**《J2EE企业应用开发项目实践说明书》**

1. **概述**

**1.1开发环境及安装配置**

服务器硬件：

目前PC服务器基本都能满足本系统的需求，要求：

内存：不少于2G

CPU：Intel 2.8以上

硬盘：RAID，720G

软件环境

服务器软件

MS Windows XP

MyEclipse 8.5

SQLServler2012

Tomcat 6.0.29

DreamWaver 2004

StartUML-5.0

开发平台

MyEclipse 8.5

SQLServler2012

**1.2项目的需求分析**

实现提高高校宿舍管理部门的工作效率；在充分利用资源；减少不必要的人力、物力和财力的支出；方便宿舍管理部门的工作人员全面的掌握学生住宿情况等目的。

用户登录：对用户身份进行验证，用户根据权限操作响应的工能模块；

用户管理：用户分为超级管理员和宿管员两种类型，超级管理员可以添加、删除、修改、查询；宿管员有修改、查询、删除的功能。

权限设置：权限规定了用户操作某项工能的限制，其包含用户是否可以操作某个页面控件，权限可以浏览，新增，编辑和删除；

系统退出：登出，离开后台。

1. **程序概要设计**

**2.1程序的功能模块**

此宿舍管理系统是C/S结构，总体可以分五个个功能模块。但组要的有三个功能模块，可以对楼宇信息的管理、宿管信息的管理和学生信息的管理，系统都可以对各项信息进行增删改查的操作，具体结构图如下：

楼信息添加

查询数据

楼信息修改

删除

楼信息

楼宇信息的管理

宿管员信息的管理

学生信息的管理

宿管员信息添加

宿管员信息查询

宿管员信息修改

宿管员信息删除

学生信息添加

学生信息查询

学生信息修改

学生信息删除

宿舍信息管理系统

在整体设计中，除了宿舍管理系统以上三个主要功能模块以外，还有连个辅佐模块，分别是：信息登录模块和宿舍信息查询模块，每个模块将实现不同的功能。

如信息登录模块会对用户分为三个等级，分别是：超级管理员，宿管员和学生。超级管理员账号无需注册，直接得到，能对宿舍管理系统的所有信息进行统一管理，还可以添加宿管员和学生信息，使得他们得到账号登录，用例图如下，



宿舍信息查询就能查询宿舍的所有信息，包括管理的宿管员及宿管电话。

**2.2程序的流程**

流程图如下所示，



**2.3程序的文件结构分析**

程序文件根目录下有四个文件夹，五个文件，其中Webroot文件下有文件夹有css、image、META-INF和WEB-INF文件，以及所有的jsp文件，src文件夹下有com文件夹及struts.properties和struts.xml文件，src文件夹下放有action、dao、model、server、tool几个文件夹。

**2.3数据库的设计**

1. Super\_login(超级管理员用户表)

表3-12超级管理员用户表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 中文名 | 字段名 | 字段类型 | 长度 | 是否为空 | 主键 | 描述 |
| 超管id | Sid | int | 10 | 否 | 是 | 自动增长 |
| 工作编号 | Ad\_id | int | 10 | 否 | 是 |  |
| 密码 | Spassword | verchar | 20 | 否 | 否 |  |

1. Dorm\_login（宿舍管理员用户表）

表3-13宿舍管理员用户表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 中文名 | 字段名 | 字段类型 | 长度 | 是否为空 | 主键 | 描述 |
| 宿管id | Did | int | 10 | 否 | 是 | 自动增长 |
| 工作编号 | Ad\_id | int | 10 | 否 | 是 |  |
| 密码 | Dpassword | verchar | 20 | 否 | 否 |  |

1. Building（楼宇信息表）

表3-14楼宇信息表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 中文名 | 字段名 | 字段类型 | 长度 | 是否空 | 主键 | 描述 |
| 楼号 | BId | int | 10 | 否 | 是 | 自动增长 |
| 性别 | Sex | verchar | 4 | 否 | 否 |  |
| 标准 | Lid | int | 5 | 否 | 是 |  |

