInfo(@RequestBody StaffInformationVO
staffInformationVO);


@PostMapping(value = "/staff-account-show")
    public Response showStaffAccount();
```

需接口：
来自service.staffBusiness.StaffAccountService的

```
creatStaffAccount(StaffInformationVO vo);
changeStaffInfo(StaffInformationVO vo);
showAllStaffAccount();
```

员工打卡
web.controller.staffManagementController.DailyAttendanceController

供接口：

```
@RequestMapping(path = "/attendance")

@GetMapping (value = "/thisDayAttendance")
    public Response doesThisDayHeAttend(@RequestParam("date")String date,
                                        @RequestParam("staffId")String staffId)
@GetMapping(value = "/findAll")
    public Response findAllAttendanceRecord(@RequestParam("yearAndMonth")String
yearAndMonth)

@GetMapping(value = "/findSomebody")
    public Response findSomebodyAttendanceRecord(@RequestParam("yearAndMonth")String
yearAndMonth,
                                                 @RequestParam("staffId")String
staffId)
@GetMapping(value = "/register")
    public Response attendanceRecordRegister(@RequestParam("staffId")String staffId,
                                             @RequestParam("name")String name,
                                             @RequestParam("role")String role)
@GetMapping(value = "/findStaffIdByName")
    public Response findStaffIdByName(@RequestParam ("name")String name)
```

需接口：

来自service.staffBusiness.DailyAttendanceService的

```
doesThisDayHeAttend(String staffId,String date);
findALLStaffAttendance(String yearAndMonth);
findOneStaffAttendance(String yearAndMonth,String staffId);
attendanceRegister(String staffId,String name,String role);
findIdByName(String name);
```

6.4 业务逻辑层分解

负责接受网络层的控制并调用数据层接口来实现具体业务逻辑

serviceImpl实现service

具体职责&接口规范如下

员工打卡：

serviceImpl.staffbusinessImpl.DailyAttendanceServiceImpl

供接口：

```
    /**
     *  查询某人是否某日已打卡
     * @param staffId 员工id
     * @param date
     */
    boolean doesThisDayHeAttend(String staffId, String date);

    /**
     * 查询所有员工某年某月打卡记录
     * @param date
     */
    List<AttendanceVO> findALLStaffAttendance(String date);

    /**
     *  查询某位员工某年某月打卡记录
     * @param date
     * @param staffId 员工id
     */
    List<AttendanceVO> findOneStaffAttendance(String date,String staffId);

    /**
     *  打卡
     * @param staffId 员工id
     */
    void attendanceRegister(String staffId,String name,String role);

    String findIdByName(String name);
```

需接口：

来自dao.staffDao.DailyAttendanceDao的：

```
doesThisDayHeAttend(String staffId,Date beginDate,Date endDate);
findAll(Date beginDate,Date endDate)
findOne(Date beginDate,Date endDate,String staffId)
findIdByName(String staffName)
AttendanceRegister(AttendancePo po);
```

员工管理: 增加改变查询员工信息

serviceImpl.staffBusinessImpl.StaffAccountServiceImpl

供接口：

```
    /**
     * 登记入职员工信息，系统创建账户
     * @param staffInformationVO
     */
    void creatStaffAccount(StaffInformationVO staffInformationVO);
    /**
     * 修改入职员工信息
     * @param staffInformationVO
     */
    void changeStaffInfo(StaffInformationVO staffInformationVO);
    /**
     * 根据staffId找员工
     * @param staffId
     */
    StaffInformationPO findByStaffId(Integer staffId);

    List<StaffInformationVO> showAllStaffAccount();
```

需接口：

来自dao.staffDao.StaffAccountDao的：

```
     /**
     * 存入一条员工信息记录
     * @param toSave 一条工资单记录
     * @return 影响的行数
     */
    int saveAccount(StaffInformationPO toSave);

    /**
     * 根据id寻找员工
     * @param staffId
     * @return
     */
    StaffInformationPO findById(Integer staffId);

    /**
     * 更新员工信息
     * @param toChange
     * @return
     */
    int changeInfo(StaffInformationPO toChange);

    List<StaffInformationPO> findAll();
```

账户管理：

serviceImpl.overallManagementBusinessImpl.CompanyBankAccountServiceImpl

供接口：

