**基于SSM框架的账单报销系统**

# 功能需求

功能模块:

1. 部门信息管理
2. 员工信息管理
3. 报销单处理

主要角色:

1. 员工
2. 部门经理
3. 总经理
4. 财务

员工功能:

登录,注册,填写并保存报销单,提交报销单,修改报销单,查看报销单,查询自己的报销单

部门经理:

查询部门经理待审核报销单

部门经理审核报销单

总经理:

查询总经理待审核报销单

总经理审核报销单

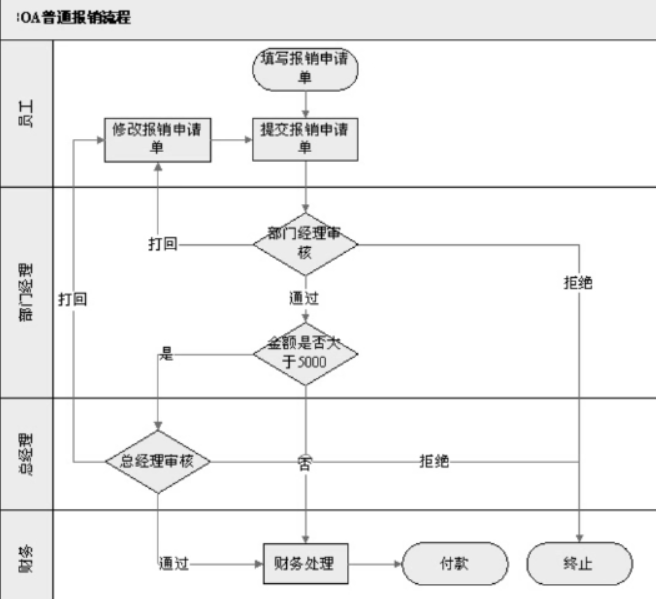
财务:

查询财务待处理报销单

财务处理报销单

报销单处理流程:

填写->提交->审核->打款



# 主要技术

SpringIOC Spring+Mybatis整合 声明式事务 Spring标签库 Spring拦截器

# 项目结构

## 三层架构

- 持久层--Mybatis

- 表现层--SpringMVC

- 业务层--JavaBean

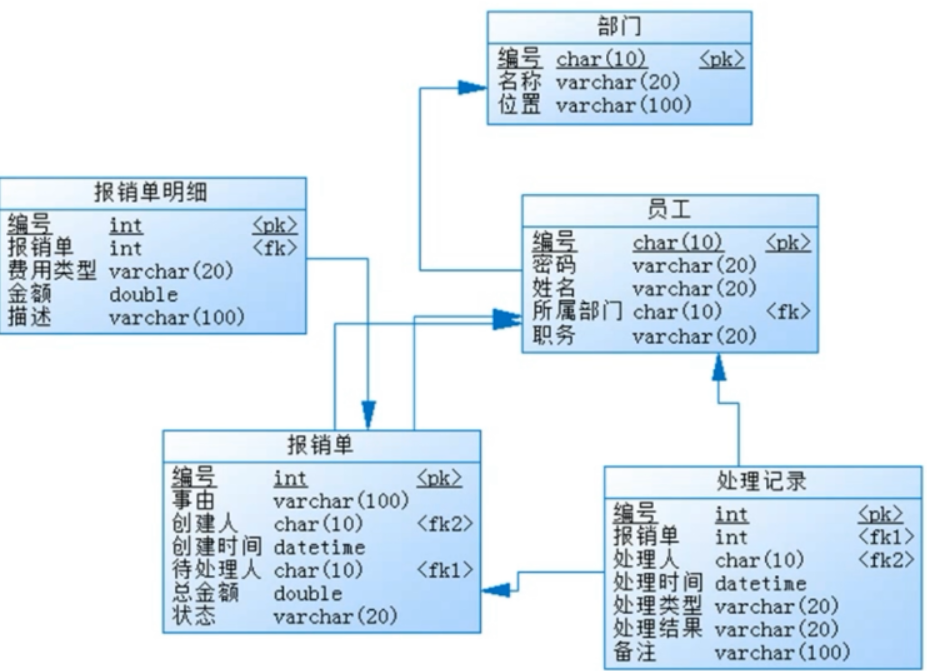
## 基于MVC模式

视图--Jsp

模型--JavaBean

控制器--SpringController

# 数据库的设计



|  |
| --- |
| drop database if exists leonard;  create database leonard;  use leonard;  /\*==============================================================\*/  /\* Table: claim\_voucher \*/  /\*==============================================================\*/  create table claim\_voucher  (  id int not null auto\_increment,  cause varchar(100),  create\_sn char(5),  create\_time datetime,  next\_deal\_sn char(5),  total\_amount double,  status varchar(20),  primary key (id)  );  /\*==============================================================\*/  /\* Table: claim\_voucher\_item \*/  /\*==============================================================\*/  create table claim\_voucher\_item  (  id int not null auto\_increment,  claim\_voucher\_id int,  item varchar(20),  amount double,  comment varchar(100),  primary key (id)  );  /\*==============================================================\*/  /\* Table: deal\_record \*/  /\*==============================================================\*/  create table deal\_record  (  id int not null auto\_increment,  claim\_voucher\_id int,  deal\_sn char(5),  deal\_time datetime,  deal\_way varchar(20),  deal\_result varchar(20),  comment varchar(100),  primary key (id)  );  /\*==============================================================\*/  /\* Table: department \*/  /\*==============================================================\*/  create table department  (  sn char(5) not null,  name varchar(20),  address varchar(100),  primary key (sn)  );  /\*==============================================================\*/  /\* Table: employee \*/  /\*==============================================================\*/  create table employee  (  sn char(5) not null,  password varchar(20),  name varchar(20),  department\_sn char(5),  post varchar(20),  primary key (sn)  );  alter table claim\_voucher add constraint FK\_Reference\_2 foreign key (next\_deal\_sn)  references employee (sn) on delete restrict on update restrict;  alter table claim\_voucher add constraint FK\_Reference\_3 foreign key (create\_sn)  references employee (sn) on delete restrict on update restrict;  alter table claim\_voucher\_item add constraint FK\_Reference\_4 foreign key (claim\_voucher\_id)  references claim\_voucher (id) on delete restrict on update restrict;  alter table deal\_record add constraint FK\_Reference\_5 foreign key (claim\_voucher\_id)  references claim\_voucher (id) on delete restrict on update restrict;  alter table deal\_record add constraint FK\_Reference\_6 foreign key (deal\_sn)  references employee (sn) on delete restrict on update restrict;  alter table employee add constraint FK\_Reference\_1 foreign key (department\_sn)  references department (sn) on delete restrict on update restrict;  insert into department values('10001','总经理办公室','星星大厦E座1201');  insert into department values('10002','财务部','星星大厦E座1202');  insert into department values('10003','事业部','星星大厦E座1101');  insert into employee values('10001','000000','刘备','10001','总经理');  insert into employee values('10002','000000','孙尚香','10002','财务');  insert into employee values('10003','000000','关羽','10003','部门经理');  insert into employee values('10004','000000','周仓','10003','员工'); |

# 五.项目构建与maven的依赖配置

采用idea构建maven项目,步骤:

Create new project->选择maven->不勾选(不选择模板类型)

创建父module和子module

## Leonard-父module-全局定义组织模块

## Leonard\_dao-持久层

需要依赖:mybatis依赖,spring依赖,m-s整合依赖

## Leonard\_biz-业务层

需要依赖:Aspectj依赖,AOP依赖

## Leonard\_web-表现层

Servlet依赖,SpringMVC依赖

目录命名:

Global 组织成数据字典,有很多业务名称

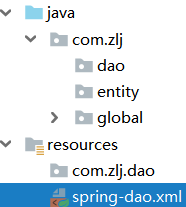
Entity:表与Java中的对象语言程序中的对象

Dto:Java中的对象数据与页面相关联

# 包与全局配置

## 1.Zlj\_dao

- 创建目录:dao,entity,global

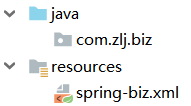


- 配置文件(spring-dao.xml):数据源,Session工厂,映射器接口

|  |
| --- |
| <**beans** *xmlns***="http://www.springframework.org/schema/beans"** *xmlns:xsi***="http://www.w3.org/2001/XMLSchema-instance"** *xmlns:context***="http://www.springframework.org/schema/context"** *xsi:schemaLocation***="http://www.springframework.org/schema/beans  http://www.springframework.org/schema/beans/spring-beans.xsd  http://www.springframework.org/schema/context  http://www.springframework.org/schema/context/spring-context.xsd http://www.springframework.org/schema/context "**>    <!--开启自动扫描-->  <**context:component-scan** *base-package***="com.zlj.dao"**/>   <!--配置数据源-->  <**bean** *id***="dataSource"** *class***="org.springframework.jdbc.datasource.DriverManagerDataSource"**>  <**property** *name***="driverClassName"** *value***="com.mysql.jdbc.Driver"**/>  <**property** *name***="url"** *value***="jdbc:mysql://localhost:3306/leonard?useUnicode=true&amp;characterEncoding=utf-8"**/>  <**property** *name***="username"** *value***="root"**/>  <**property** *name***="password"** *value***="root"**/>  </**bean**>   <!--配置session工厂-->  <**bean** *id***="sessionFactory"** *class***="org.mybatis.spring.SqlSessionFactoryBean"**>  <**property** *name***="dataSource"** *ref***="dataSource"**/>  <**property** *name***="typeAliasesPackage"** *value***="com.zlj.entity"**/>  </**bean**>   <!--映射器接口:自动调用-->  <**bean** *class***="org.mybatis.spring.mapper.MapperScannerConfigurer"**>  <**property** *name***="sqlSessionFactoryBeanName"** *value***="sessionFactory"**/>  <!--将这个包对应的所有接口与映射文件对应起来-->  <**property** *name***="basePackage"** *value***="com.zlj.dao"**/>  </**bean**> </**beans**> |