4) Dormitory（宿舍信息表）

表3-15宿舍信息表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 中文名 | 字段名 | 字段类型 | 长度 | 是否空 | 主键 | 描述 |
| 房间号 | Id | int | 10 | 否 | 是 |  |
| 楼号 | Bid | int | 10 | 否 | 是 |  |
| 工作编号 | Ad\_id | int | 10 |  | 是 |  |

5)Student(学生信息表)

表3-16学生信息表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 中文名 | 字段名 | 字段类型 | 长度 | 是否空 | 主键 | 描述 |
| 学号 | Stu\_id | int | 10 | 否 | 是 | 自动增长 |
| 房间号 | Id | int | 10 | 否 | 是 |  |
| 姓名 | Name | verchar | 10 |  |  |  |
| 性别 | Sex | verchar | 4 | 否 |  |  |
| 院系 | Department | verchar | 20 | 否 |  |  |
| 专业 | Major | verchar | 20 | 否 |  |  |
| 宿舍号 | Bid | int |  | 否 |  |  |
| 入学时间 | Time | datetime | date | 否 |  |  |
| 联系方式 | Call | verchar | 20 | 否 |  |  |

6）AdminMessage（宿舍管理员信息表）

表3-17宿舍管理员信息表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 中文名 | 字段名 | 字段类型 | 长度 | 是否空 | 主键 | 描述 |
| 工作编号 | Ad\_id | int | 10 | 否 | 是 | 自动增长 |
| 管理楼号 | BId | int | 10 | 否 | 是 |  |
| 姓名 | Name | verchar | 10 | 否 | 否 |  |
| 用户级别 | Userlevel | int | 5 |  | 否 |  |
| 联系方式 | Call | verchar | 20 |  | 否 |  |

1. BuildingLevel（楼宇标准）

表3-18楼宇标准

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 中文名 | 字段名 | 字段类型 | 长度 | 是否空 | 主键 | 描述 |
| 标准编号号 | Lid | int | 5 | 否 | 是 | 自动增长 |
| 住人数 | Number | int | 5 | 否 |  |  |
| 收费 | Tip | int | 10 | 否 |  |  |

1. **程序详细设计**

系统的功能是主要是增删改查操作，其中添加能是jsp界面得到数据以后，将其提交到action文件进行处理，如BulidingAction.文件大妈如下，

public String add() {

superman = (SuperLogin) ActionContext.getContext().getSession().get(

"loginuser");

if (building.getBid() == 0) {

bmesg = "楼号不能为空！";

return INPUT;

}

if (building.getSex() == null || building.getSex() == "") {

smesg = "楼宇性别标识不能为空！";

return INPUT;

}

if (buildinglevel.getNumber() == 0) {

nmesg = "楼宇容纳人数不能为空！";

return INPUT;

}

if (buildinglevel.getTip() == 0) {

tmesg = "楼宇收费标准不能为空！";

return INPUT;

}

if (superman != null) {

boolean flag = false;

System.out.println(building.getLid()+" "+buildinglevel.getLid());

if (!building.getLid().equals(buildinglevel.getLid())) {

addmesg = "您输入的楼宇标准与标准信息中的标准号不一致！";

return INPUT;

}

flag = buildingservice.add(building, buildinglevel);

if (flag) {

ActionContext.getContext().getSession().put("building",

building);

ActionContext.getContext().getSession().put("buildinglevel",

buildinglevel);

System.out.println("楼宇功能实现！");

return SUCCESS;

} else {

return INPUT;

}

} else {

loginmesg = "请登录后进行操作！";

return LOGIN;

}

}

Action先把信息封装成一个类，add（）方法再通过if语句判断所得到的信息是否为空，有为空则返回INPUT，重新在jsp界面上输入，确认信息无误后则调用buildingservice的add()方法，并且把所得到的类保存到session中。程序的service是通过动态代理方式实现，buildingservice只是一个接口，真正是方法在buildingservice.impl类中，具体add（）方法如下，

public boolean add(Building building, Buildinglevel buildinglevel) {

boolean flag=false;

Buildinglevel findbuildinglevel=new Buildinglevel();

findbuildinglevel=buildingleveldao.findById(buildinglevel.getLid());

if(findbuildinglevel!=null){

System.out.println(buildinglevel.getLid()+"楼标准已经存在");

flag=true;

