

# 代码代码

## cn.edu.lingnan.controller

### FlowerAdminController.java

```
1  package cn.edu.lingnan.controller.admin;
2
3  import cn.edu.lingnan.pojo.Flower;
4  import cn.edu.lingnan.service.FlowersService;
5  import cn.edu.lingnan.util.MyUtil;
6  import org.springframework.beans.factory.annotation.Autowired;
7  import org.springframework.stereotype.Controller;
8  import org.springframework.ui.Model;
9  import org.springframework.web.bind.annotation.PathVariable;
10 import org.springframework.web.bind.annotation.RequestMapping;
11 import org.springframework.web.multipart.MultipartFile;
12
13 import javax.servlet.http.HttpServletRequest;
14 import javax.servlet.http.HttpSession;
15 import java.io.File;
16 import java.io.IOException;
17 import java.util.List;
18 import java.util.UUID;
19
20 @Controller
21 @RequestMapping("/admin/flower")
22 public class FlowerAdminController {
23
24     @Autowired
25     private FlowersService flowersService;
26
27     //后台数据显示
28     @RequestMapping("/flowerManagement")
29     public String getList(Model model) {
30         List<Flower> list = flowersService.getList();
31         model.addAttribute("list",list);
32         return "admin/flowers-management";
33     }
34
35     //按关键字查找
36     @RequestMapping("/flowerManagement/sort")
37     public String getByKey(String key,Model model){
38         System.out.println(key);
39         if(key !=null){
40             List<Flower> list = flowersService.getByKeys(key);
41             if (list != null){
42                 model.addAttribute("list",list);
43                 return "admin/flowers-management";
44             }
45         }
46     }
47 }
```

```

46         return getList(model);
47     }
48
49     //删除
50     @RequestMapping(value="/flowerManagement/del/{id}")
51     public String deleteFlower(@PathVariable Integer id, Model model){
52
53         flowerService.deleteById(id);
54
55         return getList(model);
56     }
57
58     //跳转到修改
59     @RequestMapping("/flowerManagement/{id}/jumpUpdate")
60     public String updateFlower(@PathVariable Integer id, HttpSession
session){
61         Flower bean = flowerService.getById(id);
62         session.setAttribute("flower",bean);
63         return "admin/flower-update";
64     }
65
66     //修改
67     @RequestMapping("/flowerManagement/update")
68     public String updateFlower(Flower flower, MultipartFile pictureFile,
Model model, HttpServletRequest request,HttpSession session) throws
IOException {
69
70         //如果有上传图片执行操作
71         String picPath=null;
72         if (!pictureFile.isEmpty()) {
73             //UUID随机生成图片名字
74             String imgName = UUID.randomUUID().toString().replace("-", "");
75             //获取原始文件名并截取后缀
76             String fileName = pictureFile.getOriginalFilename();
77             String imaLastName =
fileName.substring(fileName.lastIndexOf("."));
78             //获取项目路径
79             String filePath =
request.getSession().getServletContext().getRealPath("/")+"flowers/assets/
img/flowers/";
80             //创建本地文件流，存放图片路径
81             File file = new File(filePath+imgName+imaLastName);
82             //写入磁盘
83             pictureFile.transferTo(file);
84             picPath = "flowers/assets/img/flowers/"+imgName+imaLastName;
85         }
86         System.out.println(picPath);
87         //set注入路径
88         flower.setPhoto(picPath);
89         boolean flag = flowerService.updateById(flower);
90         if (flag != true) {
91             model.addAttribute("msg", "修改失败，请重新输入");
92             return "admin/flower-update";
93         }
94         return getList(model);
95     }
96
97     //跳转到添加页面

```

```

98     @RequestMapping("/flowerManagement/jumpAdd")
99     public String jumpAdd(){
100         return "admin/flower-add";
101     }
102
103     //添加,method= {RequestMethod.POST,RequestMethod.GET}
104     @RequestMapping("/flowerManagement/add")
105     public String addFlower(Flower flower, MultipartFile pictureFile,
106 Model model, HttpServletRequest request) throws IOException {
107         //判断是否存在重复商品
108         String name = flower.getName();
109         System.out.println(name);
110         if (flowerService.selectByName(flower.getName())!=null){
111             model.addAttribute("msg", "数据库中已存在此商品");
112             return "admin/flower-add";
113         }
114         //如果有上传图片执行操作
115         String picPath=null;
116         if (!pictureFile.isEmpty()) {
117             //UUID随机生成图片名字
118             String imgName = UUID.randomUUID().toString().replace("-", "");
119             //获取原始文件名并截取图片后缀名
120             String fileName = pictureFile.getOriginalFilename();
121             String imaLastName =
122 fileName.substring(fileName.lastIndexOf("."));
123             //获取项目路径
124             String filePath =
125 request.getSession().getServletContext().getRealPath("/")+"flowers/assets/
126 img/flowers/";
127             //创建本地文件流, 存放图片路径
128             File file = new File(filePath+imgName+imaLastName);
129             //写入磁盘
130             pictureFile.transferTo(file);
131             picPath = "flowers/assets/img/flowers/"+imgName+imaLastName;
132         }
133         System.out.println(picPath);
134         //set注入路径
135         flower.setPhoto(picPath);
136         System.out.println(flower);
137         boolean flag = flowerService.insertFlower(flower);
138         //失败返回添加页面
139         if(flag!=true){
140             model.addAttribute("msg", "添加失败, 请重新输入");
141             return "admin/flower-add";
142         }
143         return getList(model);
144     }
145 }

```

## FlowerController.java

```

1 package cn.edu.lingnan.controller.before;
2
3 import cn.edu.lingnan.pojo.Comment;
4 import cn.edu.lingnan.pojo.Flower;