```
    /**
     * 新建公司银行账户
     * @param companyBankAccountVO 公司银行账户
     */
    void createCompanyBankAccount(CompanyBankAccountVO companyBankAccountVO);
    /**
     * 删除公司银行账户
     * @param id 公司银行账户id
     */
    void deleteCompanyBankAccount(String id);
    /**
     * 修改账户属性(只能修改账户名)
     * @param companyBankAccountVO 公司银行账户
     */
    void changeCompanyBankAccountName(CompanyBankAccountVO companyBankAccountVO);
    /**
     * 根据账户名模糊查找
     * @param companyBankAccountName 公司账户名
     * @return 模糊查找得到的账户
     */
    List<CompanyBankAccountVO> getCompanyBankAccountByName(String
companyBankAccountName);

    List<CompanyBankAccountVO> getAllCompanyBankAccount();
```

需接口：
来自dao.overallManagementDao.CompanyBankAccountDao的：

```
    /**
     * 存入新增加的公司账户
     * @param toSave
     * @return 影响的行数
     */
    int saveAccount(CompanyBankAccountPO toSave);
    /**
     * 删除公司账户
     * @param companyBankAccountName
     * @return 影响的行数
     */
    int deleteAccount(String companyBankAccountName);
    /**
     * 修改公司账户属性（名称）
     * @param id, companyBankAccountName
```

```
     */
    int changeAccountName(String id, String companyBankAccountName);

    /**
     * 根据账户名模糊查找
     * @param companyBankAccountName 公司账户名
     * @return 模糊查找得到的账户
     */
    List<CompanyBankAccountPO> findAllByName(String companyBankAccountName);

    List<CompanyBankAccountPO> findAll();
```

制订促销策略：

serviceImpl.saleManagementBusinessImpl.PromotionStrategyServiceImpl

供接口：

```
    /**
     * 制订代金券策略
     * @param po
     */
    void addVoucher(PromotionPO po);

    /**
     * 制订赠送策略
     * @param po
     */
    void addGift(PromotionPO po);

    /**
     * 制订折扣策略
     * @param po
     */
    void addDiscount(PromotionPO po);
```

需接口：

来自dao.saleDao.SalarySheetDao

供接口：

```
    /**
     * 获取最近一条工资单
     */
    SalarySheetPO getLatestSheet();

    /**
     * 存入一条工资单记录
     * @param toSave 一条工资单记录
     * @return 影响的行数
     */
    int saveSheet(SalarySheetPO toSave);

    /**
     * 查找所有工资单（根据状态查找时若为null需要用到这里
     */
    List<SalarySheetPO> findAllSheet();

    /**
     * 根据state返回工资单
     * @param state 工资单状态
```

```
     * @return 工资单列表
     */
    List<SalarySheetPO> findAllByState(SalarySheetState state);

    /**
     * 查找指定id的工资单
     * @param id id
     */
    SalarySheetPO findSheetById(String id);

    /**
     * 根据时间返回工资单
     * @param startTime,endTime
     * @return 工资单列表
     */
    List<SalarySheetPO> findAllByTime(Date startTime, Date endTime);

    /**
     * 更新指定工资单的状态
     * @param sheetId 编号
     * @param state 状态
     */
    int updateSheetState(String sheetId, SalarySheetState state);

    /**
     * 制订薪酬规则
     */

    int makeSalaryRule(String sheetId, SalaryRule rule);
```

制订、审批、查看库存赠送单

serviceImpl.saleManagementBusinessImpl.GiftServiceImpl

供接口:

```
    /**
     * 根据单据状态获取库存赠送单
     * @param state
     * @return
     */
    List<GiftSheetPO> getGiftSheetByState(GiftSheetState state);

    /**
     * 审批单据
     * @param giftSheetId
     * @param state
     */
    void approval(String giftSheetId, GiftSheetState state);


    /**
     * 根据销售单Id搜索库存赠送单信息
     * @param giftSheetId Id
     * @return 库存赠送单全部信息
     */
    GiftSheetPO getGiftSheetById(String giftSheetId);
```

需接口：
来自dao.saleDao.GiftSheetDao的：

```
    /**
     *  根据id寻找
     * @param giftSheetId
     * @return
     */
    GiftSheetPO findOneById(String giftSheetId);

    /**
     * 审批
     * @param id
     */
    void updateState(String id, GiftSheetState state);


