代码代码

cn.edu.lingnan.controller

FlowerAdminController.java

```java
package cn.edu.lingnan.controller.admin;

import cn.edu.lingnan.pojo.Flower;
import cn.edu.lingnan.service.FlowerService;
import cn.edu.lingnan.util.MyUtil;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.multipart.MultipartFile;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpSession;
import java.io.File;
import java.io.IOException;
import java.util.List;
import java.util.UUID;

@Controller
@RequestMapping("/admin/flower")
public class FlowerAdminController {

    @Autowired
    private FlowerService flowerService;

    //后台数据显示
    @RequestMapping("/flowerManagement")
    public String getList(Model model) {
        List<Flower> list = flowerService.getList();
        model.addAttribute("list",list);
        return "admin/flowers-management";
    }

    //按关键字查找
    @RequestMapping("/flowerManagement/sort")
    public String getByKey(String key,Model model){
        System.out.println(key);
        if(key !=null){
            List<Flower> list = flowerService.getByKeys(key);
            if (list != null){
                model.addAttribute("list",list);
                return "admin/flowers-management";
            }
        }
```

```
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
```

```
    @RequestMapping("/flowerManagement/jumpAdd")
    public String jumpAdd(){
        return "admin/flower-add";
    }

    //添加,method= {RequestMethod.POST,RequestMethod.GET}
    @RequestMapping("/flowerManagement/add")
    public String addFlower(Flower flower, MultipartFile pictureFile,
Model model, HttpServletRequest request) throws IOException {
        //判断是否存在重复商品
        String name = flower.getName();
        System.out.println(name);
        if (flowerService.selectByName(flower.getName())!=null){
            model.addAttribute("msg", "数据库中已存在此商品");
            return "admin/flower-add";
        }
        //如果有上传图片执行操作
        String picPath=null;
        if (!pictureFile.isEmpty()) {
            //UUID随机生成图片名字
            String imgName = UUID.randomUUID().toString().replace("-","");
            //获取原始文件名并截取图片后缀名
            String fileName = pictureFile.getOriginalFilename();
            String imaLastName =
fileName.substring(fileName.lastIndexOf("."));
            //获取项目路径
            String filePath =
request.getSession().getServletContext().getRealPath("/")+"flowers/assets/
img/flowers/";
            //创建本地文件流，存放图片路径
            File file = new File(filePath+imgName+imaLastName);
            //写入磁盘
            pictureFile.transferTo(file);
            picPath = "flowers/assets/img/flowers/"+imgName+imaLastName;
        }
        System.out.println(picPath);
        //set注入路径
        flower.setPhoto(picPath);
        System.out.println(flower);
        boolean flag = flowerService.insertFlower(flower);
        //失败返回添加页面
        if(flag!=true){
            model.addAttribute("msg", "添加失败，请重新输入");
            return "admin/flower-add";
        }
        return getList(model);
    }

}
```

FlowerController.java

```
package cn.edu.lingnan.controller.before;

import cn.edu.lingnan.pojo.Comment;
import cn.edu.lingnan.pojo.Flower;
```

```java
import cn.edu.lingnan.service.CommentService;
import cn.edu.lingnan.service.FlowerService;
import cn.edu.lingnan.util.MyUtil;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.*;

import javax.servlet.http.HttpSession;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.List;

@Controller
@RequestMapping("/flower")
public class FlowerController {

    @Autowired
    private FlowerService flowerService;
    @Autowired
    private CommentService commentService;

    //跳转至主页
    @RequestMapping("/flowerHome")
    public String toFlowerHome(Model model){
        List<Flower> list = flowerService.getList();
        //把list传到jsp
        model.addAttribute("flowerList",list);
        return "forward:../index.jsp";
    }

    //商品浏览页
    @RequestMapping("/flowerShop")
    public String queryFlowers(Model model) {
        List<Flower> list = flowerService.getList();
        //把list传到jsp
        model.addAttribute("flowerList",list);
        return "forward:../flowerShop.jsp";
    }

    //按关键字查找
    @RequestMapping("/flowerSort")
    public String queryByKey(String key,Model model){
        System.out.println(key);
        if (key != null){
            List<Flower> list = flowerService.getByKeys(key);
            model.addAttribute("flowerList",list);
            return "forward:../flowerShop.jsp";
        }
        return queryFlowers(model);
    }

    //商品详细页+评论显示
    @RequestMapping(value = "/{id}/detailFlower",method =
RequestMethod.GET)
    public String detailFlower(@PathVariable Integer id, HttpSession
session,Model model) {
```

```
        System.out.println(id);
        //flower数据
        Flower bean = flowerService.getById(id);
        session.setAttribute("flower",bean);
        //session.getAttribute("flower");
        //评价
        List<Comment> list = commentService.getByFlowerId(id);
        if(list!=null){
            model.addAttribute("commentList",list);
        }
        return "../../flowersDetails";
    }

}
```