```

```

5  import cn.edu.lingnan.service.CommentService;
6  import cn.edu.lingnan.service.FlowerService;
7  import cn.edu.lingnan.util.MyUtil;
8  import org.springframework.beans.factory.annotation.Autowired;
9  import org.springframework.stereotype.Controller;
10 import org.springframework.ui.Model;
11 import org.springframework.web.bind.annotation.*;
12
13 import javax.servlet.http.HttpSession;
14 import java.text.ParseException;
15 import java.text.SimpleDateFormat;
16 import java.util.Date;
17 import java.util.List;
18
19 @Controller
20 @RequestMapping("/flower")
21 public class FlowerController {
22
23     @Autowired
24     private FlowerService flowerService;
25     @Autowired
26     private CommentService commentService;
27
28     //跳转至主页
29     @RequestMapping("/flowerHome")
30     public String toFlowerHome(Model model){
31         List<Flower> list = flowerService.getList();
32         //把list传到jsp
33         model.addAttribute("flowerList",list);
34         return "forward:../index.jsp";
35     }
36
37     //商品浏览页
38     @RequestMapping("/flowerShop")
39     public String queryFlowers(Model model) {
40         List<Flower> list = flowerService.getList();
41         //把list传到jsp
42         model.addAttribute("flowerList",list);
43         return "forward:../flowerShop.jsp";
44     }
45
46     //按关键字查找
47     @RequestMapping("/flowerSort")
48     public String queryByKey(String key,Model model){
49         System.out.println(key);
50         if (key != null){
51             List<Flower> list = flowerService.getByKeys(key);
52             model.addAttribute("flowerList",list);
53             return "forward:../flowerShop.jsp";
54         }
55         return queryFlowers(model);
56     }
57
58     //商品详情页+评论显示
59     @RequestMapping(value =("/{id}/detailFlower",method =
60 RequestMethod.GET)
61     public String detailFlower(@PathVariable Integer id, HttpSession
62 session,Model model) {

```

```

61         System.out.println(id);
62         //flower数据
63         Flower bean = flowerService.getById(id);
64         session.setAttribute("flower",bean);
65         //session.getAttribute("flower");
66         //评价
67         List<Comment> list = commentService.getByFlowerId(id);
68         if(list!=null){
69             model.addAttribute("commentList",list);
70         }
71         return "../flowersDetails";
72     }
73
74 }

```

## CommentController.java

```

1  package cn.edu.lingnan.controller.before;
2
3  import cn.edu.lingnan.pojo.Comment;
4  import cn.edu.lingnan.pojo.Orderform;
5  import cn.edu.lingnan.service.CommentService;
6  import cn.edu.lingnan.service.OrderformService;
7  import cn.edu.lingnan.util.MyUtil;
8  import org.springframework.beans.factory.annotation.Autowired;
9  import org.springframework.stereotype.Controller;
10 import org.springframework.ui.Model;
11 import org.springframework.web.bind.annotation.RequestMapping;
12 import org.springframework.web.bind.annotation.RequestMethod;
13
14 import javax.servlet.http.HttpSession;
15 import java.text.ParseException;
16 import java.text.SimpleDateFormat;
17 import java.util.Date;
18
19 @Controller
20 @RequestMapping("/comment")
21 public class CommentController {
22
23     @Autowired
24     CommentService commentService;
25
26     //添加评论
27     @RequestMapping(value="/flower")//method = RequestMethod.POST
28     public String addComment(Comment comment, HttpSession session) throws
    ParseException {
29         System.out.println(comment+"comment");
30         comment.setUserId(MyUtil.getUserId(session));
31         System.out.println(MyUtil.getUserId(session));
32         comment.setLoginName(MyUtil.getAccount(session));
33         comment.setFloId(MyUtil.getFlowerId(session));
34         //String date = new SimpleDateFormat("yyyy-MM-dd
    HH:mm:ss").format(new Date());
35         SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
36         comment.setDate( sdf.parse(sdf.format( new Date() )) );
37         commentService.addComment(comment);

```

```
38         return "../index1";
39     }
40
41 }
```

# cn.edu.lingnan.interceptor

## AdminLoginIntercreptor.java

```
1  package cn.edu.lingnan.interceptor;
2
3  import org.springframework.web.servlet.HandlerInterceptor;
4  import org.springframework.web.servlet.ModelAndView;
5
6  import javax.servlet.http.HttpServletRequest;
7  import javax.servlet.http.HttpServletResponse;
8
9  public class AdminLoginInterceptor implements HandlerInterceptor {
10
11     @Override//请求拦截
12     public boolean preHandle(HttpServletRequest request,
13                             HttpServletResponse response, Object handler) throws Exception {
14         //获取请求地址
15         StringBuffer url = request.getRequestURL();
16         if( url.toString().indexOf("/flowerHome")>= 0 ||
17            url.toString().indexOf("/flowerShop")>= 0 ||
18            url.toString().indexOf("/flowerSort")>= 0 ||
19            url.toString().indexOf("/detailFlower")>= 0 ||
20            url.toString().endsWith("/admin/login") ||
21            url.toString().endsWith("/tologin")
22        ){
23             return true;
24         } else {
25             Object objA = request.getSession().getAttribute("loginAdmin");
26             String path = request.getContextPath();
27             if (objA == null){
28                 request.setAttribute("msg","登录后进行操作");
29                 response.sendRedirect(path+"/tologin");
30             } else {
31                 return true;
32             }
33         }
34         return false;
35     }
36
37     @Override//请求通过
38     public void postHandle(HttpServletRequest request, HttpServletResponse
39                             response, Object handler, ModelAndView modelAndView) throws Exception {
40         System.out.println("postHandle");
41     }
42
43     @Override//请求完成, 页面渲染
44     public void afterCompletion(HttpServletRequest request,
45                                 HttpServletResponse response, Object handler, Exception ex) throws
46                                 Exception {
```

```

43         System.out.println("afterCompletion");
44     }
45
46 }

