

day43-项目项目第三天

学习目标

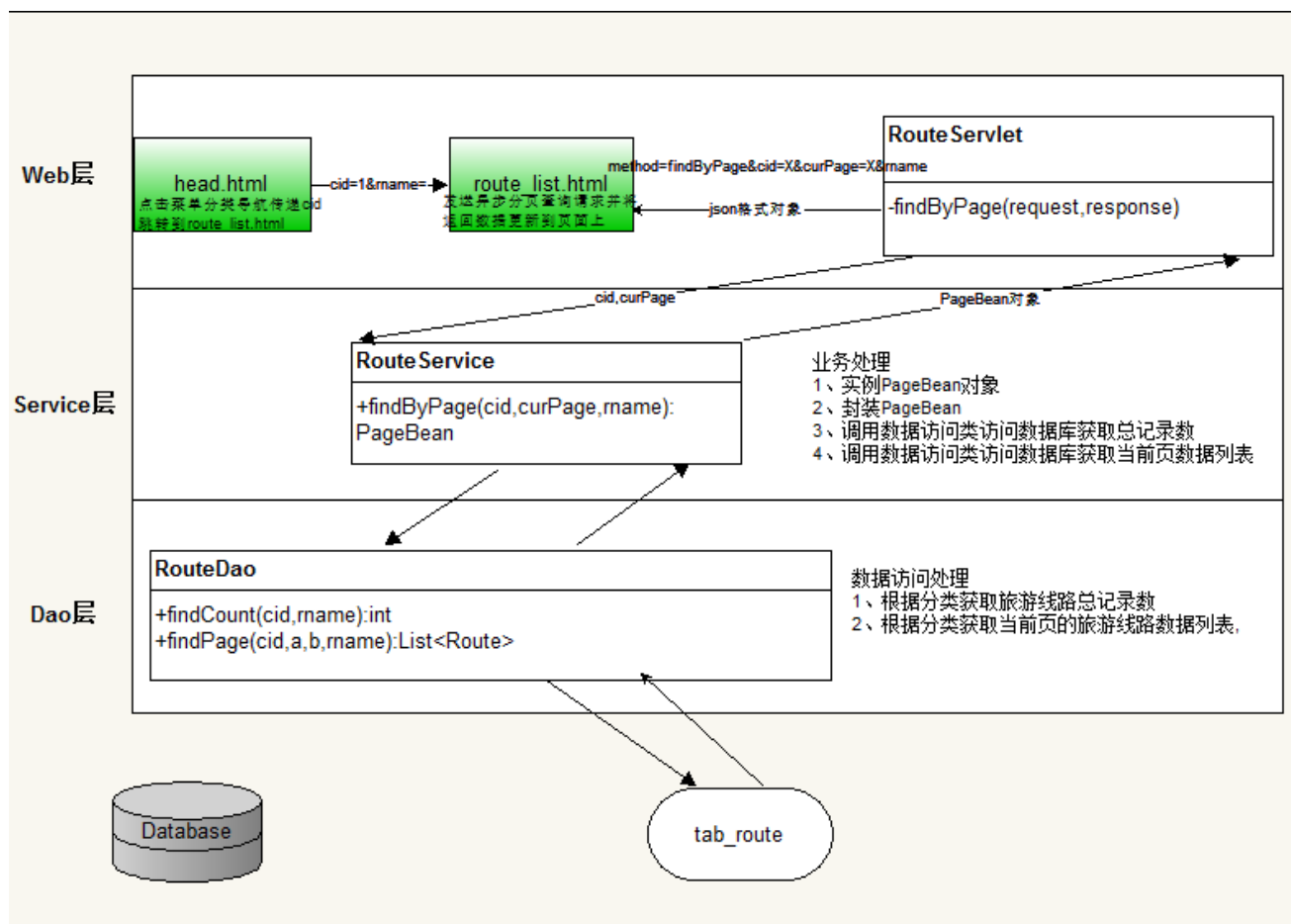
- 1. 能够完成头部搜索旅游线路分页展现数据案例
- 2. 能够完成查看旅游线路详情案例
- 3. 能够完成添加收藏案例
- 4. 能够进行面向接口编程