CommentController.java

```
package cn.edu.lingnan.controller.before;

import cn.edu.lingnan.pojo.Comment;
import cn.edu.lingnan.pojo.Orderform;
import cn.edu.lingnan.service.CommentService;
import cn.edu.lingnan.service.OrderformService;
import cn.edu.lingnan.util.MyUtil;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;

import javax.servlet.http.HttpSession;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.Date;

@Controller
@RequestMapping("/comment")
public class CommentController {

    @Autowired
    CommentService commentService;

    //添加评论
    @RequestMapping(value="/flower")//method = RequestMethod.POST
    public String addComment(Comment comment, HttpSession session) throws
ParseException {
        System.out.println(comment+"comment");
        comment.setUserId(MyUtil.getUserId(session));
        System.out.println(MyUtil.getUserId(session));
        comment.setLoginName(MyUtil.getAccount(session));
        comment.setFloId(MyUtil.getFlowerId(session));
        //String date = new SimpleDateFormat("yyyy-MM-dd
HH:mm:ss").format(new Date());
        SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
        comment.setDate( sdf.parse(sdf.format( new Date() )) );
        commentService.addComment(comment);
```

```
        return "../../index1";
    }

}
```

cn.edu.lingnan.interceptor

AdminLoginIntercreptor.java

```
package cn.edu.lingnan.interceptor;

import org.springframework.web.servlet.HandlerInterceptor;
import org.springframework.web.servlet.ModelAndView;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class AdminLoginInterceptor implements HandlerInterceptor {

    @Override//请求拦截
    public boolean preHandle(HttpServletRequest request,
HttpServletResponse response, Object handler) throws Exception {
        //获取请求地址
        StringBuffer url = request.getRequestURL();
        if( url.toString().indexOf("/flowerHome")>= 0 ||
                url.toString().indexOf("/flowerShop")>= 0 ||
                url.toString().indexOf("/flowerSort")>= 0 ||
                url.toString().indexOf("/detailFlower")>= 0 ||
                url.toString().endsWith("/admin/login") ||
                url.toString().endsWith("/tologin")
        ){
            return true;
        } else {
            Object objA = request.getSession().getAttribute("loginAdmin");
            String path = request.getContextPath();
            if (objA == null){
                request.setAttribute("msg","登录后进行操作");
                response.sendRedirect(path+"/tologin");
            } else {
                return true;
            }
        }
        return false;
    }

    @Override//请求通过
    public void postHandle(HttpServletRequest request, HttpServletResponse
response, Object handler, ModelAndView modelAndView) throws Exception {
        System.out.println("postHandle");
    }

    @Override//请求完成，页面渲染
    public void afterCompletion(HttpServletRequest request,
HttpServletResponse response, Object handler, Exception ex) throws
Exception {
```

```
        System.out.println("afterCompletion");
    }

}
```

UserLoginInterceptor.java

```
package cn.edu.lingnan.interceptor;

import org.springframework.web.servlet.HandlerInterceptor;
import org.springframework.web.servlet.ModelAndView;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class UserLoginInterceptor implements HandlerInterceptor {

    @Override//请求拦截
    public boolean preHandle(HttpServletRequest request,
HttpServletResponse response, Object handler) throws Exception {
        //获取请求地址
        StringBuffer url = request.getRequestURL();
        if( url.toString().indexOf("/flowerHome")>= 0 ||
            url.toString().indexOf("/flowerShop")>= 0 ||
            url.toString().indexOf("/flowerSort")>= 0 ||
            url.toString().endsWith("/detailFlower") ||
            url.toString().endsWith("/user/login") ||
            url.toString().endsWith("/user/reg")
        ){
            return true;
        } else {
            Object objU = request.getSession().getAttribute("loginUser");
            if (objU == null){
                request.setAttribute("msg","登录后进行操作");

 request.getRequestDispatcher("/login.jsp").forward(request,response);
            } else {
                return true;
            }
        }
        return false;
    }

    @Override//请求通过
    public void postHandle(HttpServletRequest request, HttpServletResponse
response, Object handler, ModelAndView modelAndView) throws Exception {
        System.out.println("postHandle");
    }

    @Override//请求完成，页面渲染
    public void afterCompletion(HttpServletRequest request,
HttpServletResponse response, Object handler, Exception ex) throws
Exception {
        System.out.println("afterCompletion");
    }