}else{

flag=buildingleveldao.save(buildinglevel);

}

if(flag){

Building findbuilding=new Building();

findbuilding=buildingdao.findById(building.getBid());

if(findbuilding!=null){

System.out.println(building.getLid()+"楼房已经存在");

return true;

}else{

return buildingdao.save(building);

}

}else{

return false;

}

}

此方法先通过buildingleveldao.findById（）方法查找楼收费表是否存在，若存在则直接储存楼号及性别信息，若不存在则再储存楼收费信息。其中buildingdao.findById()方法代码如下，

public Building findById(java.lang.Integer id) {

log.debug("getting Building instance with id: " + id);

try {

Building instance = (Building) getHibernateTemplate().get(

"com.model.Building", id);

return instance;

} catch (RuntimeException re) {

log.error("get failed", re);

throw re;

}

}

此方法组要调用的是getHibernateTemplate()类的get("com.model.Building", id)方法，返回一个building类。另外buildingleveldao类的save（）方法是用于调用Hiberanate框架去实现保存信息，代码如下，

public boolean save(Building transientInstance) {

log.debug("saving Building instance");

try {

getHibernateTemplate().save(transientInstance);

log.debug("save successful");

System.out.println("BuildingDAOImpl楼宇添加功能成功实现！");

return true;

} catch (RuntimeException re) {

log.error("save failed", re);

return false;

}

}

此方法是调用Hiberate去对早已用数据库逆向工程封装号的model类进行数据储存。

在action在方法中，读出信息时，还使用了分页功能，只需要将有限的信息添加到jsp的页面中去，以免信息量过大，具体实现代码如下，

pageCount = buildingservice.findAll().size();

totalPage = pageSize / pageCount + 1;

if(pageSize>pageCount){

totalPage=1;

}

if (pageNumber < 1) {

pageNumber = 1;

}

if (pageNumber > totalPage) {

pageNumber = totalPage;

}

build = buildingservice.querybypage(pageSize, pageNumber);

变量pageNumber就是用来记录所变化到的页码。

在登录方面，本系统在实现了划分权限时，为了是页面简洁，特意在几个实现功能的jsp页面上添加了一段js代码来判断。在登录是session除了记录用户名和密码的同时，还记录了用户的权限级别，到了功能页面js代码会自动判断，如果权限不够，右边div会自动跳转到superadmin.jsp，阻止用户越权操作。具体js代码如下，

<%

String path = request.getContextPath();

String basePath = request.getScheme()+"://"+request.getServerName()+":"+request.getServerPort()+path+"/";

Object obj=ActionContext.getContext().getSession().get("dorman");

Object loginusername=ActionContext.getContext().getSession().get("loginusername");

if(obj!=null||loginusername!=null){

ActionContext.getContext().put("tip", "您的权限不够！");

request.getRequestDispatcher("superadmin.jsp").forward(request, response);

return;

}

%>

除此添加方法以外，删除、修改和查询方法原理也跟上述一样，通过jsp页面提交数据给action，action将其封装好提交到使用动态代理方法的service，service在调用专门对数据库进行操作的dao类，将整个封装好的model类储存到数据库中去（查找则从数据库中读出）。

而tool文件夹下的java类主要用于model之间跨类之间的操作，方便页面之间的信息传递。

本系统利用的是ssh三个框架，将Hiberate框架和struts框架都交给了spring框架去操作，首先要对Hiberate逆向生成的model中文件进行配置，