    /**
     * 根据状态查询所有库存赠送单
     * @return
```

```
     */

    List<GiftSheetPO> findAllByState(GiftSheetState state);
```

制订、审批、查看收款单

serviceImpl.saleManagementBusinessImpl.ReceivableImpl

供接口：

```
     /**
     * 制定收款单
     * @param receivableSheetVO 收款单
     */
    void makeReceivableSheet(UserVO userVO, ReceivableSheetVO receivableSheetVO);

    /**
     * 根据收款单id进行审批(state == "审批完成"/"审批失败")
     * 在controller层进行权限控制
     * @param receivableSheetId 收款单id
     * @param state 收款单修改后的状态
     */
    void approval(String receivableSheetId, ReceivableSheetState state);

    List<ReceivableSheetVO> getAllReceivableSheet();
```

需接口：

来自dao.saleDao.ReceivableSheetDao的

```
    /**
     * 存入一条收款单记录
     * @param toSave 一条收款单记录
     * @return 影响的行数
     */
    int save(ReceivableSheetPO toSave);
    /**
     * 把收款单上转账列表的存入数据库
     * @param receivableSheetContentPO 收款单上的转账列表
     */
    void saveBatch(List<ReceivableSheetContentPO> receivableSheetContentPO);
    /**
     * 获取最近一条收款单
     * @return 最近一条收款单
     */
    ReceivableSheetPO getLatest();


    int updateState(String receivableSheetId, ReceivableSheetState state);


    ReceivableSheetPO findOneById(String receivableSheetId);

    List<ReceivableSheetPO> findAll();
```

```
    List<ReceivableSheetContentPO> findContentBySheetId(String sheetId);
```

制订、审批、查看付款单

serviceImpl.purchaseManagementBusinessImpl.PayableImpl

供接口：

```
    /**
     * 制定付款单
     * @param payableSheetVO 付款单
     */
    void makePayableSheet(UserVO userVO, PayableSheetVO payableSheetVO);

    /**
     * 根据付款单id进行审批(state == "审批完成"/"审批失败")
     * 在controller层进行权限控制
     * @param payableSheetId 付款单id
     * @param state 付款单修改后的状态
     */
    void approval(String payableSheetId, PayableSheetState state);

    List<PayableSheetVO> getAllPayableSheet();
```

需接口：

来自dao.purchaseDao.PayableSheetDao的

```
     /**
     * 存入一条付款单记录
     * @param toSave 一条付款单记录
     * @return 影响的行数
     */
    int save(PayableSheetPO toSave);
    /**
     * 把付款单上条目清单的存入数据库
     * @param payableSheetContentPO 付款单上的条目清单
     */
    void saveBatch(List<PayableSheetContentPO> payableSheetContentPO);
    /**
     * 获取最近一条付款单
     * @return 最近一条付款单
     */
    PayableSheetPO getLatest();


    int updateState(String payableSheetId, PayableSheetState state);


    PayableSheetPO findOneById(String payableSheetId);

    /**
     * 用与选取符合条件的(经营历程列表
     * @param startTime
```

```
     * @param endTime
     * @param supplier
     * @param salesman
     * @return
     */

    List<PayableSheetPO> findPayableSheet(Date startTime, Date endTime, Integer
supplier, String salesman);

    List<PayableSheetPO> findAll();

    List<PayableSheetContentPO> findContentBySheetId(String sheetId);
```

来自dao.overallManagementDao.CompanyBankAccountDao的：

```
void reduceAccountMoney(String bankAccount, BigDecimal totalAmount);
```

来自service.customerManagementBusiness.CustomerServiceImpl的：

```
CustomerPO findCustomerById(String customerId);

void updateCustomer(CustomerPO po);
```

制订薪酬规则，制订、审批、查看工资单

serviceImpl.staffManagementBusinessImpl.SalaryServiceImpl

供接口：

```
    /**
     * 制定工资单
     * @param salarySheetVO 工资单
     */
    void makeSalarySheet(UserVO userVO, SalarySheetVO salarySheetVO);

    /**
     * 根据状态获取工资单(state == null 则获取所有工资单)
     * @param state 工资单状态
     * @return 工资单
     */
    List<SalarySheetVO> getSalarySheetByState(SalarySheetState state);