案例一 头部搜索旅游线路分页展现数据

一、案例需求



二、思路分析

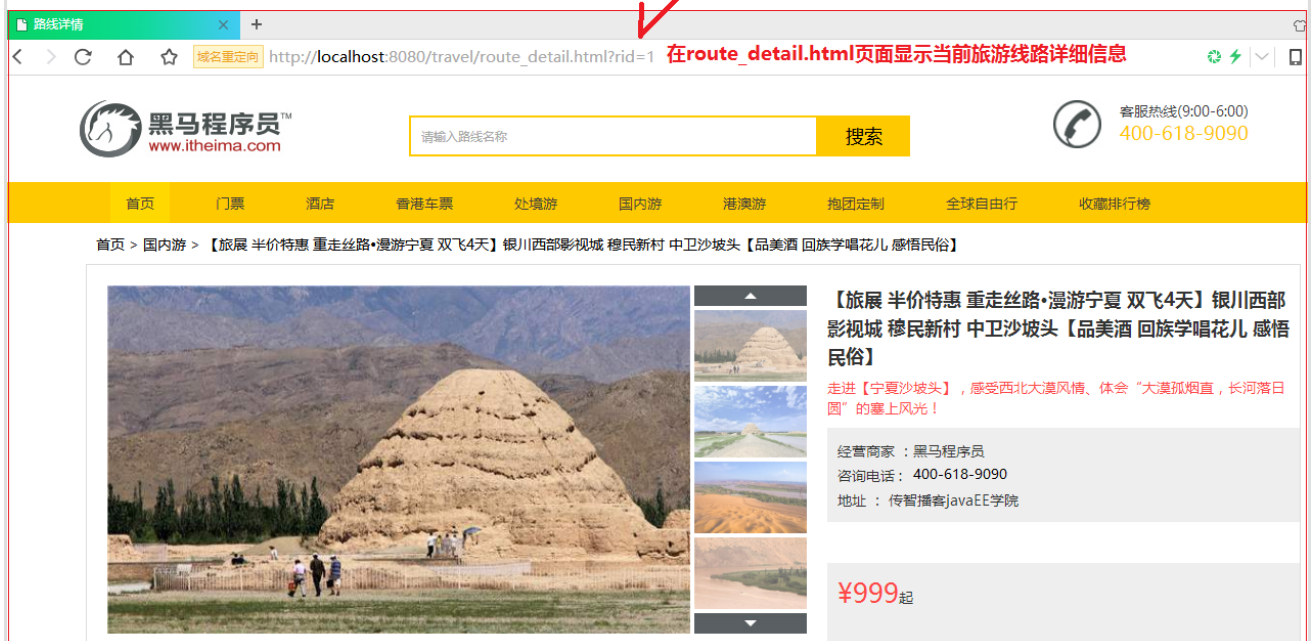


三,代码实现

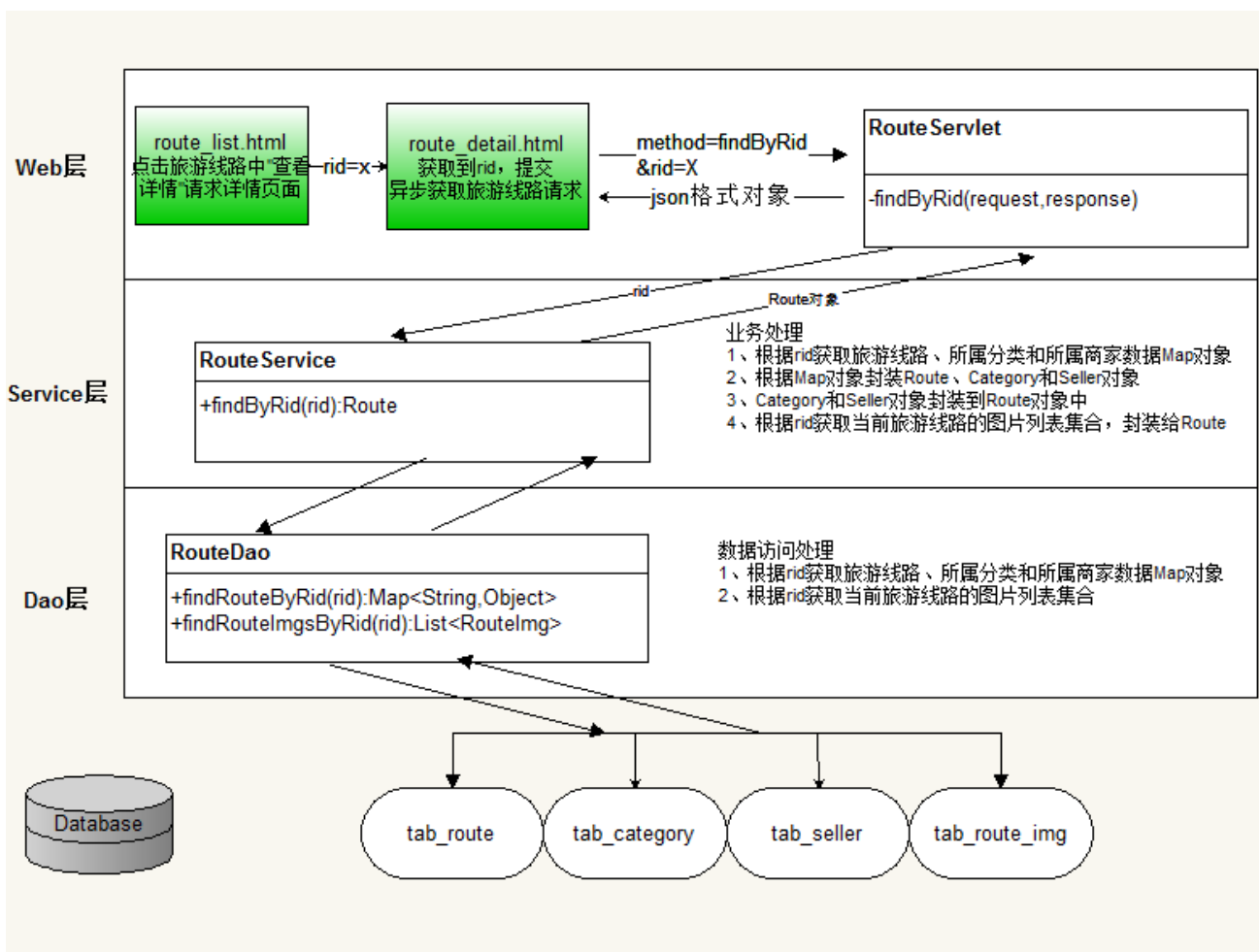
案例二：查看旅游线路详情

一,案例需求

点击route_list.html搜索列表中旅游线路的“查看详情”进入route_detail.html旅游线路详情页面并显示相关数据



二, 思路分析



```

1 |
2 | "data": {
3 |   "count": 169,
4 |   "curPage": 1,
5 |   "curSize": 8,
6 |   "list": [
7 |     {
8 |       "cid": 5,
9 |       "count": 990,
10 |      "isThemeTour": "0",
11 |      "price": 1799.0,
12 |      "rdate": "2018-02-09 01:13:17",
13 |      "rflag": "1",
14 |      "rid": 2,
15 |      "rimage": "img/product/small/m34866f055de8630e94e25c40f277a79ba.jpg",
16 |      "rname": "【官网专享 立减7500 张家界天门山+大峡谷+凤凰古城+玻璃栈道+",
17 |      "routeIntroduce": "官网专线, 顶级品质! 全程超豪华住宿, 2晚入住超豪华",
18 |      "sid": 1,
19 |      "sourceId": "22066"
20 |     },
21 |     {
22 |       "cid": 5,
23 |       "count": 0,
24 |       "isThemeTour": "0",
25 |       "price": 1099.0,
26 |       "rdate": "2018-02-09 01:13:17",
  
```

三,代码实现

• Header

点击导航条进入分页查询后,再填写输入框进行搜索,因为首页没有cid 搜索页在Header.html内,而Header.html贯穿整个页面,当输入关键字搜索后,由Header.html跳转到route_list.html并携带cid和搜索关键字searchkey

```

<script>
function search() {
var searchkey=$("#search_input").val();
var cid=getParameter("cid");
location.href="route_list.html?cid="+cid+"&searchkey="+searchkey+"&curPage=1";
}
</script>

```

• route_list.html

通过js插件获取当前页面的关键字searchkey,模块cid,当前页码curPage;注意浏览器传输给浏览器需要解码

在原有分页查询基础上增加searchkey,post请求给服务器,浏览器接受结果集处理方式不用变,因为只变了查询条件

当搜索后点击下一页时需要作判断,如果关键字searchkey等于null字符串则超链接中不添加searchkey条件直接搜索全部,如果关键字searchkey不等于null字符串,则超链接增加searchkey条件实现分页查询

同时运用到了else if的特殊能力if后会覆盖else if中的内容。

例如if(a>10)else if(a>6)相当于if(a>10)else if(a<=10&&a>6)

当点击查看详情超链接后,将rid结果附带到route_detail.html页面

```

<script>
//根据js插件调用方法
var searchkey=getParameter("searchkey");
//这里searchkey由浏览器传输给浏览器需要解码
if(searchkey!=null&&searchkey!=""){
    searchkey = decodeURI(searchkey);
}
var cid=getParameter("cid");
var curPage=getParameter("curPage");
$.post("rout", {method: "perpage", cid: cid, curPage: curPage, searchkey: searchkey}, function (result)
{
if(result.flag){
var data=result.data;
var list=data.list;
var listhtml="";
$(list).each(function (i,obj) {
    listhtml+="- >\n" +
        "
        <div class=\"img\"><img src=\""+obj.rimage+"\" alt=\"\">
</div>\n" +
        "
        <div class=\"text1\">\n" +
        "
        <p>"+obj.rname+"</p>\n" +
        "
        <br/>\n" +
        "
        <p>"+obj.routeIntroduce+"</p>\n" +
        "
        </div>\n" +
        "
        <div class=\"price\">\n" +
        "
        <p class=\"price_num\">\n" +
        "
        <span>&yen;</span>\n" +
        "
        <span>"+obj.price+"</span>\n" +
        "
        <span>起</span>\n" +
        "
        </p>\n" +