<property name="mappingResources">

<list>

<value>com/model/Building.hbm.xml</value>

<value>com/model/Student.hbm.xml</value>

<value>com/model/SuperLogin.hbm.xml</value>

<value>com/model/Buildinglevel.hbm.xml</value>

<value>com/model/Dormitory.hbm.xml</value>

<value>com/model/Adminmessage.hbm.xml</value>

<value>com/model/DormLogin.hbm.xml</value></list>

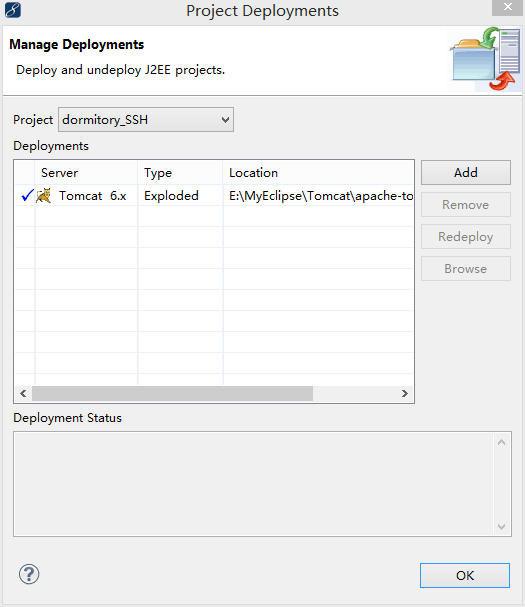
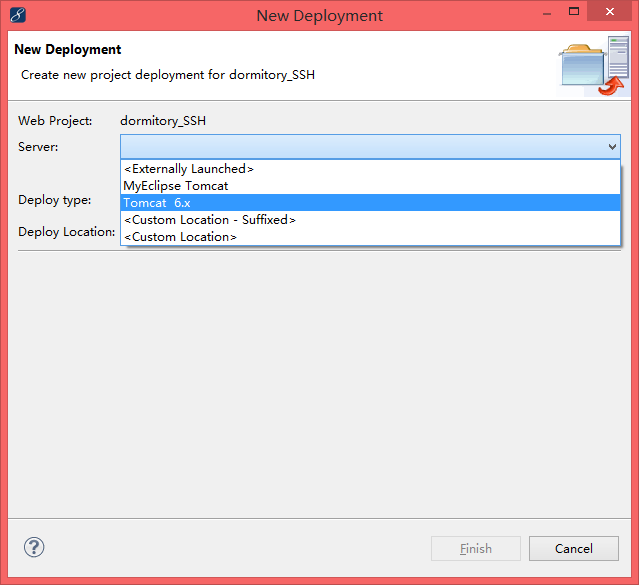
</property>

其次就是将struts交给spring去操作，在配置文件struts.properties中添加配置信息，struts.objectFactory=spring。

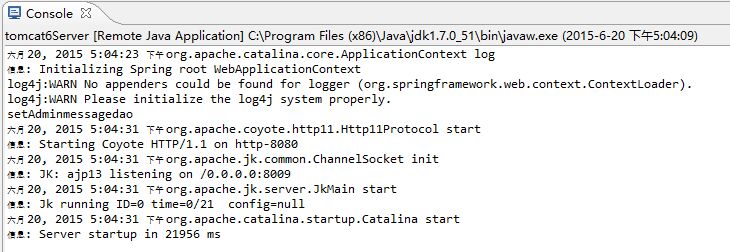
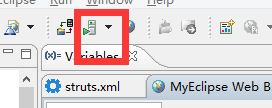
1. **程序发布和测试**