    /**
     * 根据工资单id进行审批(state == "审批完成"/"审批失败")
     * 在controller层进行权限控制
     * @param salarySheetId 工资单id
     * @param state 工资单修改后的状态
     */
    void approval(String salarySheetId, SalarySheetState state);

    /**
     * 根据工资单Id搜索进货单信息
     * @param salarySheetId 工资单Id
     * @return 工资单全部信息
     */
    SalarySheetVO getSalarySheetById(String salarySheetId);

    /**
     * 改变id为staffId员工的薪酬规则为rule
     * @param staffId 员工Id
     * @param rule 改变后的规则，间SalaryRule枚举类
     */
    void makeSalaryRule(String staffId, SalaryRule rule);
```

需接口：

来自dao.staffDao.SalarySheetDao的

```
    /**
     * 获取最近一条工资单
     */
    SalarySheetPO getLatestSheet();

    /**
```

```
     * 存入一条工资单记录
     * @param toSave 一条工资单记录
     * @return 影响的行数
     */
    int saveSheet(SalarySheetPO toSave);

    /**
     * 查找所有工资单（根据状态查找时若为null需要用到这里
     */
    List<SalarySheetPO> findAllSheet();

    /**
     * 根据state返回工资单
     * @param state 工资单状态
     * @return 工资单列表
     */
    List<SalarySheetPO> findAllByState(SalarySheetState state);

    /**
     * 查找指定id的工资单
     * @param id id
     */
    SalarySheetPO findSheetById(String id);

    /**
     * 根据时间返回工资单
     * @param startTime,endTime
     * @return 工资单列表
     */
    List<SalarySheetPO> findAllByTime(Date startTime, Date endTime);

    /**
     * 更新指定工资单的状态
     * @param sheetId 编号
     * @param state 状态
     */
    int updateSheetState(String sheetId, SalarySheetState state);

    /**
     * 制订薪酬规则
     */

    int makeSalaryRule(String sheetId, SalaryRule rule);
```

员工管理：增加改变查询员工信息

serviceImpl.staffBusinessImpl.StaffAccountServiceImpl

供接口：

```
    /**
     * 登记入职员工信息，系统创建账户
     * @param staffInformationVO
     */
    void creatStaffAccount(StaffInformationVO staffInformationVO);
    /**
     * 修改入职员工信息
     * @param staffInformationVO
     */
    void changeStaffInfo(StaffInformationVO staffInformationVO);
    /**
     * 根据staffId找员工
     * @param staffId
     */
    StaffInformationPO findByStaffId(Integer staffId);

    List<StaffInformationVO> showAllStaffAccount();
```

需接口

来自dao.staffDao.StaffAccountDao的

```
    /**
     * 存入一条员工信息记录
     * @param toSave 一条工资单记录
     * @return 影响的行数
     */
    int saveAccount(StaffInformationPO toSave);

    /**
     * 根据id寻找员工
     * @param staffId
     * @return
     */
    StaffInformationPO findById(Integer staffId);

    /**
     * 更新员工信息
     * @param toChange
     * @return
     */
    int changeInfo(StaffInformationPO toChange);

    List<StaffInfoPO> findAll();
```

销售明细表

service.overallManagementBusiness.SaleStatusServiceImpl

供接口：

```
    /**
     * 根据起始时间，结束时间，商品名，客户，业务员查询
     * @param
     * @return 销售明细列表
     */
    List<SaleStatusSheetVO> getSaleStatusSheet(String startTime, String endTime,
String name, Integer supplier, String salesman);
```

需接口

来自dao.warehouseDao.ProductDao的

```
    ProductPO findById(String id);

    ProductPO findByName(String name);
```

来自dao.saleDao.SaleSheetDao的

```
    /**
     * 用与选取符合条件的(销售明细列表
     * @param startTime
     * @param endTime
     * @param pid
     * @param supplier
     * @param salesman
     * @return
     */
    List<SaleStatusSheetPO> findSheet(Date startTime, Date endTime, String pid,
Integer supplier, String salesman);
```

经营历程表

service.overallManagementBusiness.BusinessProcessServiceImpl

供接口：

```
    /**
     * 根据起始时间，结束时间，客户，业务员查询
     * @param
     * @return 销售单
     */
    List<SaleSheetVO> findSaleSheet(String startTime, String endTime, Integer
supplier, String salesman);