```

## UserLoginInterceptor.java

```

1  package cn.edu.lingnan.interceptor;
2
3  import org.springframework.web.servlet.HandlerInterceptor;
4  import org.springframework.web.servlet.ModelAndView;
5
6  import javax.servlet.http.HttpServletRequest;
7  import javax.servlet.http.HttpServletResponse;
8
9  public class UserLoginInterceptor implements HandlerInterceptor {
10
11     @Override//请求拦截
12     public boolean preHandle(HttpServletRequest request,
13                             HttpServletResponse response, Object handler) throws Exception {
14         //获取请求地址
15         StringBuffer url = request.getRequestURL();
16         if( url.toString().indexOf("/flowerHome")>= 0 ||
17            url.toString().indexOf("/flowerShop")>= 0 ||
18            url.toString().indexOf("/flowerSort")>= 0 ||
19            url.toString().endsWith("/detailFlower") ||
20            url.toString().endsWith("/user/login") ||
21            url.toString().endsWith("/user/reg")
22         ){
23             return true;
24         } else {
25             Object objU = request.getSession().getAttribute("loginUser");
26             if (objU == null){
27                 request.setAttribute("msg","登录后进行操作");
28
29                 request.getRequestDispatcher("/login.jsp").forward(request,response);
30             } else {
31                 return true;
32             }
33         }
34         return false;
35     }
36
37     @Override//请求通过
38     public void postHandle(HttpServletRequest request, HttpServletResponse
39                             response, Object handler, ModelAndView modelAndView) throws Exception {
40         System.out.println("postHandle");
41     }
42
43     @Override//请求完成，页面渲染
44     public void afterCompletion(HttpServletRequest request,
45                                 HttpServletResponse response, Object handler, Exception ex) throws
46     Exception {
47         System.out.println("afterCompletion");
48     }
49 }

```

# cn.edu.lingnan.mapper

## CommentMapper.java

```
1 package cn.edu.lingnan.mapper;
2
3 import cn.edu.lingnan.pojo.Comment;
4 import org.apache.ibatis.annotations.Mapper;
5 import org.apache.ibatis.annotations.Param;
6 import org.springframework.stereotype.Repository;
7
8 import java.util.List;
9
10 @Mapper
11 @Repository
12 public interface CommentMapper {
13
14     List<Comment> getList();
15
16     List<Comment> getByKey(String key);
17
18     List<Comment> getByBean(Comment bean);
19
20     List<Comment> getUserId(@Param("userId") Integer id);
21
22     List<Comment> getOrderId(@Param("orderId") Integer id);
23
24     List<Comment> getFlowerId(@Param("flowerId") Integer id);
25
26     Comment getById(Integer id);
27
28     boolean addComment(Comment bean);
29
30     boolean deleteById(Integer id);
31
32     boolean updateById(Comment bean);
33
34 }
```

## CommentMapper.xml

```
1 <?xml version="1.0" encoding="UTF-8" ?>
2 <!DOCTYPE mapper
3     PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
4     "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
5 <mapper namespace="cn.edu.lingnan.mapper.CommentMapper">
6
7     <select id="getList" resultType="Comment">
8         select * from comment
9     </select>
10
```



```
11 <select id="getByUserId" resultType="Comment">
12     select * from comment where user_id = #{userId}
13 </select>
14
15 <select id="getByOrderId" resultType="Comment">
16     select * from comment where order_id = #{orderId}
17 </select>
18
19 <select id="getByFlowerId" resultType="Comment">
20     select * from comment where flo_id = #{floId}
21 </select>
22
23 <select id="getByBean" resultType="Comment">
24     select * from comment
25     <where>
26         <if test="id!=null">
27             and id = #{id}
28         </if>
29         <if test="userId!=null">
30             and user_id = #{userId}
31         </if>
32         <if test="loginName!=null">
33             and login_name = #{loginName}
34         </if>
35         <if test="floId!=null">
36             and flo_id = #{floId}
37         </if>
38         <if test="orderId!=null">
39             and order_id = #{orderId}
40         </if>
41     </where>
42 </select>
43
44 <select id="getByKeys" resultType="Comment">
45     select * from comment where concat
46     <trim prefix="(" suffix= ")" suffixOverrides=",">
47         <if test="loginName!=null">
48             login_name,
49         </if>
50         <if test="comment!=null">
51             comment,
52         </if>
53     </trim>
54     like '%${keys}%'
55 </select>
56
57 <insert id="addComment">
58     insert into comment
59     <trim prefix="(" suffix= ")" suffixOverrides=",">
60         <if test="id!=null">
61             id,
62         </if>
63         <if test="userId!=null">
64             user_id,
65         </if>
66         <if test="loginName!=null">
67             login_name,
68         </if>
```

```

69         <if test="floId!=null">
70             flo_id,
71         </if>
72         <if test="orderId!=null">
73             order_id,
74         </if>
75         <if test="date!=null">
76             date,
77         </if>
78         <if test="comment!=null">
79             comment,
80         </if>
81     </trim>
82     values
83     <trim prefix="(" suffix=")" suffixOverrides=",">
84         <if test="id!=null">
85             #{id},
86         </if>
87         <if test="userId!=null">
88             #{userId},
89         </if>
90         <if test="loginName!=null">
91             #{loginName},
92         </if>
93         <if test="floId!=null">
94             #{floId},
95         </if>
96         <if test="orderId!=null">
97             #{orderId},
98         </if>
99         <if test="date!=null">
100             #{date},
101         </if>
102         <if test="comment!=null">
103             #{comment},
104         </if>
105     </trim>
106 </insert>
107
108 <delete id="deleteById">
109     delete from comment where id = #{id}
110 </delete>
111
112 <update id="updateById">
113     update comment set
114     <trim suffixOverrides=",">
115         <if test="name!=null">
116             user_id=#{userId},
117         </if>
118         <if test="loginName!=null">
119             login_name=#{loginName},
120         </if>
121         <if test="floId!=null">
122             flo_id=#{floId},
123         </if>
124         <if test="orderId!=null">
125             order_id=#{orderId},
126         </if>