```

```

        "                                <p><a href=\"route_detail.html?rid="+obj.rid+"\">查看详
情</a></p>\n" +
        "                                </div>\n" +
        "                                </li>";
    });

$("#listUIId").html(listhtml);
$("#sumPageId").html(data.sumPage);
$("#countId").html(data.count);
var pagehtml="";
if(data.curPage>1&&searchkey!="&&searchkey!="null"&&searchkey!=null){
    pagehtml+="- <a href=\"route_list.html?cid="+cid+"&searchkey="+searchkey+"&curPage=1\">首
页</a></li>\n" +
        "                                <li class=\"threeword\"><a href=\"route_list.html?
cid="+cid+"&searchkey="+searchkey+"&curPage="+data.curPage-1+"\">上一页</a></li>";
}else if(data.curPage>1){
    pagehtml+="- <a href=\"route_list.html?cid="+cid+"&curPage=1\">首页</a></li>\n" +
        "                                <li class=\"threeword\"><a href=\"route_list.html?
cid="+cid+"&curPage="+data.curPage-1+"\">上一页</a></li>";
}
var begin;
var end;
if(data.sumPage<=10){
    begin=1;
    end=data.sumPage;
}else{
    begin=data.curPage-5;
    end=data.curPage+4;
    if(begin<1){
        begin=1;
        end=10;
    }
    if(end>data.sumPage){
        begin=data.sumPage-9;
        end=data.sumPage;
    }
}

for(var i=begin;i<=end;i++){
    if(i==data.curPage){
        pagehtml+="-

```

```

pagehtml+="                                <li class=\"threeword\"><a href=\"route_list.html?
cid="+cid+"&searchkey="+searchkey+"&curPage="+data.curPage+1)+"\">下一页</a></li>\n" +
                                <li class=\"threeword\"><a href=\"route_list.html?
cid="+cid+"&searchkey="+searchkey+"&curPage="+data.sumPage+"\">末页</a></li>";
}else if(data.curPage<data.sumPage){
    pagehtml+="                                <li class=\"threeword\"><a href=\"route_list.html?
cid="+cid+"&curPage="+data.curPage+1)+"\">下一页</a></li>\n" +
                                <li class=\"threeword\"><a href=\"route_list.html?
cid="+cid+"&curPage="+data.sumPage+"\">末页</a></li>";
}
$("#pageUId").html(pagehtml);
}else{
    alert(result.msg);
}
}, "json")
</script>

```

- **route detail.html**

展示详情通过ajax携带rid请求服务器，得到结果数据后将大图和小图填充页面

点击收藏通过ajax携带rid请求服务器，首先查询的结果flag没异常再判断数据是否可编辑，如果可编辑，则将隐藏按钮的功能移除

给按钮添加一个点击事件，当点击按钮后再发送ajax请求给服务器，判断查询是否有异常，如果没有异常再判断是否为登录状态，如果不是登录状态直接跳转到登录页面，如果是登录状态点击后将原按钮隐藏变为不可点击，点击功能取消，同时将响应数据已收藏覆盖显示到页面

```
<script>
    var rid =getParameter("rid");
    $.post("favorite",{method:"isEdit",rid:rid},function (result) {
        if(result.flag){
            if(result.data.isEdit){
                $(".collect a").removeClass("already");
                $(".collect a").removeAttr("disabled");
            }
        }else{
            alert(result.msg);
        }
    }, "json");

function adddeclard() {
    $.post("favorite",{method:"addetailed",rid:rid},function (result) {
        if(result.flag){
            if(result.data.isLogin){
                $(".collect a").addClass("already");
                $(".collect a").Attr("disabled");
                $(".collect a").removeAttr("onclick");
                $("#collectSpanId").html("已收藏"+result.data.count+"次");
            }else{

                location.href="login.html";
            }
        }else{

```

```

        alert(result.msg);
    }
    }, "json");
};

$.post("routn",{method:"detailed",rid:rid},function (result) {
    if(result.flag){
        var data=result.data;
        $(".bread_box").html("<a href='\"/\"'>首页</a>\n" +
            "        <span> &gt;</span>\n" +
            "        <a href='\"#\"'+data.category.cname+'\"><a><span> &gt;</span>\n" +
            "        <a href='\"#\"'+data.rname+'\"></a>");
        $(".pros_title").html(data.rname);
        $(".hot").html(data.routeIntroduce);
        $(".pros_other").html("                <p>经营商家  : "+data.seller.sname+"</p>\n" +
            "                <p>咨询电话 : "+data.seller.consphone+"</p>\n" +
            "                <p>地址 : "+data.seller.address+"</p> ");
        $(".price strong").html("&#36;"+data.price+"");
        $(".#collectSpanId").html("&#2013;已收藏"+data.count+"次");
        var imglisthtml="&lt;dt>\n" +
            "                &lt;img alt='\"\"\" class='\"big_img\"\"\"
src='\"'+data.routeImgList[0].bigPic+'\">\n" +
            "                &lt;/dt>\n" +
            "                &lt;dd> &lt;a class='\"up_img up_img_disable\"\">&lt;/a>";
        var rountlist=data.routeImgList;
        for(var i=0;i<rountlist.length;i++){
            var rountimg=rountlist[i];
            if(i<4){
                imglisthtml+="&lt;a title='\"\"\" class='\"little_img cur_img\"\"\" +
                    "                data-bigpic='\"'+rountimg.bigPic+'\">\n" +
                    "                &lt;img src='\"'+rountimg.smallPic+'\">\n" +
                    "                &lt;/a>";
            }else{
                imglisthtml+="&lt;a title='\"\"\" class='\"little_img\"\"\" +
                    "                data-bigpic='\"'+rountimg.bigPic+'\">\n" +
                    "                style='\"display:none;\">\n" +
                    "                &lt;img src='\"'+rountimg.smallPic+'\">\n" +
                    "                &lt;/a>";
            }
        }
    }

    imglisthtml+="&lt;a class='\"down_img down_img_disable\"\"\" style='\"margin-bottom: 0;\">&lt;/a>\n" +
        "                &lt;/dd>";
    $(".prosum_left").html(imglisthtml);
    imgGo();
    }else{
        alert(result.msg);
    }

    }, "json");
}
</script>