```

```
}
```

cn.edu.lingnan.mapper

CommentMapper.java

```java
package cn.edu.lingnan.mapper;

import cn.edu.lingnan.pojo.Comment;
import org.apache.ibatis.annotations.Mapper;
import org.apache.ibatis.annotations.Param;
import org.springframework.stereotype.Repository;

import java.util.List;

@Mapper
@Repository
public interface CommentMapper {

    List<Comment> getList();

    List<Comment> getByKeys(String key);

    List<Comment> getByBean(Comment bean);

    List<Comment> getByUserId(@Param("userId") Integer id);

    List<Comment> getByOrderId(@Param("orderId") Integer id);

    List<Comment> getByFlowerId(@Param("floId") Integer id);

    Comment getById(Integer id);

    boolean addComment(Comment bean);

    boolean deleteById(Integer id);

    boolean updateById(Comment bean);

}
```

CommentMapper.xml

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

    <select id="getList" resultType="Comment">
        select * from comment
    </select>

```

```
    <select id="getByUserId" resultType="Comment">
        select * from comment where user_id = #{userId}
    </select>

    <select id="getByOrderId" resultType="Comment">
        select * from comment where order_id = #{orderId}
    </select>

    <select id="getByFlowerId" resultType="Comment">
        select * from comment where flo_id = #{floId}
    </select>

    <select id="getByBean" resultType="Comment">
        select * from comment
        <where>
            <if test="id!=null">
                and id = #{id}
            </if>
            <if test="userId!=null">
                and user_id = #{userId}
            </if>
            <if test="loginName!=null">
                and login_name = #{loginName}
            </if>
            <if test="floId!=null">
                and flo_id = #{floId}
            </if>
            <if test="orderId!=null">
                and order_id = #{orderId}
            </if>
        </where>
    </select>

    <select id="getByKeys" resultType="Comment">
        select * from comment where concat
        <trim prefix="(" suffix=")" suffixOverrides=",">
            <if test="loginName!=null">
                login_name,
            </if>
            <if test="comment!=null">
                comment,
            </if>
        </trim>
        like '%${keys}%'
    </select>

    <insert id="addComment">
        insert into comment
        <trim prefix="(" suffix=")" suffixOverrides=",">
            <if test="id!=null">
                id,
            </if>
            <if test="userId!=null">
                user_id,
            </if>
            <if test="loginName!=null">
                login_name,
            </if>
```

```
            <if test="floId!=null">
                flo_id,
            </if>
            <if test="orderId!=null">
                order_id,
            </if>
            <if test="date!=null">
                date,
            </if>
            <if test="comment!=null">
                comment,
            </if>
        </trim>
        values
        <trim prefix="(" suffix=")" suffixOverrides=",">
            <if test="id!=null">
                #{id},
            </if>
            <if test="userId!=null">
                #{userId},
            </if>
            <if test="loginName!=null">
                #{loginName},
            </if>
            <if test="floId!=null">
                #{floId},
            </if>
            <if test="orderId!=null">
                #{orderId},
            </if>
            <if test="date!=null">
                #{date},
            </if>
            <if test="comment!=null">
                #{comment},
            </if>
        </trim>
    </insert>

    <delete id="deleteById">
        delete from comment where id = #{id}
    </delete>

    <update id="updateById">
        update comment set
        <trim suffixOverrides=",">
            <if test="name!=null">
                user_id=#{userId},
            </if>
            <if test="loginName!=null">
                login_name=#{loginName},
            </if>
            <if test="floId!=null">
                flo_id=#{floId},
            </if>
            <if test="orderId!=null">
                order_id=#{orderId},
            </if>
```

```
            <if test="date!=null">
                date=#{date},
            </if>
            <if test="comment!=null">
                comment=#{comment},
            </if>
        </trim>
        where id = #{id}
    </update>