```

```

127         <if test="date!=null">
128             date=#{date},
129         </if>
130         <if test="comment!=null">
131             comment=#{comment},
132         </if>
133     </trim>
134     where id = #{id}
135 </update>
136
137 </mapper>

```

## FlowerMapper.java

```

1  package cn.edu.lingnan.mapper;
2
3  import cn.edu.lingnan.pojo.Flower;
4  import org.apache.ibatis.annotations.Mapper;
5  import org.apache.ibatis.annotations.Param;
6  import org.springframework.stereotype.Repository;
7
8  import java.util.List;
9
10 @Mapper
11 @Repository
12 public interface FlowerMapper {
13
14     Flower getByBean(Flower bean);
15
16     List<Flower> getByName(String name);
17
18     Flower selectByName(String name);
19
20     Flower getById(Integer id);
21
22     List<Flower> getByKeys(String key);
23
24     List<Flower> getList();
25
26     boolean insertFlower(Flower bean);
27
28     boolean deleteById(Integer id);
29
30     boolean updateById(Flower bean);
31
32 }

```

## FlowerMapper.xml

```

1  <?xml version="1.0" encoding="UTF-8" ?>
2  <!DOCTYPE mapper
3      PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
4      "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
5  <mapper namespace="cn.edu.lingnan.mapper.FlowerMapper">
6

```

```

7      <select id="getByBean" resultType="Flower">
8          select * from flower
9          <where>
10             <if test="id!=null">
11                 and id = #{id}
12             </if>
13             <if test="name!=null">
14                 and name like '%${name}%'
15             </if>
16             <if test="type!=null">
17                 and type like '%${type}%'
18             </if>
19          </where>
20      </select>
21
22      <select id="getByName" resultType="Flower">
23          select * from flower where name like '%${name}%'
24      </select>
25
26      <select id="selectByName" resultType="Flower">
27          select * from flower where name = #{name}
28      </select>
29
30      <select id="getById" resultType="Flower">
31          select * from flower where id = #{id}
32      </select>
33
34      <select id="getByKey" resultType="Flower">
35          select * from flower where concat
36              <trim prefix="(" suffix=")" suffixOverrides=",">
37                  <if test="name!=null">
38                      name,
39                  </if>
40                  <if test="type!=null">
41                      type,
42                  </if>
43                  <if test="floMean!=null">
44                      flo_mean,
45                  </if>
46              </trim>
47          like '%${keys}%'
48      </select>
49
50      <select id="getList" resultType="Flower">
51          select * from flower
52      </select>
53
54      <insert id="insertFlower">
55          insert into flower
56          <trim prefix="(" suffix=")" suffixOverrides=",">
57              <if test="id!=null">
58                  id,
59              </if>
60              <if test="name!=null">
61                  name,
62              </if>
63              <if test="type!=null">
64                  type,

```

```

65         </if>
66         <if test="stock!=null">
67             stock,
68         </if>
69         <if test="price!=null">
70             price,
71         </if>
72         <if test="floMean!=null">
73             flo_mean,
74         </if>
75         <if test="photo!=null">
76             photo,
77         </if>
78     </trim>
79     values
80     <trim prefix="(" suffix=)" suffixOverrides=",">
81         <if test="id!=null">
82             #{id},
83         </if>
84         <if test="name!=null">
85             #{name},
86         </if>
87         <if test="type!=null">
88             #{type},
89         </if>
90         <if test="stock!=null">
91             #{stock},
92         </if>
93         <if test="price!=null">
94             #{price},
95         </if>
96         <if test="floMean!=null">
97             #{floMean},
98         </if>
99         <if test="photo!=null">
100             #{photo},
101         </if>
102     </trim>
103 </insert>
104
105 <delete id="deleteById">
106     delete from flower where id = #{id}
107 </delete>
108
109 <update id="updateById">
110     update flower set
111     <trim suffixOverrides=",">
112         <if test="name!=null">
113             name=#{name},
114         </if>
115         <if test="type!=null">
116             type=#{type},
117         </if>
118         <if test="stock!=null">
119             stock=#{stock},
120         </if>
121         <if test="price!=null">
122             price=#{price},

```

```

123         </if>
124         <if test="floMean!=null">
125             flo_mean=#{floMean},
126         </if>
127         <if test="photo!=null">
128             photo=#{photo},
129         </if>
130     </trim>
131     where id = #{id}
132 </update>
133
134 </mapper>

```

## cn.edu.lingnan.pojo

### Comment.java

```

1  package cn.edu.lingnan.pojo;
2
3  import org.springframework.stereotype.Component;
4
5  import java.util.Date;
6
7  @Component
8  public class Comment {
9
10     private Integer id;
11     private Integer userId;
12     private String loginName;
13     private Integer floId;
14     private Integer orderId;
15     private Date date;//评论时间
16     private String comment;
17
18     public Comment() {
19     }
20
21     public Comment(Integer id, Integer userId, String loginName, Integer
floId, Integer orderId, Date date, String comment) {
22         this.id = id;
23         this.userId = userId;
24         this.loginName = loginName;
25         this.floId = floId;
26         this.orderId = orderId;
27         this.date = date;
28         this.comment = comment;
29     }
30
31     public Integer getId() {
32         return id;
33     }
34
35     public void setId(Integer id) {
36         this.id = id;
37     }