```


• RountServlet

从浏览器请求数据得到关键字searchkey在原有分页查询基础上增加一个searchkey条件,当不点击搜索时,默认searchkey为null,经过Dao层判断后不影响原来分页结果

为解决首页搜索bug,将cid赋值为0,再判断传入Dao,Dao具有动态添加功能再次判断cid>0不满足条件直接查全部
请请求消息中获取rid , 然后查询数据库将对象响应给浏览器

```
@WebServlet("/rout")
public class RountServlet extends BaseServlet {
    private RountService rountService=new RountService();
    public void selectiontravel(HttpServletRequest request, HttpServletResponse response) throws
    IOException {
        String data=null;
        ResultInfo resultInfo=null;
        ObjectMapper objectMapper=new ObjectMapper();

        try {
            Map<String,List<Route>> map=rountService.selectionall();
            resultInfo=new ResultInfo(true,map,"精选OK");
        } catch (Exception e) {
            e.printStackTrace();
            resultInfo=new ResultInfo(false,null,"精选无数据");
        }finally{
            data=objectMapper.writeValueAsString(resultInfo);
            response.getWriter().print(data);
        }
    }

    public void perpage(HttpServletRequest request, HttpServletResponse response) throws
    IOException {
        String data=null;
        ResultInfo resultInfo=null;
        ObjectMapper objectMapper=new ObjectMapper();
        try {
            String searchkey = request.getParameter("searchkey");
            int cid=0;
            String cidstr = request.getParameter("cid");
            if(cidstr!=null&&"".equals(cidstr)&&"null".equals(cidstr))
            {
                cid= Integer.parseInt(cidstr);
            }
            int curPage = Integer.parseInt(request.getParameter("curPage"));
            PageBean<Route> pageBean=rountService.perPage(cid,curPage,searchkey);
            resultInfo=new ResultInfo(true,pageBean,"查询成功");
        } catch (Exception e) {
            resultInfo=new ResultInfo(false,null,"查询失败");
            e.printStackTrace();
        } finally {
            String perdata = objectMapper.writeValueAsString(resultInfo);
            response.getWriter().print(perdata);
        }
    }
}
```

```

    public void detailed(HttpServletRequest request, HttpServletResponse response) throws
IOException {
        String data=null;
        ResultInfo resultInfo=null;
        ObjectMapper objectMapper=new ObjectMapper();
        try {
            int rid = Integer.parseInt(request.getParameter("rid"));
            Route route = rountService.detailedser(rid);
            resultInfo=new ResultInfo(true,route,"详情正确");
        } catch (Exception e) {
            e.printStackTrace();
            resultInfo=new ResultInfo(false,null,"详情错误");
        } finally {
            String s = objectMapper.writeValueAsString(resultInfo);
            response.getWriter().print(s);
        }
    }
}

```

• RountService

pageBean里面封装了5个参数 总数量,总页码,当前页码,一页的数量,当前页的list数据

总数量和当前页的list数据需要查数据库,其他通过这两个数算出来

查看详情时有几处的数据为不同的表不能一次性封装,需要使用BeanUtil多次封装

```

public class RountService {
    private RouteDao routeDao=new RouteDao();
    public PageBean<Route> perPage(int cid, int curPage,String searchkey)throws Exception {
        int count=routeDao.getCount(cid,searchkey);
        int curSize= Constants.Route_curSize;
        int sumPage;
        if(count%curSize==0){
            sumPage=count/curSize;
        }else{
            sumPage=count/curSize+1;
        }
        List<Route> list;
        int b=curSize;
        int a=b*(curPage-1);
        list=routeDao.perPage(cid,a,b,searchkey);
        PageBean<Route> pageBean=new PageBean<>(sumPage,count,curSize,curPage,list);
        return pageBean;
    }

    public Route detailedser(int rid) throws Exception {
        Route route=new Route();
        Map<String,Object> map=routeDao.detailedddao(rid);
        BeanUtils.populate(route,map);
        Category category=new Category();
        BeanUtils.populate(category,map);
        route.setCategory(category);

        Seller seller = new Seller();
    }
}

```

```

        BeanUtils.populate(seller,map);
        route.setSeller(seller);
        List<RouteImg> list=routeDao.imggrid(rid);
        route.setRouteImgList(list);
        return route;
    }
}

```

• RouteDao

新增参数后,查询数据库需要写严密,判断是否为空,空字符串,是否大于0等

sql语句拼接字符串,list集合存取实现判断功能,动态添加功能

如果关键字为空,则sql语句不添加关键字语句同时查询语句中不会出现关键字参数,走原来的分页查询

```

public class RouteDao {
    private JdbcTemplate jdbcTemplate=new JdbcTemplate(C3P0Util.getDataSource());
    private int count;
    public int getCount(int cid,String searchkey)throws Exception {
        String sql="select count(*) from tab_route where rflag='1'";
        List<Object> list=new ArrayList<>();
        if(cid>0){
            sql+=" and cid=?";
            list.add(cid);
        }
        if(searchkey!=null&&" ".equals(searchkey)){
            sql+=" and rname like ?";
            list.add("%"+searchkey+"%");
        }
        int i = jdbcTemplate.queryForObject(sql,Integer.class,list.toArray());
        return i;
    }
    public Map<String,Object> detaileddao(int rid) throws Exception{
        String sql="select * from tab_route a,tab_seller b,tab_category c where a.cid = c.cid and a.sid= b.sid and a.rflag='1' and a.rid=?";
        return jdbcTemplate.queryForMap(sql,rid);
    }

    public List<Route> perPage(int cid, int a, int b,String searchkey) throws Exception{
        List<Object>para=new ArrayList<>();
        String sql="select * from tab_route where rflag='1'";
        // and cid= ? limit ?,?
        if(cid>0){
            para.add(cid);
            sql+=" and cid=? ";
        }
        if(searchkey!=null&&" ".equals(searchkey)){
            para.add("%"+searchkey+"%");
            sql+=" and rname like ?";
        }
        sql+=" limit ?,?";
        para.add(a);
    }
}

```

```

        para.add(b);

        List<Route> list = jdbcTemplate.query(sql, new BeanPropertyRowMapper<>(Route.class),
para.toArray());
        return list;
    }
}

```

• FavoriteServlet

首先获取请求从Session中获得登录信息,如果未登录,将显示按钮同时封装可编辑标记isEdit为true,表示可点因为要跳转

如果已登陆则根据user的uid和rid查询数据库并作判断,如果查询成功,则不可编辑isEdit设置为false并存到map中,响应给浏览器后,浏览器通过json解析map判断isEdit按钮是否可点击