</mapper>
```

FlowerMapper.java

```java
package cn.edu.lingnan.mapper;

import cn.edu.lingnan.pojo.Flower;
import org.apache.ibatis.annotations.Mapper;
import org.apache.ibatis.annotations.Param;
import org.springframework.stereotype.Repository;

import java.util.List;

@Mapper
@Repository
public interface FlowerMapper {

    Flower getByBean(Flower bean);

    List<Flower> getByName(String name);

    Flower selectByName(String name);

    Flower getById(Integer id);

    List<Flower> getByKeys(String key);

    List<Flower> getList();

    boolean insertFlower(Flower bean);

    boolean deleteById(Integer id);

    boolean updateById(Flower bean);

}
```

FlowerMapper.xml

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

```

```
    <select id="getByBean" resultType="Flower">
        select * from flower
        <where>
            <if test="id!=null">
                and id = #{id}
            </if>
            <if test="name!=null">
                and name like '%${name}%'
            </if>
            <if test="type!=null">
                and type like '%${type}%'
            </if>
        </where>
    </select>

    <select id="getByName" resultType="Flower">
        select * from flower where name like '%${name}%'
    </select>

    <select id="selectByName" resultType="Flower">
        select * from flower where name = #{name}
    </select>

    <select id="getById" resultType="Flower">
        select * from flower where id = #{id}
    </select>

    <select id="getByKeys" resultType="Flower">
        select * from flower where concat
        <trim prefix="(" suffix=")" suffixOverrides=",">
            <if test="name!=null">
                name,
            </if>
            <if test="type!=null">
                type,
            </if>
            <if test="floMean!=null">
                flo_mean,
            </if>
        </trim>
        like '%${keys}%'
    </select>

    <select id="getList" resultType="Flower">
        select * from flower
    </select>

    <insert id="insertFlower">
        insert into flower
        <trim prefix="(" suffix=")" suffixOverrides=",">
            <if test="id!=null">
                id,
            </if>
            <if test="name!=null">
                name,
            </if>
            <if test="type!=null">
                type,
```

```
            </if>
            <if test="stock!=null">
                stock,
            </if>
            <if test="price!=null">
                price,
            </if>
            <if test="floMean!=null">
                flo_mean,
            </if>
            <if test="photo!=null">
                photo,
            </if>
        </trim>
            values
        <trim prefix="(" suffix=")" suffixOverrides=",">
            <if test="id!=null">
                #{id},
            </if>
            <if test="name!=null">
                #{name},
            </if>
            <if test="type!=null">
                #{type},
            </if>
            <if test="stock!=null">
                #{stock},
            </if>
            <if test="price!=null">
                #{price},
            </if>
            <if test="floMean!=null">
                #{floMean},
            </if>
            <if test="photo!=null">
                #{photo},
            </if>
        </trim>
    </insert>

    <delete id="deleteById">
        delete from flower where id = #{id}
    </delete>

    <update id="updateById">
        update flower set
        <trim suffixOverrides=",">
            <if test="name!=null">
                name=#{name},
            </if>
            <if test="type!=null">
                type=#{type},
            </if>
            <if test="stock!=null">
                stock=#{stock},
            </if>
            <if test="price!=null">
                price=#{price},
```

```
            </if>
            <if test="floMean!=null">
                flo_mean=#{floMean},
            </if>
            <if test="photo!=null">
                photo=#{photo},
            </if>
        </trim>
        where id = #{id}
    </update>

</mapper>
```

cn.edu.lingnan.pojo

Comment.java

```
package cn.edu.lingnan.pojo;

import org.springframework.stereotype.Component;

import java.util.Date;

@Component
public class Comment {

    private Integer id;
    private Integer userId;
    private String loginName;
    private Integer floId;
    private Integer orderId;
    private Date date;//评论时间
    private String comment;

    public Comment() {
    }

    public Comment(Integer id, Integer userId, String loginName, Integer floId, Integer orderId, Date date, String comment) {
        this.id = id;
        this.userId = userId;
        this.loginName = loginName;
        this.floId = floId;
        this.orderId = orderId;
        this.date = date;
        this.comment = comment;
    }

    public Integer getId() {
        return id;
    }

    public void setId(Integer id) {
        this.id = id;
    }
```

```
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 +
```

```
                ", comment='" + comment + '\'' +
                '}';
    }

}
```