    /**
     * 根据起始时间，结束时间，客户，业务员查询
     * @param
     * @return 销售退货单
     */
    List<SaleReturnsSheetVO> findSaleReturnsSheet(String startTime, String endTime,
Integer supplier, String salesman);

    /**
     * 根据起始时间，结束时间，客户，业务员查询
     * @param
     * @return 进货单
     */
    List<PurchaseSheetVO> findPurchaseSheet(String startTime, String endTime, Integer
supplier, String salesman);

    /**
     * 根据起始时间，结束时间，客户，业务员查询
     * @param
     * @return 进货退货单
     */
    List<PurchaseReturnsSheetVO> findPurchaseReturnsSheet(String startTime, String
endTime, Integer supplier, String salesman);

    /**
     * 根据起始时间，结束时间，客户，业务员查询
     * @param
     * @return 收款单
     */
    List<ReceivableSheetVO> findReceivableSheet(String startTime, String endTime,
Integer supplier, String salesman);

    /**
     * 根据起始时间，结束时间，客户，业务员查询
     * @param
     * @return 付款单
```

```
     */
    List<PayableSheetVO> findPayableSheet(String startTime, String endTime, Integer
supplier, String salesman);

    /**
     * 根据起始时间，结束时间，客户，业务员查询
     * @param
     * @return 工资单
     */
    List<SalarySheetVO> findSalarySheet(String startTime, String endTime, Integer
supplier, String salesman);
```

需接口：

来自dao.saleDao.SaleSheetDao的

```
    /**
     * 用与选取符合条件的(经营历程列表
     * @param startTime
     * @param endTime
     * @param supplier
     * @param salesman
     * @return
     */
    List<SaleSheetPO> findSaleSheet(Date startTime, Date endTime, Integer supplier,
String salesman);

    /**
     * 查找指定销售单下具体的商品内容
     * @param sheetId
     */
    List<SaleSheetContentPO> findContentBySheetId(String sheetId);
```

来自dao.saleDao.SaleReturnsSheetDao的

```
    /**
     * 通过saleReturnsSheetId找到对应的content条目
     * @param saleReturnsSheetId
     * @return
     */
    List<SaleReturnsSheetContentPO> findContentBySaleReturnsSheetId(String
saleReturnsSheetId);

    /**
     * 用与选取符合条件的(经营历程列表
     * @param startTime
     * @param endTime
     * @param supplier
     * @param salesman
```

```
     * @return
     */
    List<SaleReturnsSheetPO> findSaleReturnsSheet(Date startTime, Date endTime,
Integer supplier, String salesman);
```

来自dao.purchaseDao.PurchaseSheetDao的

```
    List<PurchaseSheetContentPO> findContentByPurchaseSheetId(String purchaseSheetId);

    /**
     * 用与选取符合条件的(经营历程列表
     * @param startTime
     * @param endTime
     * @param supplier
     * @param salesman
     * @return
     */
    List<PurchaseSheetPO> findPurchaseSheet(Date startTime, Date endTime, Integer
supplier, String salesman);
```

来自dao.purchaseDao.PurchaseSheetReturnsDao的

```
    /**
     * 用与选取符合条件的(经营历程列表
     * @param startTime
     * @param endTime
     * @param supplier
     * @param salesman
     * @return
     */
    List<PurchaseReturnsSheetPO> findPurchaseReturnsSheet(Date startTime, Date
endTime, Integer supplier, String salesman);

    /**
     * 通过purchaseReturnsSheetId找到对应的content条目
     * @param purchaseReturnsSheetId
     * @return
     */
    List<PurchaseReturnsSheetContentPO> findContentByPurchaseReturnsSheetId(String
purchaseReturnsSheetId);
```

来自dao.purchaseDao.PayableSheetDao的

```
    /**
     * 用与选取符合条件的(经营历程列表
     * @param startTime
     * @param endTime
```

```
     * @param supplier
     * @param salesman
     * @return
     */
    List<PayableSheetPO> findPayableSheet(Date startTime, Date endTime, Integer
supplier, String salesman);

    List<PayableSheetContentPO> findContentBySheetId(String sheetId);


    List<ReceivableSheetContentPO> findContentBySheetId(String sheetId);