当用户点击收藏按钮后,首先判断用户是否登陆,如果未登录直接封装map数据isLogin为false响应给浏览器,浏览器判断后,跳转登录页面

如果用户登录了,则直接更新数据库,将收藏次数和用户及商品关系表数据更新,然后将isLogin状态和count封装到map中,响应给浏览器

```

@WebServlet("/favorite")
public class FavoriteServlet extends BaseServlet{
    private FavoriteService favoriteService=new FavoriteService();
    private RountService rountService=new RountService();
    public void isEdit(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
        String data=null;
        ObjectMapper objectMapper=new ObjectMapper();
        ResultInfo resultInfo=null;
        try {
            User user = (User) request.getSession().getAttribute("user");
            Map<String,Object> map=new HashMap<>();
            if(user==null){
                map.put("isEdit",true);
                resultInfo=new ResultInfo(true,map,"未登录");
            }else{
                int rid = Integer.parseInt(request.getParameter("rid"));
                boolean flag=favoriteService.booleanEdit(user.getUid(),rid);
                if(flag){
                    map.put("isEdit",false);
                    resultInfo=new ResultInfo(true,map,"已收藏");
                }else{
                    map.put("isEdit",true);
                    resultInfo=new ResultInfo(true,map,"未收藏");
                }
            }
        } catch (Exception e) {
            e.printStackTrace();
            resultInfo=new ResultInfo(false,null,"服务器异常");
        } finally {
            String s = objectMapper.writeValueAsString(resultInfo);

```

```

        response.getWriter().print(s);
    }
}

public void addetailed(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
    String data=null;
    ObjectMapper objectMapper=new ObjectMapper();
    ResultInfo resultInfo=null;
    Map<String,Object> map=new HashMap<>();
    try {
        int rid = Integer.parseInt(request.getParameter("rid"));
        User user = (User) request.getSession().getAttribute("user");
        if(user==null){
            map.put("isLogin",false);

            resultInfo=new ResultInfo(true,map,"未登录");
        }else{
            favoriteService.addfav(user.getUid(),rid);
            int count = rountService.detailedser(rid).getCount();
            map.put("isLogin",true);
            map.put("count",count);
            resultInfo=new ResultInfo(true,map,"已登录");
        }
    } catch (Exception e) {
        e.printStackTrace();
        resultInfo=new ResultInfo(false,null,"服务器异常");
    } finally {
        data=objectMapper.writeValueAsString(resultInfo);
        System.out.println(data);
        response.getWriter().print(data);
    }
}
}

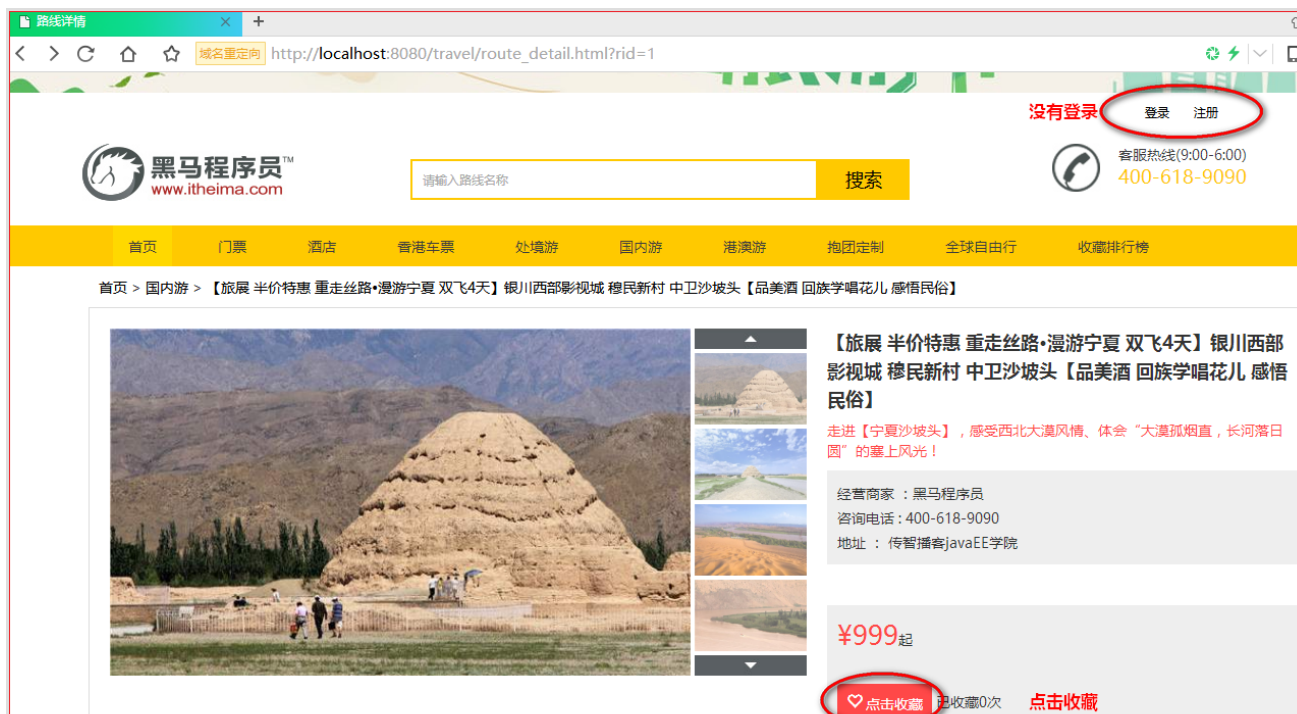
```

案例三-添加收藏

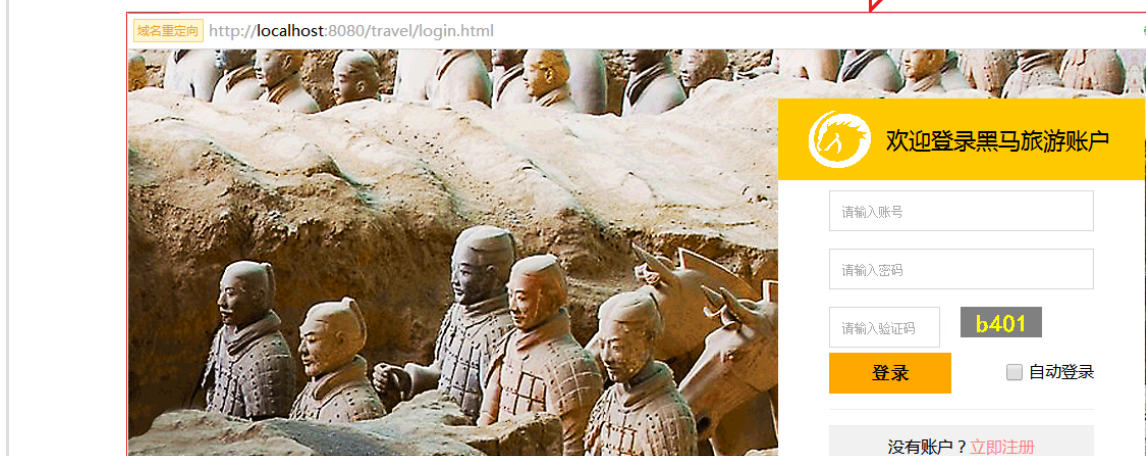
一,案例需求

用户在旅游线路详情页面点击“添加收藏”可以进行收藏该旅游线路，注意只有已经登录的用户才可以收藏当前旅游线路。此功能需要实现2个效果，第1个效果在旅游详情页面加载时根据是否已被收藏显示“点击收藏”是否可编辑；第2个效果是登录用户收藏当前旅游线路成功后的效果。