Flower.java

```
package cn.edu.lingnan.pojo;

import org.springframework.stereotype.Component;

@Component
public class Flower {
//商品
    private Integer id;
    private String name;
    private String type;
    private Integer stock;
    private Float price;
    private String floMean;
    private String photo;

    public Flower() {
    }

    public Flower(Integer id, String name, String type, Integer stock,
Float price, String floMean, String photo) {
        this.id = id;
        this.name = name;
        this.type = type;
        this.stock = stock;
        this.price = price;
        this.floMean = floMean;
        this.photo = photo;
    }

    public Integer getId() {
        return id;
    }

    public void setId(Integer id) {
        this.id = id;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getType() {
        return type;
    }
```

```
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

```java
package cn.edu.lingnan.service;

import cn.edu.lingnan.pojo.Comment;
import org.apache.ibatis.annotations.Param;

import java.util.List;

public interface CommentService {

    List<Comment> getList();

    List<Comment> getByKeys(String key);

    List<Comment> getByBean(Comment bean);

    List<Comment> getByUserId(Integer id);

    List<Comment> getByOrderId(Integer id);

    List<Comment> getByFlowerId(Integer id);

    Comment getById(Integer id);

    boolean addComment(Comment bean);

    boolean deleteById(Integer id);

    boolean updateById(Comment bean);

}
```

CommentServiceImpl.java

```java
package cn.edu.lingnan.service.impl;


import cn.edu.lingnan.mapper.CommentMapper;
import cn.edu.lingnan.pojo.Comment;
import cn.edu.lingnan.service.CommentService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import java.util.List;

@Service
public class CommentServiceImpl implements CommentService {

    @Autowired
    CommentMapper commentMapper;

    @Override
    public List<Comment> getList() {
        return commentMapper.getList();
    }

    @Override
```

```
    public List<Comment> getByKeys(String key) {
        return commentMapper.getByKeys(key);
    }

    @Override
    public List<Comment> getByBean(Comment bean) {
        return null;
    }

    @Override
    public List<Comment> getByUserId(Integer id) {
        return commentMapper.getByUserId(id);
    }

    @Override
    public List<Comment> getByOrderId(Integer id) {
        return commentMapper.getByOrderId(id);
    }

    @Override
    public List<Comment> getByFlowerId(Integer id) {
        return commentMapper.getByFlowerId(id);
    }

    @Override
    public Comment getById(Integer id) {
        return commentMapper.getById(id);
    }

    @Override
    public boolean addComment(Comment bean) {
        return commentMapper.addComment(bean);
    }

    @Override
    public boolean deleteById(Integer id) {
        return commentMapper.deleteById(id);
    }

    @Override
    public boolean updateById(Comment bean) {
        return updateById(bean);
    }

}
```

FlowerService.java

```
package cn.edu.lingnan.service;

import cn.edu.lingnan.pojo.Flower;

import java.util.List;

public interface FlowerService {

```

```
    Flower getByBean(Flower bean);

    List<Flower> getByName(String name);

    Flower selectByName(String name);

    Flower getById(Integer id);

    List<Flower> getByKeys(String key);

    List<Flower> getList();

    boolean insertFlower(Flower bean);

    boolean deleteById(Integer id);

    boolean updateById(Flower bean);

}
```

FlowerServiceImpl.java

```
package cn.edu.lingnan.service.impl;

import cn.edu.lingnan.mapper.FlowerMapper;
import cn.edu.lingnan.pojo.Flower;
import cn.edu.lingnan.service.FlowerService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import java.util.List;

@Service
public class FlowerServiceImpl implements FlowerService {

    @Autowired
    private FlowerMapper flowerMapper;

    @Override
    public Flower getByBean(Flower bean) {
        return flowerMapper.getByBean(bean);
    }

    @Override
    public List<Flower> getByName(String name) {
        return flowerMapper.getByName(name);
    }

    @Override
    public Flower selectByName(String name) {
        return flowerMapper.selectByName(name);
    }

    @Override
    public Flower getById(Integer id) {
        return flowerMapper.getById(id);
```

```
    }

    @Override
    public List<Flower> getByKeys(String key) {
        return flowerMapper.getByKeys(key);
    }

    @Override
    public List<Flower> getList() {
        return flowerMapper.getList();
    }

    @Override
    public boolean insertFlower(Flower bean) {
        return flowerMapper.insertFlower(bean);
    }