**4.1程序发布过程**

加载到tomcat6.0：

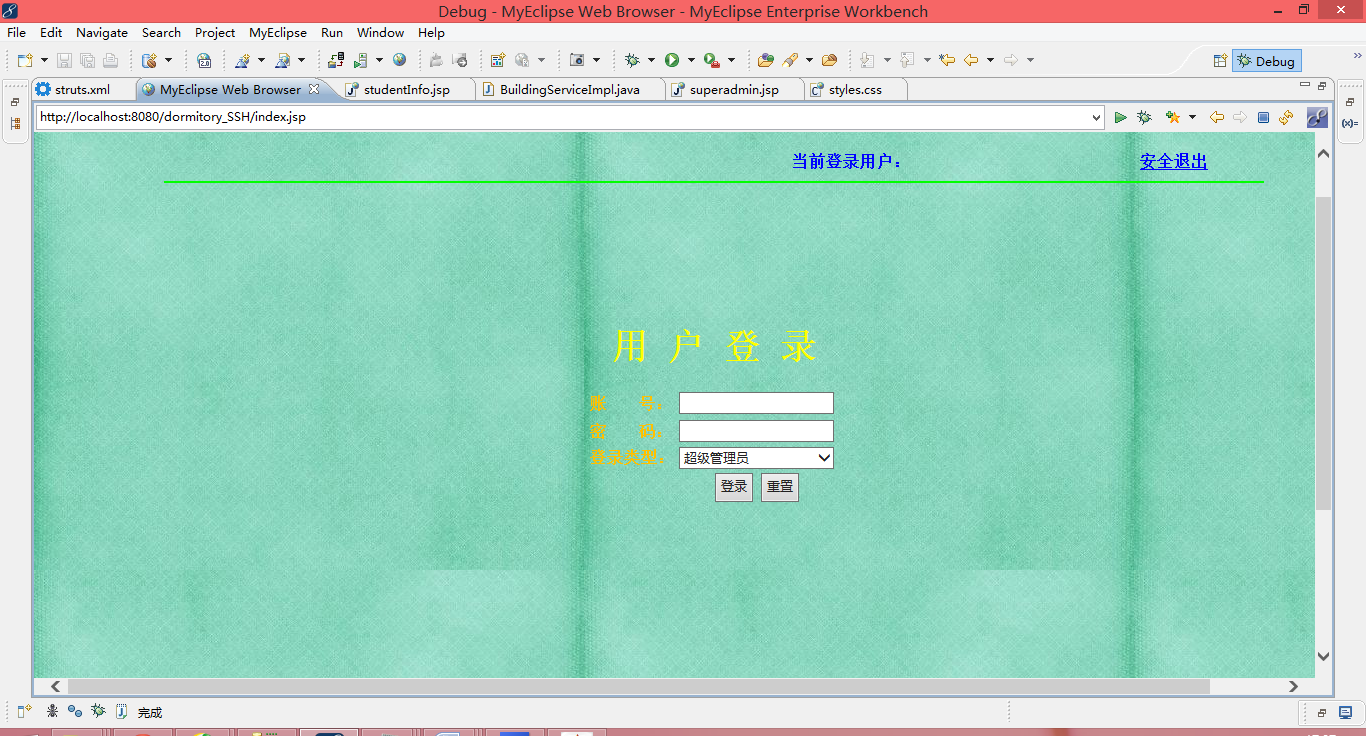


运行tomcat服务器：

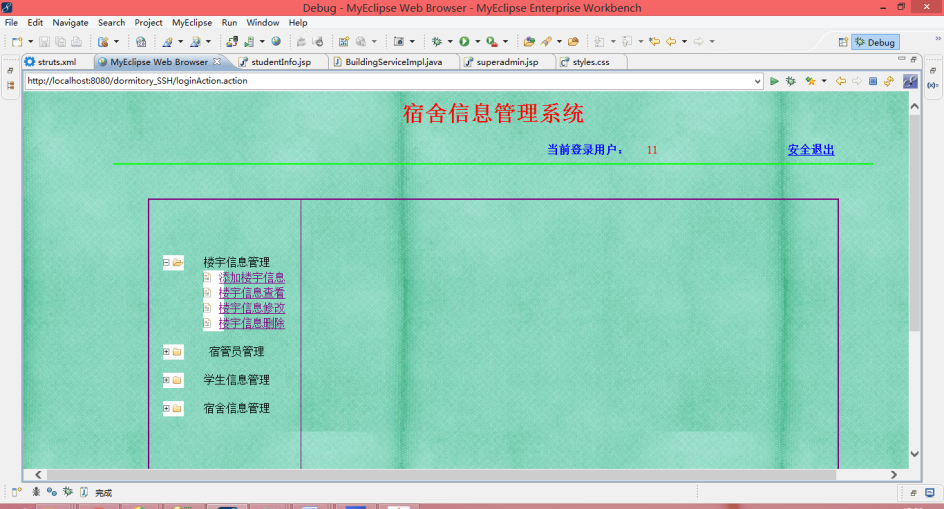