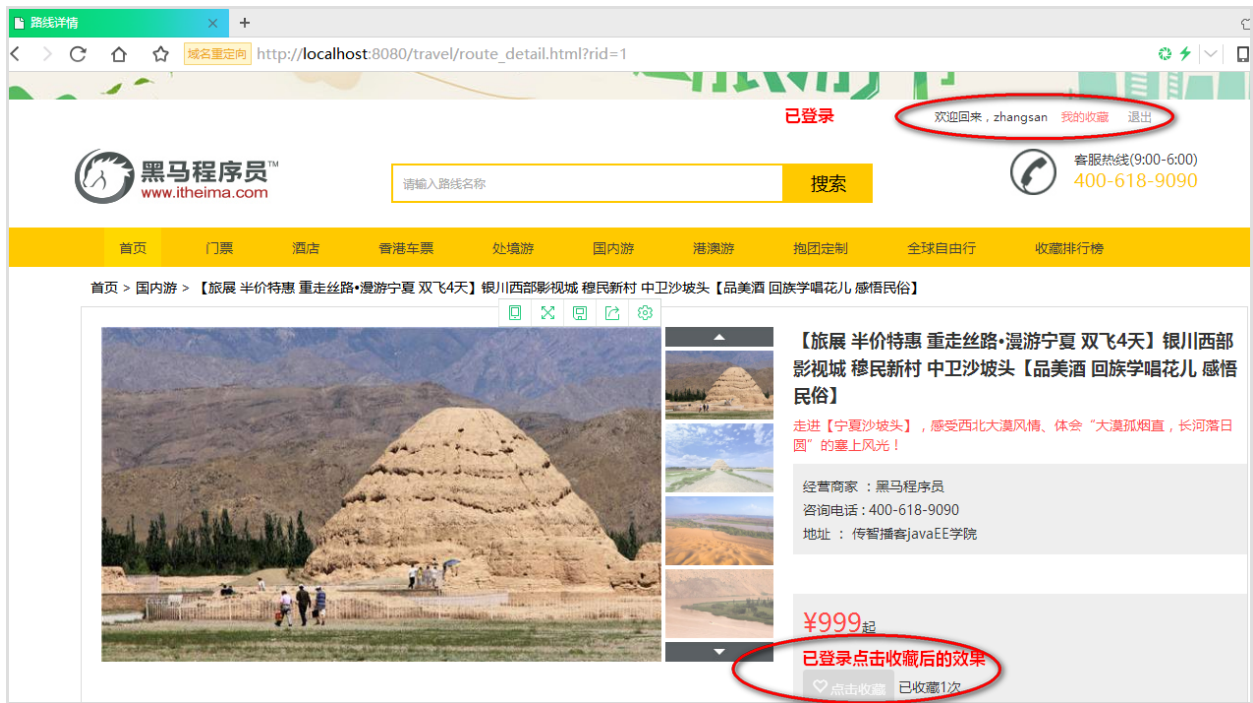
- 没有登录,显示可编辑状态, 点击“点击收藏”跳转到登录页面



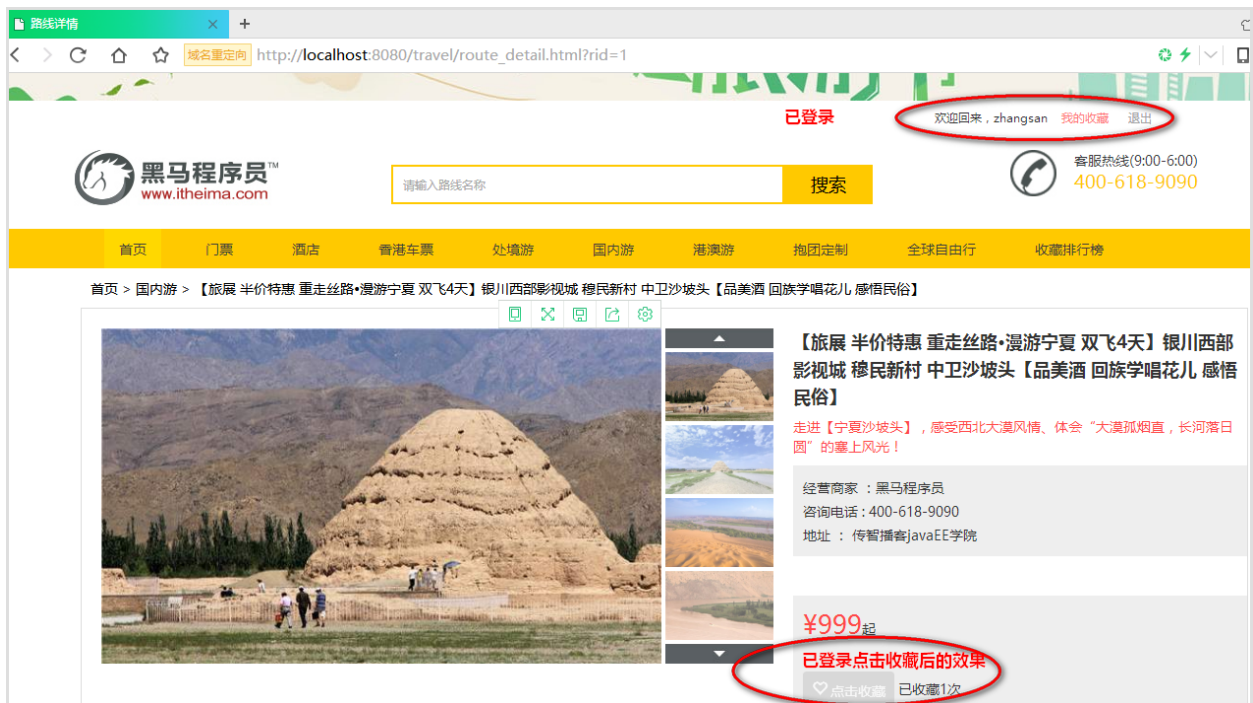
没有登录点击“点击收藏”跳转到登录页面



- 已经登录
 - 已经收藏, 显示不可编辑状态, 显示收藏次数



- 没有收藏, 显示可编辑状态, 点击“点击收藏”实现成功收藏, 并更新收藏次数效果



二,思路分析

1.详情页面“点击收藏”显示是否可编辑分析

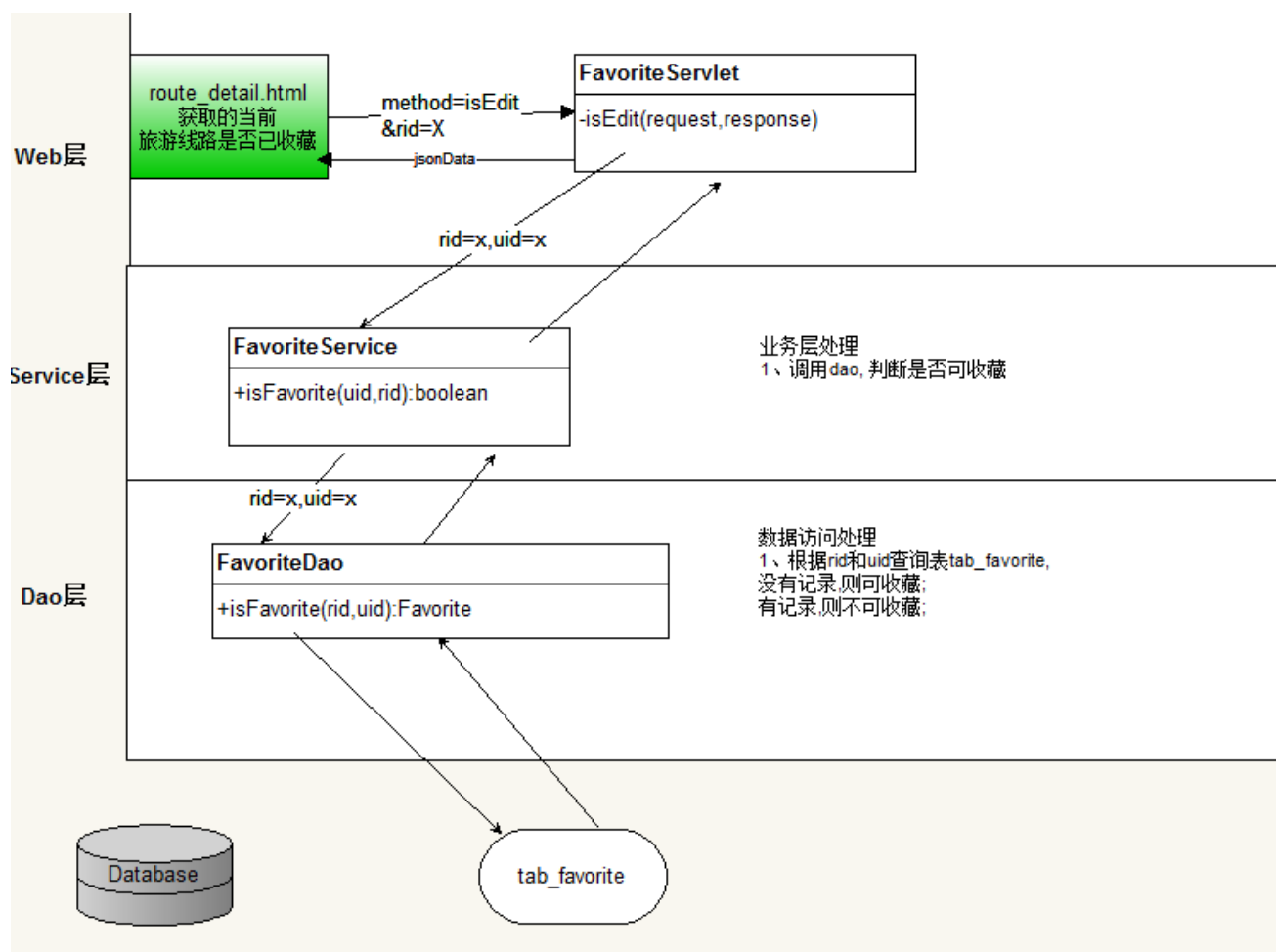
没有登录



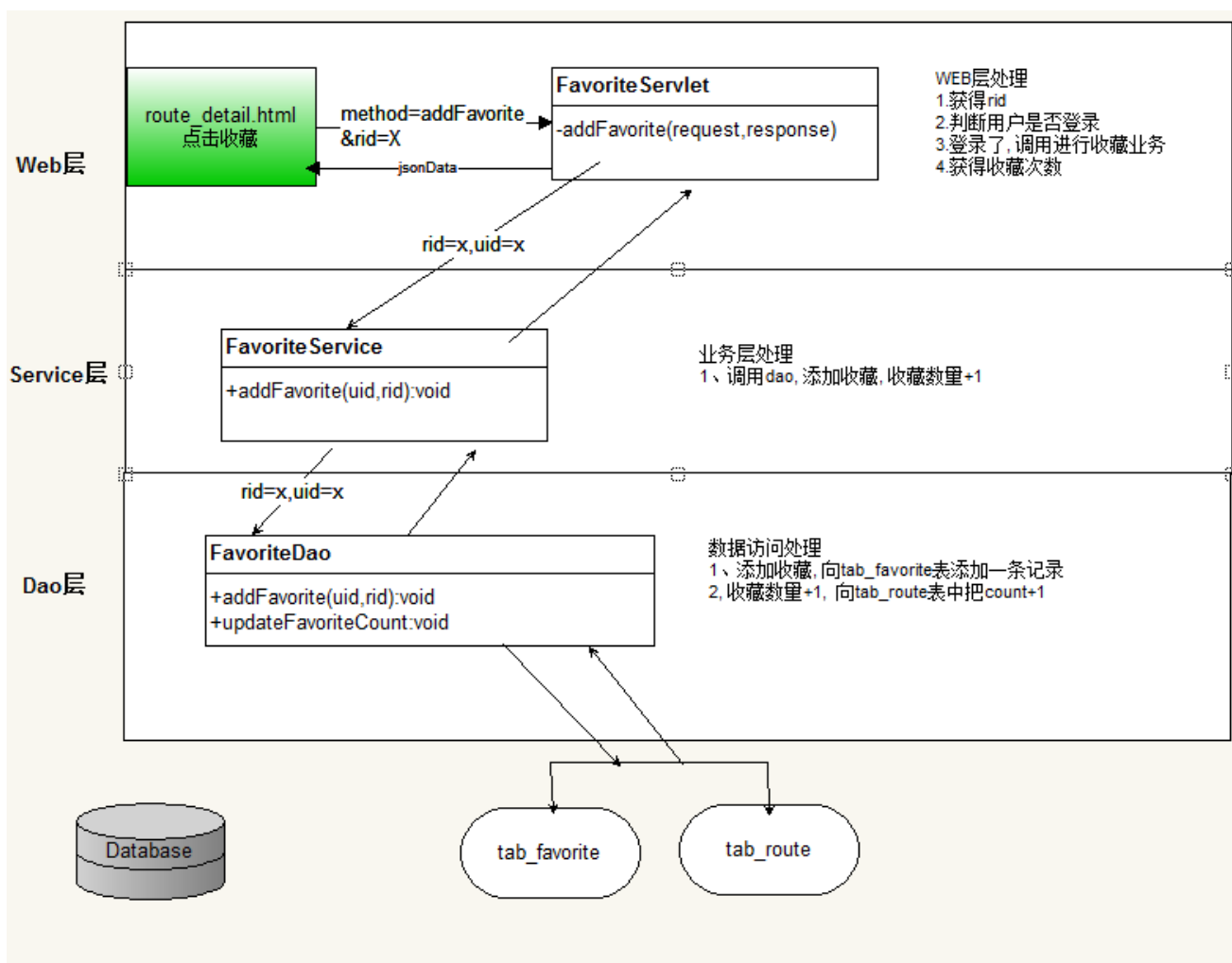
登录了,但是没有收藏



登录了,收藏了



2. 点击“点击收藏”实现增加收藏数据分析



三,代码实现

面向接口编程

一,面向接口编程

1.介绍

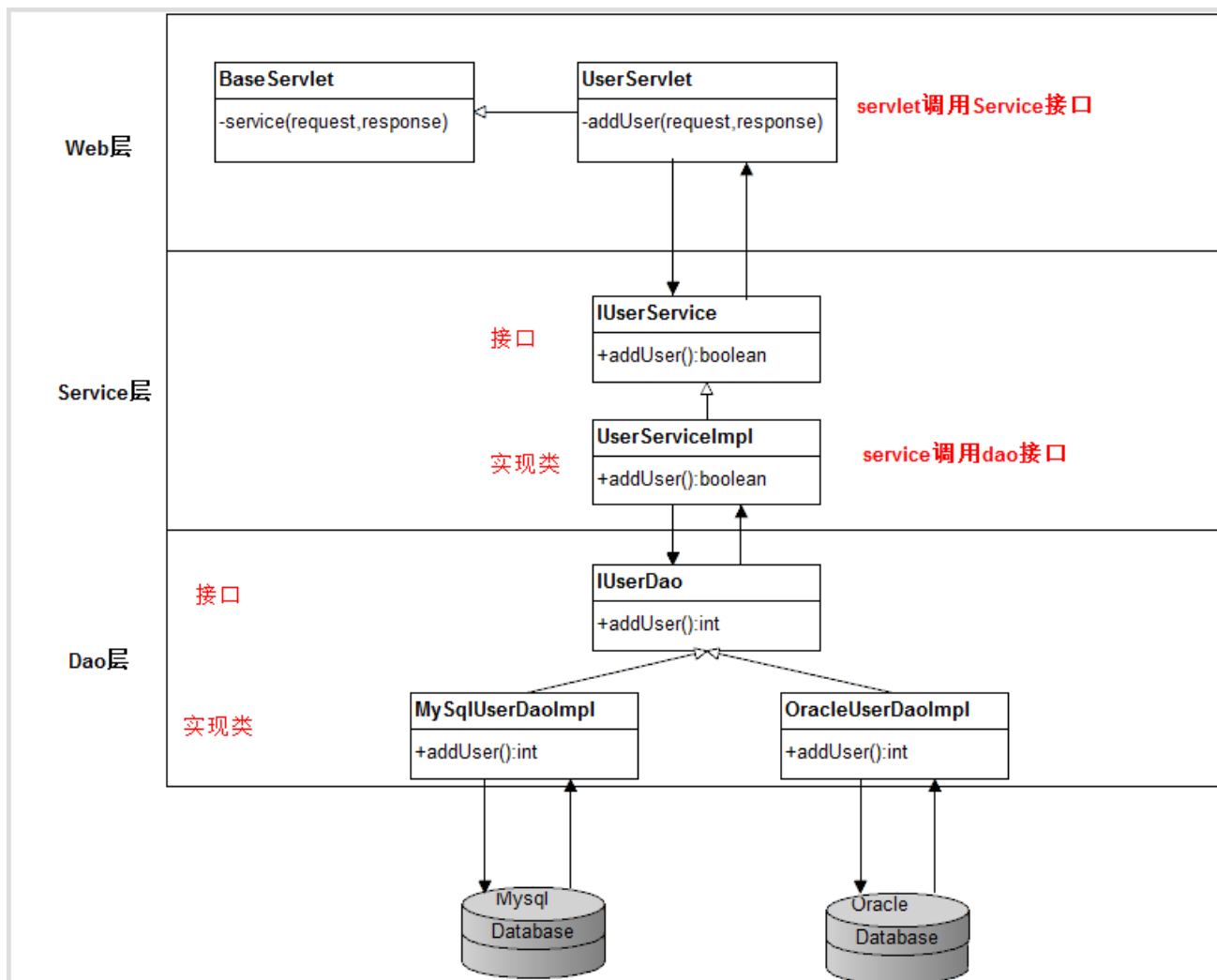
面向接口编程是开发程序的功能先定义接口，接口中定义约定好的功能方法声明，通过实现该接口进行功能的实现，完成软件或项目的要求。软件或项目随着时间的不断变化，软件的功能要进行升级或完善，开发人员可以创建不同的新类重新实现该接口中所有方法的方式进行开发，从而达到系统升级和扩展的目的。

2.面向接口编程案例