    /**
     * 用与选取符合条件的(经营历程列表
     * @param startTime
     * @param endTime
     * @param supplier
     * @param salesman
     * @return
     */
    List<ReceivableSheetPO> findReceivableSheet(Date startTime, Date endTime, Integer
supplier, String salesman);

    List<ReceivableSheetContentPO> findContentBySheetId(String sheetId);
```

来自dao.staffDao.SalarySheetDao的

```
    /**
     * 用与选取符合条件的(经营历程列表
     * @param startTime
     * @param endTime
     * @param staffId
     * @param salesman
     * @return
     */
    List<SalarySheetPO> findSalarySheet(Date startTime, Date endTime, Integer staffId,
String salesman);
```

制订年终奖
serviceImpl.staffBusinessImpl.AnnualBonusServiceImpl

供接口：

```
    /**
     * 设置年终奖
     * @param staffId
     * @param annualAmount
     */
    void setAnnualAmount(String staffId, BigDecimal annualAmount);

    /**
     * 根据员工id查找年终奖
     * @param staffId
     * @return
     */
    BigDecimal findAnnualAmountById(String staffId);
```

需接口：
来自dao.staffDao.AnnualBonusDao的
save(AnnualBonusPO po);
getAmount(Sting staffId)

经营情况表

service.overallManagementBusiness.BusinessConditionServiceImpl

供接口：

```
    /**
     * 根据起始时间，结束时间
     * @param
     * @return 经营情况
     */
    BusinessConditionSheetVO getBusinessConditionSheet(String startTime, String
endTime);
```

需接口：

来自dao.saleDao.SaleSheetDao的

```
    /**
     * 根据时间返回销售单
     * @param startTime,endTime
     * @return 销售列表
     */
    List<SaleSheetPO> findAllByTime(Date startTime, Date endTime);
```

来自dao.staffDao.SalarySheetDao的

```
    /**
     * 根据时间返回工资单
     * @param startTime,endTime
     * @return 工资单列表
     */
    List<SalarySheetPO> findAllByTime(Date startTime, Date endTime);
```

来自dao.purchaseDao.PurchaseSheetDao的

```
    /**
     * 根据时间返回进货单
     * @param startTime,endTime
     * @return 进货列表
     */
    List<PurchaseSheetPO> findAllByTime(Date startTime, Date endTime);
```

使用销售策略
service.Impl.salesBusinessImpl.promotionStrategy
需接口:
来自dao.saleDao.PromotionDao的

```
 /**
     * 查找符合条件的最大的代金券
     * @param level 客户等级
     * @param date 使用时间
     * @return
     */
    PromotionPO findVoucherDESCByTimeAndLevel(int level,Date date);

    /**
     * 查找符合条件的最大的代金券
     * @param amount 总金额
     * @param date 使用时间
     * @return
     */
    PromotionPO findVoucherDESCByTimeAndAmount(BigDecimal amount, Date date);

    /**
     * 查找符合条件的商品组合降价策略
     * @param date 使用时间
     * @param sheetId sale sheet id
     * @return
     */
    List<PromotionPO> findCompositionDiscountASC(Date date,String sheetId);

    /**
     * 查找符合条件的最大折扣
     * @param level 客户等级
     * @param date 使用时间
     * @return
     */
    PromotionPO findDiscountASCByTimeAndLevel(int level,Date date);

    /**
     * 寻找符合条件的赠品策略,并选取对等级要求最严的一个
     * @param level 客户等级
     * @param date 时间
     * @return
     */
    PromotionPO findGiftByTimeAndLevel(int level,Date date);

    /**
     * 寻找符合条件的赠品策略，并选取对总金额要求最严格的一个
     * @param amount 总金额
     * @param date 时间
     * @return
```

```
     */
    PromotionPO findGiftByTimeAndAmount(BigDecimal amount,Date date);
```

6.5 数据层分解

负责接受业务逻辑层的调用，操作数据库

具体职责&接口规范如下

账户管理：
dao.overallManagementDao.CompanyBankAccountDao

供接口：