```

```
38
39     public Integer getUserId() {
40         return userId;
41     }
42
43     public void setUserId(Integer userId) {
44         this.userId = userId;
45     }
46
47     public String getLoginName() {
48         return loginName;
49     }
50
51     public void setLoginName(String loginName) {
52         this.loginName = loginName;
53     }
54
55     public Integer getFloId() {
56         return floId;
57     }
58
59     public void setFloId(Integer floId) {
60         this.floId = floId;
61     }
62
63     public Integer getOrderId() {
64         return orderId;
65     }
66
67     public void setOrderId(Integer orderId) {
68         this.orderId = orderId;
69     }
70
71     public Date getDate() {
72         return date;
73     }
74
75     public void setDate(Date date) {
76         this.date = date;
77     }
78
79     public String getComment() {
80         return comment;
81     }
82
83     public void setComment(String comment) {
84         this.comment = comment;
85     }
86
87     @Override
88     public String toString() {
89         return "Comment{" +
90             "id=" + id +
91             ", userId=" + userId +
92             ", loginName='" + loginName + '\'' +
93             ", floId=" + floId +
94             ", orderId=" + orderId +
95             ", date=" + date +
```

```
96         ", comment='" + comment + '\\'" +
97         '}'';
98     }
99
100 }
```

## Flower.java

```
1  package cn.edu.lingnan.pojo;
2
3  import org.springframework.stereotype.Component;
4
5  @Component
6  public class Flower {
7      //商品
8      private Integer id;
9      private String name;
10     private String type;
11     private Integer stock;
12     private Float price;
13     private String floMean;
14     private String photo;
15
16     public Flower() {
17     }
18
19     public Flower(Integer id, String name, String type, Integer stock,
20     Float price, String floMean, String photo) {
21         this.id = id;
22         this.name = name;
23         this.type = type;
24         this.stock = stock;
25         this.price = price;
26         this.floMean = floMean;
27         this.photo = photo;
28     }
29
30     public Integer getId() {
31         return id;
32     }
33
34     public void setId(Integer id) {
35         this.id = id;
36     }
37
38     public String getName() {
39         return name;
40     }
41
42     public void setName(String name) {
43         this.name = name;
44     }
45
46     public String getType() {
47         return type;
48     }
49 }
```



```

48
49     public void setType(String type) {
50         this.type = type;
51     }
52
53     public Integer getStock() {
54         return stock;
55     }
56
57     public void setStock(Integer stock) {
58         this.stock = stock;
59     }
60
61     public Float getPrice() {
62         return price;
63     }
64
65     public void setPrice(Float price) {
66         this.price = price;
67     }
68
69     public String getFloMean() {
70         return floMean;
71     }
72
73     public void setFloMean(String floMean) {
74         this.floMean = floMean;
75     }
76
77     public String getPhoto() {
78         return photo;
79     }
80
81     public void setPhoto(String photo) {
82         this.photo = photo;
83     }
84
85     @Override
86     public String toString() {
87         return "Flower{" +
88             "id=" + id +
89             ", name='" + name + '\'' +
90             ", type='" + type + '\'' +
91             ", stock=" + stock +
92             ", price=" + price +
93             ", floMean='" + floMean + '\'' +
94             ", photo='" + photo + '\'' +
95             '}';
96     }
97
98 }

```

**cn.edu.lingnan.service**

**CommentService.java**

```

1 package cn.edu.lingnan.service;
2
3 import cn.edu.lingnan.pojo.Comment;
4 import org.apache.ibatis.annotations.Param;
5
6 import java.util.List;
7
8 public interface CommentService {
9
10     List<Comment> getList();
11
12     List<Comment> getByKeys(String key);
13
14     List<Comment> getByBean(Comment bean);
15
16     List<Comment> getUserId(Integer id);
17
18     List<Comment> getOrderId(Integer id);
19
20     List<Comment> getFlowerId(Integer id);
21
22     Comment getById(Integer id);
23
24     boolean addComment(Comment bean);
25
26     boolean deleteById(Integer id);
27
28     boolean updateById(Comment bean);
29
30 }

```

## CommentServiceImpl.java

```

1 package cn.edu.lingnan.service.impl;
2
3
4 import cn.edu.lingnan.mapper.CommentMapper;
5 import cn.edu.lingnan.pojo.Comment;
6 import cn.edu.lingnan.service.CommentService;
7 import org.springframework.beans.factory.annotation.Autowired;
8 import org.springframework.stereotype.Service;
9
10 import java.util.List;
11
12 @Service
13 public class CommentServiceImpl implements CommentService {
14
15     @Autowired
16     CommentMapper commentMapper;
17
18     @Override
19     public List<Comment> getList() {
20         return commentMapper.getList();
21     }
22
23     @Override

```

```

24     public List<Comment> getByKeys(String key) {
25         return commentMapper.getByKeys(key);
26     }
27
28     @Override
29     public List<Comment> getByBean(Comment bean) {
30         return null;
31     }
32
33     @Override
34     public List<Comment> getUserId(Integer id) {
35         return commentMapper.getUserId(id);
36     }
37
38     @Override
39     public List<Comment> getOrderId(Integer id) {
40         return commentMapper.getOrderId(id);
41     }
42
43     @Override
44     public List<Comment> getFlowerId(Integer id) {
45         return commentMapper.getFlowerId(id);
46     }
47
48     @Override
49     public Comment getById(Integer id) {
50         return commentMapper.getById(id);
51     }
52
53     @Override
54     public boolean addComment(Comment bean) {
55         return commentMapper.addComment(bean);
56     }
57
58     @Override
59     public boolean deleteById(Integer id) {
60         return commentMapper.deleteById(id);
61     }
62
63     @Override
64     public boolean updateById(Comment bean) {
65         return updateById(bean);
66     }
67
68 }

```

## FlowerService.java

```

1     package cn.edu.lingnan.service;
2
3     import cn.edu.lingnan.pojo.Flower;
4
5     import java.util.List;
6
7     public interface FlowerService {
8