## Zlj\_biz

-创建目录:biz



-配置文件(spring-biz.xml):事务

|  |
| --- |
| *<?xml version***="1.0"** *encoding***="UTF-8"***?>* <**project** *xmlns***="http://maven.apache.org/POM/4.0.0"** *xmlns:xsi***="http://www.w3.org/2001/XMLSchema-instance"** *xsi:schemaLocation***="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd"**>  <**parent**>  <**artifactId**>leonard</**artifactId**>  <**groupId**>com.zlj</**groupId**>  <**version**>1.0-SNAPSHOT</**version**>  </**parent**>  <**modelVersion**>4.0.0</**modelVersion**>   <**artifactId**>zlj\_biz</**artifactId**>   <**dependencies**>   <!--业务层依赖持久层-->  <**dependency**>  <**groupId**>com.zlj</**groupId**>  <**artifactId**>zlj\_dao</**artifactId**>  <**version**>1.0-SNAPSHOT</**version**>  </**dependency**>    <!--为JDBC、Hibernate、JDO、JPA等提供的一致的声明式和编程式事务管理 -->  <**dependency**>  <**groupId**>org.springframework</**groupId**>  <**artifactId**>spring-tx</**artifactId**>  <**version**>${spring.version}</**version**>  </**dependency**>    <!--提供AOP(面向切面编程)实现 -->  <**dependency**>  <**groupId**>org.springframework</**groupId**>  <**artifactId**>spring-aop</**artifactId**>  <**version**>${spring.version}</**version**>  </**dependency**>   <!--提供对AspectJ的支持，以便可以方便的将面向切面的功能集成进IDE中-->  <**dependency**>  <**groupId**>org.aspectj</**groupId**>  <**artifactId**>aspectjweaver</**artifactId**>  <**version**>1.8.0</**version**>  </**dependency**>  </**dependencies**> </**project**> |

## Zlj\_web

-创建目录:controller,dto(与entity的区别),global

-配置文件(spring-web.xml):

|  |
| --- |
| <**beans** *xmlns***="http://www.springframework.org/schema/beans"** *xmlns:xsi***="http://www.w3.org/2001/XMLSchema-instance"** *xmlns:context***="http://www.springframework.org/schema/context"** *xmlns:mvc***="http://www.springframework.org/schema/mvc"** *xsi:schemaLocation***="http://www.springframework.org/schema/beans  http://www.springframework.org/schema/beans/spring-beans.xsd  http://www.springframework.org/schema/context  http://www.springframework.org/schema/context/spring-context.xsd  http://www.springframework.org/schema/mvc  http://www.springframework.org/schema/mvc/spring-mvc.xsd"**>   <**import** *resource***="spring-biz.xml"**/>   <!--开启自动扫描和mvc自己的注解驱动-->  <**context:component-scan** *base-package***="com.zlj.controller"**/>  <**mvc:annotation-driven**/>   <!--静态资源设置规则:静态资源交给servlet处理,没必要交给springmvc-->  <**mvc:default-servlet-handler**/>   <!--视图转换器-->  <**bean** *class***="org.springframework.web.servlet.view.InternalResourceViewResolver"**>  <**property** *name***="viewClass"** *value***="org.springframework.web.servlet.view.JstlView"**/>  <**property** *name***="prefix"** *value***="/WEB-INF/pages"**/>  <**property** *name***="suffix"** *value***=".jsp"**/>  </**bean**>   <!--配置登录的拦截器-->  <**mvc:interceptors**>  <**mvc:interceptor**>  <**mvc:mapping** *path***="/\*\*"**/>  <**bean** *class***="com.zlj.global.LoginInterceptor"**/>  </**mvc:interceptor**>  </**mvc:interceptors**>  </**beans**> |