```
    /**
     * 存入新增加的公司账户
     * @param toSave
     * @return 影响的行数
     */
    int saveAccount(CompanyBankAccountPO toSave);
    /**
     * 删除公司账户
     * @param companyBankAccountName
     * @return 影响的行数
     */
    int deleteAccount(String companyBankAccountName);
    /**
     * 修改公司账户属性（名称）
     * @param id, companyBankAccountName
     */
    int changeAccountName(String id, String companyBankAccountName);
    /**
     * 减少公司余额
     * @param id
     */
    int reduceAccountMoney(String id, BigDecimal amount);
    /**
     * 根据账户名模糊查找
     * @param companyBankAccountName 公司账户名
     * @return 模糊查找得到的账户
     */
    List<CompanyBankAccountPO> findAllByName(String companyBankAccountName);

    List<CompanyBankAccountPO> findAll();
```

制订、审批、查看付款单
dao.purchaseDao.PayableSheetDao

供接口：

```
     /**
     * 存入一条付款单记录
     * @param toSave 一条付款单记录
     * @return 影响的行数
     */
    int save(PayableSheetPO toSave);
    /**
     * 把付款单上条目清单的存入数据库
     * @param payableSheetContentPO 付款单上的条目清单
     */
    void saveBatch(List<PayableSheetContentPO> payableSheetContentPO);
    /**
     * 获取最近一条付款单
     * @return 最近一条付款单
     */
    PayableSheetPO getLatest();


    int updateState(String payableSheetId, PayableSheetState state);


    PayableSheetPO findOneById(String payableSheetId);

    /**
     * 用与选取符合条件的(经营历程列表
     * @param startTime
     * @param endTime
     * @param supplier
     * @param salesman
     * @return
     */
    List<PayableSheetPO> findPayableSheet(Date startTime, Date endTime, Integer
supplier, String salesman);

    List<PayableSheetPO> findAll();

    List<PayableSheetContentPO> findContentBySheetId(String sheetId);
```

制订销售策略
dao.saleDao.PromotionDao

供接口：

```
    /**
     * 查找符合条件的最大的代金券
     * @param level 客户等级
     * @param date 使用时间
     * @return
     */
    PromotionPO findVoucherDESCByTimeAndLevel(int level,Date date);

    /**
     * 查找符合条件的最大的代金券
     * @param amount 总金额
     * @param date 使用时间
     * @return
     */
    PromotionPO findVoucherDESCByTimeAndAmount(BigDecimal amount, Date date);

    /**
     * 查找符合条件的商品组合降价策略
     * @param date 使用时间
     * @param sheetId sale sheet id
     * @return
     */
    List<PromotionPO> findCompositionDiscountASC(Date date,String sheetId);

    /**
     * 查找符合条件的最大折扣
     * @param level 客户等级
     * @param date 使用时间
     * @return
     */
    PromotionPO findDiscountASCByTimeAndLevel(int level,Date date);

    /**
     * 寻找符合条件的赠品策略,并选取对等级要求最严的一个
     * @param level 客户等级
     * @param date 时间
     * @return
     */
    PromotionPO findGiftByTimeAndLevel(int level,Date date);

    /**
     * 寻找符合条件的赠品策略，并选取对总金额要求最严格的一个
     * @param amount 总金额
     * @param date 时间
     * @return
     */
```

```
    PromotionPO findGiftByTimeAndAmount(BigDecimal amount,Date date);

    /**
     * 建立一条代金券策略
     * @param po
     */
    void addVoucher(PromotionPO po);

    /**
     * 建立一条折扣策略
     * @param po
     */
    void addDiscount(PromotionPO po);

    /**
     * 建立一条赠品策略
     */
    void addGift(PromotionPO po);
```

制订、审批、查看库存赠送单
dao.saleDao.GiftSheetDao

供接口：

```
    /**
     * 获取最近一条库存赠送单
     */
    GiftSheetPO getLatest();

    /**
     *  根据id寻找
     * @param giftSheetId
     * @return
     */
    GiftSheetPO findOneById(String giftSheetId);

    BigDecimal findPriceById(String id);

    /**
     * 添加一个库存赠送单
     * @param giftSheetPO
     */
    void addSheet(GiftSheetPO giftSheetPO);

    /**
     * 审批
     * @param id
     */
    void updateState(String id, GiftSheetState state);


    /**
     * 查询所有库存赠送单
     * @return
     */
    List<GiftSheetPO> findAll();

    List<GiftSheetPO> findAllByState(GiftSheetState state);
```

制订、审批、查看收款单
dao.saleDao.ReceivableDao

供接口：