```

```

9      Flower getByBean(Flower bean);
10
11     List<Flower> getByName(String name);
12
13     Flower selectByName(String name);
14
15     Flower getById(Integer id);
16
17     List<Flower> getByKeys(String key);
18
19     List<Flower> getList();
20
21     boolean insertFlower(Flower bean);
22
23     boolean deleteById(Integer id);
24
25     boolean updateById(Flower bean);
26
27 }

```

## FlowerServiceImpl.java

```

1  package cn.edu.lingnan.service.impl;
2
3  import cn.edu.lingnan.mapper.FlowerMapper;
4  import cn.edu.lingnan.pojo.Flower;
5  import cn.edu.lingnan.service.FlowerService;
6  import org.springframework.beans.factory.annotation.Autowired;
7  import org.springframework.stereotype.Service;
8
9  import java.util.List;
10
11  @Service
12  public class FlowerServiceImpl implements FlowerService {
13
14      @Autowired
15      private FlowerMapper flowerMapper;
16
17      @Override
18      public Flower getByBean(Flower bean) {
19          return flowerMapper.getByBean(bean);
20      }
21
22      @Override
23      public List<Flower> getByName(String name) {
24          return flowerMapper.getByName(name);
25      }
26
27      @Override
28      public Flower selectByName(String name) {
29          return flowerMapper.selectByName(name);
30      }
31
32      @Override
33      public Flower getById(Integer id) {
34          return flowerMapper.getById(id);

```

```

35     }
36
37     @Override
38     public List<Flower> getByKeys(String key) {
39         return flowerMapper.getByKeys(key);
40     }
41
42     @Override
43     public List<Flower> getList() {
44         return flowerMapper.getList();
45     }
46
47     @Override
48     public boolean insertFlower(Flower bean) {
49         return flowerMapper.insertFlower(bean);
50     }
51
52     @Override
53     public boolean deleteById(Integer id) {
54         return flowerMapper.deleteById(id);
55     }
56
57     @Override
58     public boolean updateById(Flower bean) {
59         return flowerMapper.updateById(bean);
60     }
61
62 }

```

# cn.edu.lingnan.util

## MyUtil.java

```

1  package cn.edu.lingnan.util;
2  import cn.edu.lingnan.pojo.Admin;
3  import cn.edu.lingnan.pojo.Flower;
4  import cn.edu.lingnan.pojo.User;
5
6  import java.text.SimpleDateFormat;
7  import java.util.Date;
8  import javax.servlet.http.HttpSession;
9  public class MyUtil {
10     /**
11      * 获得时间字符串
12      */
13     public static String getStringID(){
14         String id=null;
15         Date date=new Date();
16         SimpleDateFormat sdf=new SimpleDateFormat("yyyyMMddHHmmssSSS");
17         id=sdf.format(date);
18         return id;
19     }
20
21     /**
22      * 获得用户ID

```

```

23     */
24     public static Integer getUserId(HttpSession session) {
25         User user = (User)session.getAttribute("loginUser");
26         return user.getId();
27     }
28
29     /**
30      * 获得用户登录名
31      */
32     public static String getAccount(HttpSession session) {
33         User user = (User)session.getAttribute("loginUser");
34         return user.getAccount();
35     }
36
37     /**
38      * 获得AdminName
39      */
40     public static String getAdminName(HttpSession session) {
41         Admin admin = (Admin)session.getAttribute("loginAdmin");
42         return admin.getName();
43     }
44
45     /**
46      * 获得商品ID
47      */
48     public static Integer getFlowerId(HttpSession session) {
49         Flower flower = (Flower)session.getAttribute("flower");
50         return flower.getId();
51     }
52
53     /**
54      * 获得商品名
55      */
56     public static String getFlowerName(HttpSession session) {
57         Flower flower = (Flower)session.getAttribute("flower");
58         return flower.getName();
59     }
60
61 }

```

## 部分配置

为什么是部分呢，因为其他的大家都一样反正都是自己导入的哈哈哈

### spring-mvc.xml

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <beans xmlns="http://www.springframework.org/schema/beans"
3       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4       xmlns:mvc="http://www.springframework.org/schema/mvc"
5       xsi:schemaLocation="http://www.springframework.org/schema/beans
6
7       http://www.springframework.org/schema/beans/spring-beans.xsd
8                           http://www.springframework.org/schema/mvc

```

```

8      https://www.springframework.org/schema/mvc/spring-mvc.xsd
9      ">
10
11      <bean
12      class="org.springframework.web.servlet.view.InternalResourceViewResolver">
13          <property name="prefix" value="/WEB-INF/pages/"></property>
14          <property name="suffix" value=".jsp"></property>
15      </bean>
16
17      <!-- &lt;!&dash;静态资源需要单独处理，不需要dispatcher servlet&dash;&gt;
18      <mvc:annotation-driven />
19      <mvc:resources mapping="/css/**" location="/css/" />
20      <mvc:resources mapping="/layui/**" location="/layui/">-->
21      <mvc:default-servlet-handler />
22
23      <!--将springmvc，功能全开-->
24      <!--注解驱动-->
25      <mvc:annotation-driven />
26      <!--SpringMVC conversion-service接口，数据类型转换-->
27      <mvc:annotation-driven conversion-service="convertService" />
28      <bean id="convertService"
29      class="org.springframework.format.support.FormattingConversionServiceFactor
30      yBean"/>
31
32      <!--多媒体解析器，id只能是multipartResolver-->
33      <bean id="multipartResolver"
34      class="org.springframework.web.multipart.commons.CommonsMultipartResolver">
35          <property name="defaultEncoding" value="utf-8"/>
36          <!-- 最大上传大小 8m=8*1024*1024b -->
37          <property name="maxUploadSize" value="8388608"></property>
38      </bean>
39
40      <!--静态资源放行-->
41      <mvc:resources location="/" mapping="/*"/>
42
43      <!--拦截器注册-->
44      <mvc:interceptors>
45          <!--用户拦截-->
46          <mvc:interceptor>
47              <mvc:mapping path="/flower/**" />
48              <mvc:mapping path="/comment/**" />
49              <mvc:mapping path="/cart/**" />
50              <mvc:mapping path="/orderform/**" />
51              <mvc:mapping path="/user/**" />
52              <mvc:exclude-mapping path="/bootstrap/**" />
53              <mvc:exclude-mapping path="/flowers/**" />
54              <mvc:exclude-mapping path="/layui/**" />
55              <bean class="cn.edu.lingnan.interceptor.UserLoginInterceptor"
56              />
57          </mvc:interceptor>
58          <!--管理员拦截-->
59          <mvc:interceptor>
60              <mvc:mapping path="/admin/**" />
61              <mvc:exclude-mapping path="/bootstrap/**" />
62              <mvc:exclude-mapping path="/flowers/**" />
63              <mvc:exclude-mapping path="/layui/**" />

```