    @Override
    public boolean deleteById(Integer id) {
        return flowerMapper.deleteById(id);
    }

    @Override
    public boolean updateById(Flower bean) {
        return flowerMapper.updateById(bean);
    }

}
```

cn.edu.lingnan.util

MyUtil.java

```
package cn.edu.lingnan.util;
import cn.edu.lingnan.pojo.Admin;
import cn.edu.lingnan.pojo.Flower;
import cn.edu.lingnan.pojo.User;

import java.text.SimpleDateFormat;
import java.util.Date;
import javax.servlet.http.HttpSession;
public class MyUtil {
    /**
     * 获得时间字符串
     */
    public static String getStringID(){
        String id=null;
        Date date=new Date();
        SimpleDateFormat sdf=new SimpleDateFormat("yyyyMMddHHmmssSSS");
        id=sdf.format(date);
        return id;
    }

    /**
     * 获得用户ID
```

```
     */
    public static Integer getUserId(HttpSession session) {
        User user = (User)session.getAttribute("loginUser");
        return user.getId();
    }

    /**
     * 获得用户登录名
     */
    public static String getAccount(HttpSession session) {
        User user = (User)session.getAttribute("loginUser");
        return user.getAccount();
    }

    /**
     * 获得AdminName
     */
    public static String getAdminName(HttpSession session) {
        Admin admin = (Admin)session.getAttribute("loginAdmin");
        return admin.getName();
    }

    /**
     * 获得商品ID
     */
    public static Integer getFlowerId(HttpSession session) {
        Flower flower = (Flower)session.getAttribute("flower");
        return flower.getId();
    }

    /**
     * 获得商品名
     */
    public static String getFlowerName(HttpSession session) {
        Flower flower = (Flower)session.getAttribute("flower");
        return flower.getName();
    }

}
```

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

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       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/mvc
```

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

```
            <bean class="cn.edu.lingnan.interceptor.AdminLoginInterceptor" />
        </mvc:interceptor>
    </mvc:interceptors>

</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>${flower.getFloMean()}</td>
            <td><img src="${ctx}/${flower.getPhoto()}" style="..."></td>
            <td>
                <a class="layui-btn layui-btn-xs" lay-event="edit" href="${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="${sessionScope.flower.photo!=null}">
        <img src="${ctx}/${sessionScope.flower.photo}" width="50" height="50">
    </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 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="${sessionScope.flower.photo!=null}">
            <img src="${ctx}/${sessionScope.flower.photo}" width="50" height="50">
        </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);
```

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()}">--%>
                    <img class="pri-img" src="${ctx}/${flower.getPhoto()}" alt="product" width="270" height="270">
                    <img class="sec-img" src="${ctx}/${flower.getPhoto()}" alt="product" width="270" height="270">
                </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="${ctx}/flower/{name}/detailFlower">
            <img class="pri-img" src="${ctx}/${flower.getPhoto()}" alt="product" width="270" height="270">
            <img class="sec-img" src="${ctx}/${flower.getPhoto()}" alt="product" width="270" height="270">
        </a>
    </figure>
    <div class="product-content-list">
        <h5 class="product-name"><a href="${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>
```

flowersDetail.jsp

```
                <img src="${ctx}/${sessionScope.flower.photo}" alt="product-details" />
            </div>
        </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>
```

```
        <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/${flower.getId()}/detailFlower">
                <img class="pri-img" src="${ctx}/${flower.getPhoto()}" alt="product">
                <img class="sec-img" src="${ctx}/${flower.getPhoto()}" alt="product">
            </a>
            <div class="product-badge">
                <div class="product-label new">
                    <span>new</span>
                </div>
                <div class="product-label discount">
                    <span>10%</span>
                </div>
            </div>
            <div class="button-group">
                <a href="#" data-toggle="tooltip" data-placement="left" title="Add to wishlist"><i class="lnr lnr-heart"></i></a
                <a href="#" data-toggle="modal" data-target="#quick_view"><span data-toggle="tooltip" data-placement="left" titl
                <a href="#" data-toggle="tooltip" data-placement="left" title="Add to Cart"><i class="lnr lnr-cart"></i></a>
            </div>
        </figure>
        <div class="product-caption">
            <p class="product-name">
                <a href="${ctx}/flower/{name}/detailFlower">${flower.getName()}</a>
            </p>
```

帮忙修改的不放了，就那一两行