```
 /**
     * 存入一条收款单记录
     * @param toSave 一条收款单记录
     * @return 影响的行数
     */
    int save(ReceivableSheetPO toSave);
    /**
     * 把收款单上转账列表的存入数据库
     * @param receivableSheetContentPO 收款单上的转账列表
     */
    void saveBatch(List<ReceivableSheetContentPO> receivableSheetContentPO);
    /**
     * 获取最近一条收款单
     * @return 最近一条收款单
     */
    ReceivableSheetPO getLatest();


    int updateState(String receivableSheetId, ReceivableSheetState state);


    ReceivableSheetPO findOneById(String receivableSheetId);

    /**
     * 用与选取符合条件的(经营历程列表
     * @param startTime
     * @param endTime
     * @param supplier
     * @param salesman
     * @return
     */
    List<ReceivableSheetPO> findReceivableSheet(Date startTime, Date endTime, Integer
supplier, String salesman);

    List<ReceivableSheetPO> findAll();

    List<ReceivableSheetContentPO> findContentBySheetId(String sheetId);
```

员工打卡
dao.staffDao.DailyAttendanceDao
供接口：

```
/**
     *
     * @param staffId 员工id
     * @param beginDate 开始日期（某天）
     * @param endDate 结束日期（某天）
     * @return
     */
    AttendancePO doesThisDayHeAttend(String staffId, Date beginDate, Date endDate);

    /**
     *
     * @param beginDate 开始日期（某月）
     * @param endDate 结束日期（某月）
     * @return
     */
    List<AttendancePO> findAll(Date beginDate,Date endDate);

    /**
     *查询:
     * @param beginDate  开始日期（某月）
     * @param endDate 结束日期（某月）
     * @param staffId 员工id
     * @return
     */
    List<AttendancePO> findOne(Date beginDate,Date endDate,String staffId);

    /**
     * 签到用
     * @param attendancePO
     */
    void AttendanceRegister(AttendancePO attendancePO);

    String findIdByName(String name);
```

员工管理
dao.staffDao.StaffAccountDao

供接口：

```
    /**
     * 存入一条员工信息记录
     * @param toSave 一条工资单记录
     * @return 影响的行数
     */
    int saveAccount(StaffInformationPO toSave);

    /**
     * 根据id寻找员工
     * @param staffId
     * @return
     */
    StaffInformationPO findById(Integer staffId);

    /**
     * 更新员工信息
     * @param toChange
     * @return
     */
    int changeInfo(StaffInformationPO toChange);

    List<StaffInformationPO> findAll();
```

薪酬规则制订，制订、审批、查看工资单
dao.staffDao.SalarySheetDao

供接口：

```
    /**
     * 获取最近一条工资单
     */
    SalarySheetPO getLatestSheet();

    /**
     * 存入一条工资单记录
     * @param toSave 一条工资单记录
     * @return 影响的行数
     */
    int saveSheet(SalarySheetPO toSave);

    /**
     * 查找所有工资单（根据状态查找时若为null需要用到这里
     */
    List<SalarySheetPO> findAllSheet();

    /**
     * 根据state返回工资单
     * @param state 工资单状态
     * @return 工资单列表
     */
    List<SalarySheetPO> findAllByState(SalarySheetState state);

    /**
     * 查找指定id的工资单
     * @param id id
     */
    SalarySheetPO findSheetById(String id);

    /**
     * 根据时间返回工资单
     * @param startTime,endTime
     * @return 工资单列表
     */
    List<SalarySheetPO> findAllByTime(Date startTime, Date endTime);

    /**
     * 更新指定工资单的状态
     * @param sheetId 编号
     * @param state 状态
     */
    int updateSheetState(String sheetId, SalarySheetState state);

    /**
```

```
     * 制订薪酬规则
     */

    int makeSalaryRule(String sheetId, SalaryRule rule);

    /**
     * 用与选取符合条件的(经营历程列表
     * @param startTime
     * @param endTime
     * @param staffId
     * @param salesman
     * @return
     */
    List<SalarySheetPO> findSalarySheet(Date startTime, Date endTime, Integer staffId,
String salesman);
```

制订年终奖
dao.staffDao.AnnualBonus
供接口：

```
     /**
     * 设置年终奖
     * @param annualBonusPO
     * @return 影响的行数
     */
    int save(AnnualBonusPO annualBonusPO);

    BigDecimal getAmount(String staffId);
```