```

59         <bean class="cn.edu.lingnan.interceptor.AdminLoginInterceptor"
/>
60     </mvc:interceptor>
61 </mvc:interceptors>
62
63 </beans>

```

# jsp

## flower-management.jsp

```

<div style="width:1000px;margin-left: 10px;margin-bottom: 10px">
<form action="${ctx}/admin/flower/flowerManagement/sort" method="post" style="...">
    <input type="text" name="key" lay-verify="required" placeholder="Input the keywords..." autocomplete="off" class="layui-input" style="...">
    <button type="submit" class="layui-btn" style="background-color: #4A708B;margin-top: 10px;margin-left: 5px">
        <i class="layui-icon layui-icon-search"></i> Sort
    </button>
</form>
<form action="${ctx}/admin/flower/flowerManagement/jumpAdd" style="...">
    <button type="submit" class="layui-btn" style="...">
        <i class="layui-icon">&#xe608;</i> 添加商品
    </button>
</form>
</div>

```

```

<tbody>
<!-- 数据内容 -->
<c:forEach items="${requestScope.get('list')}" var="flower">
    <tr>
        <td>${flower.getId()}</td>
        <td>${flower.getName()}</td>
        <td>${flower.getType()}</td>
        <td>${flower.getPrice()}</td>
        <td>${flower.getStock()}</td>
        <td>${flower.getFloMean()}</td>
        <td></td>
        <td>
            <a class="layui-btn layui-btn-xs" lay-event="edit" href="${ctx}/admin/flower/flowerManagement/${flower.getId()}/jumpUpdate">修改</a>
            <br>
            <a class="layui-btn layui-btn-xs" lay-event="edit" href="${ctx}/admin/flower/flowerManagement/del/${flower.getId()}" onclick="return con
        </td>
    </tr>
</c:forEach>
</tbody>

```

```

" onclick="return confirm('是否确定删除该条记录?');">删除</a>

```

## flower-add.jsp

```

<form class="layui-form layui-form-pane" action="${ctx}/admin/flower/flowerManagement/add" method="post" enctype="multipart/form-data">

```



```

<div class="layui-input-block">
  <!-- 用一个div来当作美化的上传按钮，file按钮被透明化 -->
  <div id="btn" style="...">
    文件上传
    <input type="file" name="pictureFile" class="file" onclick="daojishi()" <!--onclick="daojishi()"-->
  </div>
  <!-- 图片回显 -->
  <c:if test="${sessionScope.flower.photo!=null}">
    
  </c:if>--%>
  <!-- 文件名显示区域 -->
  <div id="filename"></div>
</div>
${requestScope.msg}
<button type="submit" class="layui-btn" id="test1" style="...">
  <i class="layui-icon">&#xe67c;</i>提交
</button>

<script src="https://www.jq22.com/jquery/jquery-3.3.1.js"></script>
<!-- 轮询文件名 -->
<script>
  function daojishi() {
    setInterval("getname()",1000);
  }
</script>

```

```

<!-- 获取文件名 -->
<script>
  function getname() {
    var filename = $("#btn .file").val();
    $("#filename").text(filename);
  }
</script>

```

## flower-update.jsp

和添加区别不大！文件回显只做了修改前的回显，修改后的还是出不来哈哈哈，因为session没改

```

<h1 style="...">修改${sessionScope.flower.name}</h1>
<div style="..."><!--enctype="multipart/form-data" modelAttribute="flower"-->
  <form class="layui-form layui-form-pane" action="${ctx}/admin/flower/flowerManagement/update" method="post" enctype="multipa
  <input type="hidden" name="id" value="${sessionScope.flower.id}">
  <div class="layui-form-item" style="...">
    <label class="layui-form-label">商品名</label>
    <div class="layui-input-block">
      <input type="text" name="name" value="${sessionScope.flower.name}" lay-verify="required" placeholder="Name" auto
    </div>
  </div>
  <div class="layui-form-item" style="...">
    <label class="layui-form-label">种类</label>
    <div class="layui-input-block">
      <input type="text" name="type" value="${sessionScope.flower.type}" lay-verify="required" placeholder="Type" auto
    </div>
  </div>
</div>

```

```

<div class="layui-form-item" style="...">
  <div class="layui-input-block">
    <!-- 用一个div来当作美化的上传按钮，file按钮被透明化 -->
    <div id="btn" style="...">
      文件上传
      <input type="file" name="pictureFile" class="file" onclick="daojishi()" <!--onclick="daojishi()"-->
    </div>
    <c:if test="${sessionScope.flower.photo!=null}">
      
    </c:if>
    <!-- 文件名显示区域 -->
    <div id="filename"></div>
  </div>
  ${requestScope.msg}
  <button type="submit" class="layui-btn" id="test1" style="...">
    <i class="layui-icon">&#xe67c;</i>提交
  </button>