-配置web.xml加载springmvc

|  |
| --- |
| *<?xml version***="1.0"** *encoding***="UTF-8"***?>* <**web-app** *xmlns***="http://xmlns.jcp.org/xml/ns/javaee"** *xmlns:xsi***="http://www.w3.org/2001/XMLSchema-instance"** *xsi:schemaLocation***="http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-app\_3\_1.xsd"** *version***="3.1"**>    <**filter**>  <**filter-name**>encoding</**filter-name**>  <**filter-class**>com.zlj.global.EncodingFilter</**filter-class**>  <**init-param**>  <**param-name**>encoding</**param-name**>  <**param-value**>utf-8</**param-value**>  </**init-param**>  </**filter**>   <**filter-mapping**>  <**filter-name**>encoding</**filter-name**>  <**url-pattern**>/\*</**url-pattern**>  </**filter-mapping**>    <**servlet-mapping**>  <**servlet-name**>default</**servlet-name**>  <**url-pattern**>/assets/\*</**url-pattern**>  <**url-pattern**>/js/\*</**url-pattern**>  <**url-pattern**>/vendor/\*</**url-pattern**>  <**url-pattern**>\*.js</**url-pattern**>  <**url-pattern**>\*.jpg</**url-pattern**>  <**url-pattern**>\*.gif</**url-pattern**>  <**url-pattern**>\*.png</**url-pattern**>  <**url-pattern**>\*.css</**url-pattern**>  </**servlet-mapping**>   <**servlet**>  <**servlet-name**>SpringMVC</**servlet-name**>  <**servlet-class**>org.springframework.web.servlet.DispatcherServlet</**servlet-class**>  <**init-param**>  <**param-name**>contextConfigLocation</**param-name**>  <**param-value**>classpath:spring-web.xml</**param-value**>  </**init-param**>  <**load-on-startup**>1</**load-on-startup**>  </**servlet**>    <**servlet-mapping**>  <**servlet-name**>SpringMVC</**servlet-name**>  <**url-pattern**>/</**url-pattern**>  </**servlet-mapping**>  </**web-app**> |

# 部门管理模块

实现功能:部门信息的crud

步骤:Dao->biz->web

-实体类

-dao接口与sql映射文件

-biz接口与其实现类

-控制器

-页面

# 员工管理模块

和部门管理模块一样实现基本的crud

但是主要关注一下几点:

1. 员工默认密码
2. 员工与部门的相互关系

# 登录以及个人中心

登录,退出,个人信息,修改密码

关注点

-Session 操作

-登录拦截器

# 报销单处理

填写报销单

流程:

-保存报销单及条目信息

关注点:

(Dto:Java中的对象数据与页面相关联)

-dto---报销单信息

-不定项表单处理

个人报销单,待处理报销单

修改报销单

(1)更新基本信息

(2)更新条目信息的三种情况:

1. 删除准备不要的条目
2. 修改已经有的条目
3. 添加插入之前没有的新条目



提交报销单

流程:

1. 修改状态
2. 记录处理流程

关注点:

部门经理的获取

审核报销单

流程:

1. 修改状态
2. 记录处理流程

关注点:

不同操作的结果

是否需要复审

打款

前端原型

|  |
| --- |
| <jsp:include page="top.jsp"/>  <!-- Page Header-->  <header class="page-header">  <div class="container-fluid">  <h2 class="no-margin-bottom">xx</h2>  </div>  </header>  <!-- Breadcrumb-->  <div class="breadcrumb-holder container-fluid">  <ul class="breadcrumb">  <li class="breadcrumb-item"><a href="#">Home</a></li>  <li class="breadcrumb-item active">xx</li>  </ul>  </div>  <!--Sections-->  <jsp:include page="bottom.jsp"/> |

表单原型

|  |
| --- |
| <!--Forms Sections-->  <section class="forms">  <div class="container-fluid">  <div class="row">  <!-- Basic Form-->  <div class="col-lg-6">  <div class="card">  <div class="card-close">  <div class="dropdown">  <button type="button" id="closeCard1" data-toggle="dropdown" aria-haspopup="true" aria-expanded="false" class="dropdown-toggle"><i class="fa fa-ellipsis-v"></i></button>  <div aria-labelledby="closeCard1" class="dropdown-menu dropdown-menu-right has-shadow"><a href="#" class="dropdown-item remove"> <i class="fa fa-times"></i>Close</a><a href="#" class="dropdown-item edit"> <i class="fa fa-gear"></i>Edit</a></div>  </div>  </div>  <div class="card-header d-flex align-items-center">  <h3 class="h4">Basic Form</h3>  </div>  <div class="card-body">  <p>(#^.^#)</p>  <form method="post" action="modify\_password">  <div class="form-group">  <label class="form-control-label">原始密码</label>  <input type="password" name="old" id="old" placeholder="Password" class="form-control">  </div>  <div class="form-group">  <button type="submit" class="btn btn-primary">确认修改</button>  </div>  </form>  </div>  </div>  </div>  </div>  </div>  </section> |

解决js缓存问题:items.js?v=1

<button type="button" class="button" onclick="javascript:window.history.go(-1);"> 返回 </button>