**4.2运行调试**

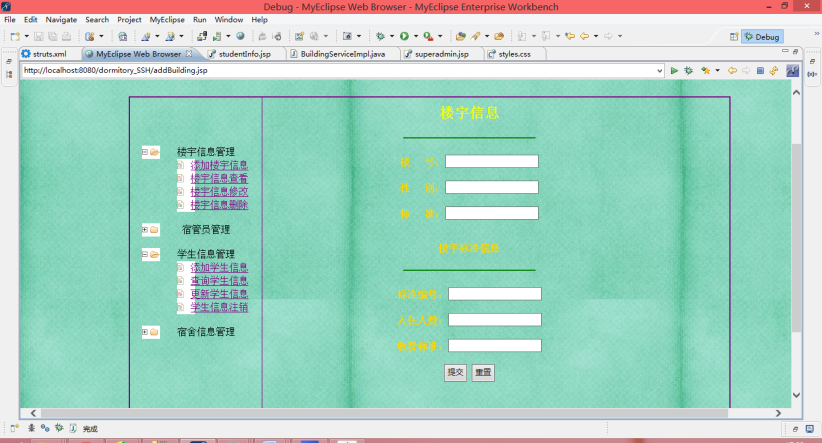
登录界面：



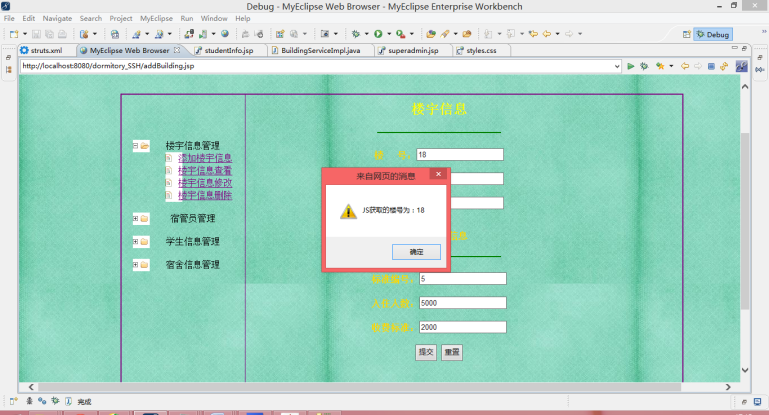
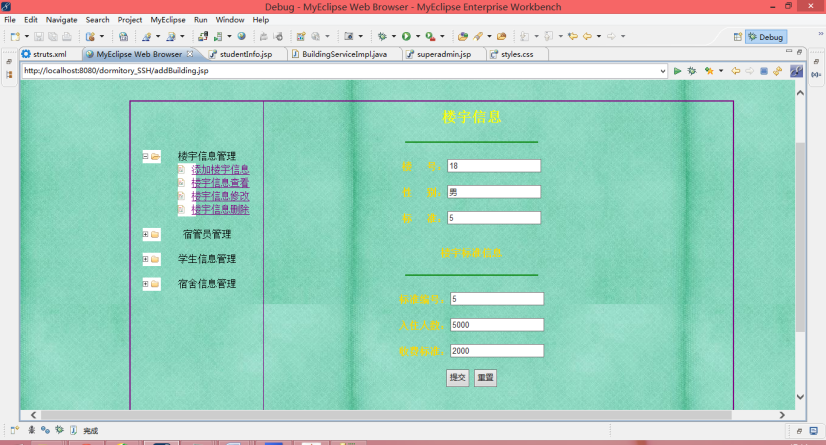
用超级管理员登录后的界面：