  <script src="https://www.jq22.com/jquery/jquery-3.3.1.js"></script>
  <!-- 轮询文件名 -->
  <script>
    function daojishi() {
      setInterval("getname()",1000);
    }
  </script>

```

## flowerShop.jsp

...价格小细节不要在意，数据库里没设计原价，就干脆先统一处理了，假装自己在打折

```

<!-- 商品浏览视图模式 -->
<!-- product item list wrapper start -->
<div class="shop-product-wrap grid-view row mbn-40">
  <!-- 遍历list内数据 -->
  <c:forEach items="${requestScope.get('flowerList')}" var="flower">
    <!-- product single item start -->
    <div class="col-md-4 col-sm-6">
      <!-- product grid start -->
      <div class="product-item">
        <figure class="product-thumb">
          <a href="${ctx}/flower/${flower.getId()}/detailFlower">
            <!--<input name="name" type="hidden" value="${flower.getName()}"-->
            
            
          </a>
          <div class="button-group">

```

```

<div class="product-caption">
  <p class="product-name">
    <a href="${ctx}/flower/{name}/detailFlower">${flower.getName()}</a>
  </p>
  <div class="price-box">
    <span class="price-regular">¥ ${flower.getPrice()}</span>
    <span class="price-old"><del>¥ ${flower.getPrice()*1.2}</del></span>
  </div>
</div>

```

```

<!-- 商品浏览列表模式 -->
<!-- product list item end -->
<div class="product-list-item">
    <figure class="product-thumb">
        <a href="{ctx}/flower/{name}/detailFlower">
            
            
        </a>
    </figure>
    <div class="product-content-list">
        <h5 class="product-name"><a href="{ctx}/flower/{name}/detailFlower">${flower.getName()}</a></h5>
        <div class="price-box">
            <span class="price-regular">¥${flower.getPrice()}</span>
            <span class="price-old"><del>¥${flower.getPrice()*1.2}</del></span>
        </div>
    </div>

```

## flowersDetail.jsp

```

        
    </div>
</div>
<div class="col-lg-7">
    <div class="product-details-des">
        <br>
        <h1 class="product-name">${sessionScope.flower.name}</h1>
        <div class="ratings d-flex">
            <span><i class="lnr lnr-star"></i></span>
            <span><i class="lnr lnr-star"></i></span>
            <span><i class="lnr lnr-star"></i></span>
            <span><i class="lnr lnr-star"></i></span>
            <span><i class="lnr lnr-star"></i></span>
            <div class="pro-review">
                <span>998 Reviews</span>
            </div>
        </div>
        <div class="price-box">
            <span class="detail-price-regular"><font size="9">¥${sessionScope.flower.price}</font></span>
            <span class="detail-price-old "><font size="6"><del>¥${sessionScope.flower.price*1.2}</del></font></span>
        </div>
        <br><br>
    </div>

```

```

        <i class="fa fa-check-circle"></i>
        <span><font size="5">${sessionScope.flower.stock} in stock</font></span>
    </div>
    <br><br>
    <!-- 花语 -->
    <p class="pro-desc"><font size="3">${sessionScope.flower.floMean}</font></p>
    <br><br>

```

```

<!-- 评论显示 -->
<div class="row" style="...">
  <h2 style="...">商品评价</h2>
  <c:forEach items="${requestScope.get('commentlist')}}" var="comment">
    <div style="...">
      <div class="layui-card" style="...">
        <div class="layui-card-header">
          ${comment.getLoginName()}<p style="...">${comment.getDate()}</p>
        </div>
        <div class="layui-card-body">
          ${comment.getComment()}
        </div>
      </div>
    </div>
  </c:forEach>
<!-- 提交评论 -->
<div style="...">
  <form action="${ctx}/comment/flower" method="post" class="layui-form layui-form-pane">
    <div class="layui-form-item layui-form-text">
      <label class="layui-form-label" style="...">对该商品进行评价</label>
      <div class="layui-input-block" style="...">
        <textarea name="comment" placeholder="请输入内容..." class="layui-textarea"></textarea>
      </div>
      <button type="submit" class="layui-btn layui-btn-radius layui-btn-warm layui-btn-fluid" style="...">提交</button>
    </div>
  </form>
</div>
</div>

```

## index1.jsp

主要是因为index页面是商品首页，如果要显示数据的话必须经过controller，所以用了个jsp自动转发

```

<body>
  <jsp:forward page="flower/flowerHome"></jsp:forward>
</body>

```

## index.jsp

和商品浏览是一样的操作，只不过这里的foreach设置了步长，只显示部分商品

```

<c:forEach items="${requestScope.get('flowerList')}}" var="flower" begin="1" end="8">
  <div class="col-lg-3 col-md-4 col-sm-6">
    <div class="product-item mt-40">
      <figure class="product-thumb">
        <a href="${ctx}/flower/${flower.getId()}/detailFlower">
          
          
        </a>
        <div class="product-badge">
          <div class="product-label new">
            <span>new</span>
          </div>
          <div class="product-label discount">
            <span>10%</span>
          </div>
        </div>
        <div class="button-group">
          <a href="#" data-toggle="tooltip" data-placement="left" title="Add to wishlist"><i class="lnr lnr-heart"></i></a>
          <a href="#" data-toggle="modal" data-target="#quick_view"><span data-toggle="tooltip" data-placement="left" title="Quick View"></span></a>
          <a href="#" data-toggle="tooltip" data-placement="left" title="Add to Cart"><i class="lnr lnr-cart"></i></a>
        </div>
      </figure>
      <div class="product-caption">
        <p class="product-name">
          <a href="${ctx}/flower/{name}/detailFlower">${flower.getName()}</a>
        </p>

```

帮忙修改的不放了，就那一两行