添加楼宇信息：



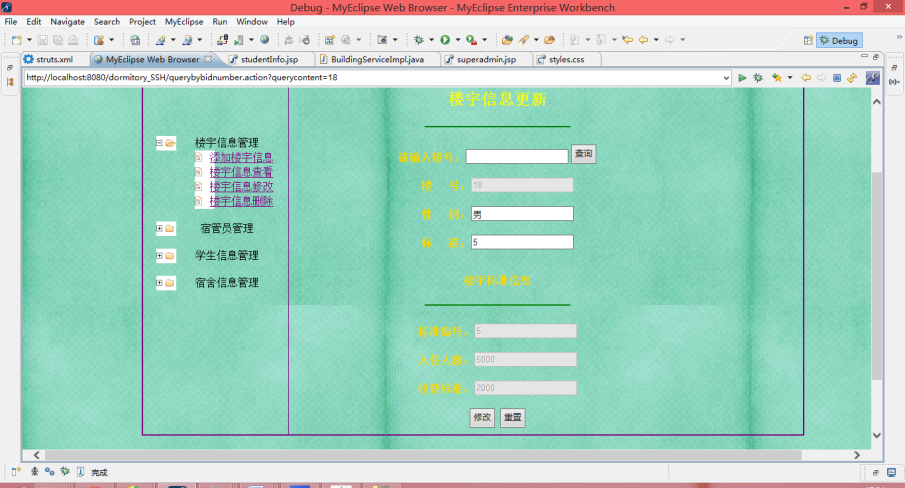
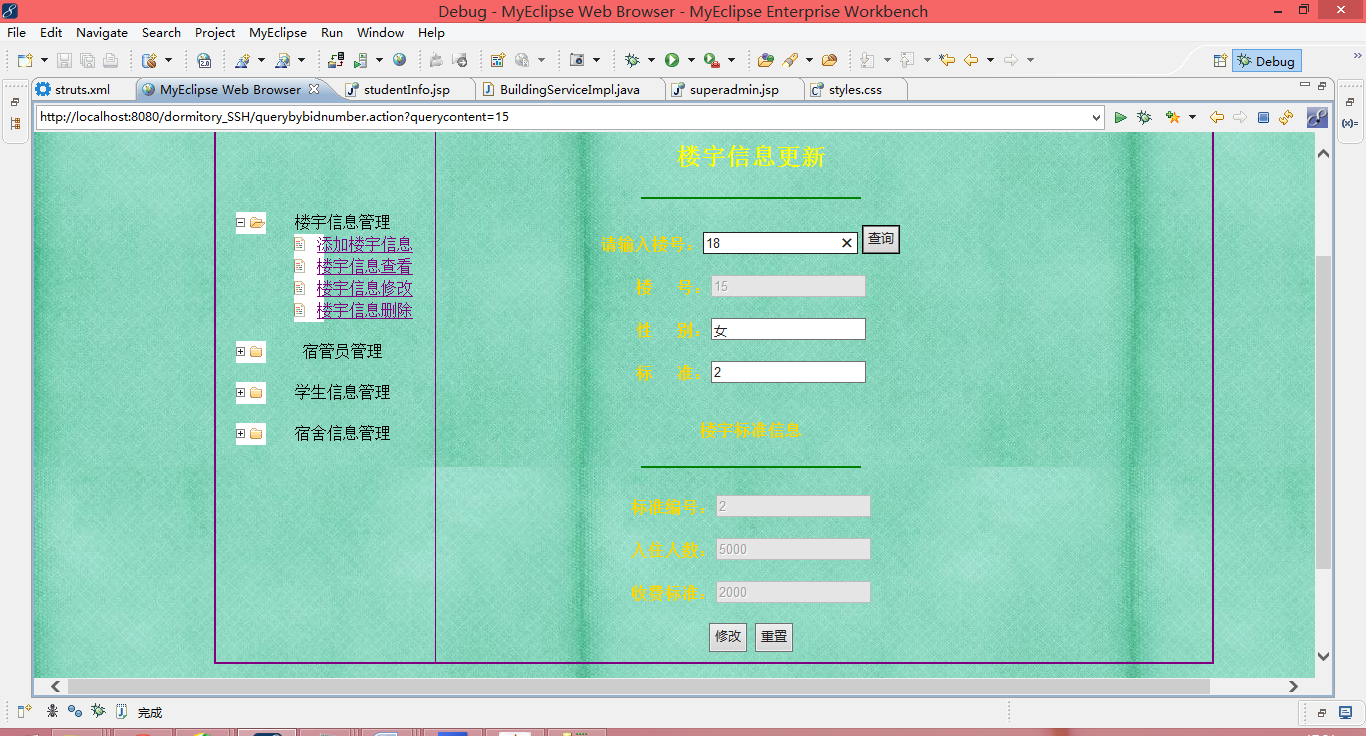
添加信息以后：



楼宇信息查询：



楼宇信息修改：



其他其他模块增删改查操作也是一样。