2.1 案例需求

在一个软件企业中开发的软件要求即可操作mySql数据库，也可以操作Oracle数据库实现数据的存储与查询。现要求开发一个软件添加用户的功能，可以通过配置文件灵活配置软件是使用mysql数据库还是oracle数据库实现功能。

2.2案例的架构图



3. 面向接口编程的好处

1. 隐藏实现

web层只调用service接口，service层只调用Dao接口，上一层只调用下一层接口，不需要知道具体实现类，从而隐藏了实现。

2. 易扩展

系统功能升级扩展，我们知道程序设计的原则是对修改是关闭，对新增是开放。面向接口编程扩展功能只需要创建新实现类重写接口方法进行升级扩展就可以了，达到了可以不修改源程序代码的基础上达到扩展的目的。

4. 实现代码

4.1 初级版本

- 步骤1：创建IUserService接口，定义addUser()接口方法

```
public interface IUserService {
    boolean addUser() throws Exception;
}
```

- 步骤2：创建UserServicelmpl实现类实现IUserService接口，实现addUser()方法

```

public class UserServiceImpl implements IUserService {
    @Override
    public boolean addUser() throws Exception {
        System.out.println("ServiceImpl执行添加用户业务方法。。。");
        return true;
    }
}

```

- 步骤3：在web层的UserServlet里面创建实例IUserService接口对象定义,并调用addUser()业务方法

```

private IUserService userServiceItf = new UserServiceImpl();
private void addUser(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    try {
        userServiceItf.addUser();//调用业务方法
    } catch (Exception e){
        e.printStackTrace();
    }
}

```

4.2终极版本

面向接口编程的目的是隐藏实现类，在初级版本里面明确使用了ServiceImpl实现类对象。那么如何实现灵活配置实现类达到动态创建实例实现类对象呢？

- 首先，定义配置文件impl.properties配置文件，存放在“src/main/resources”目录下impl.properties文件内容：通过配置文件配置实现类优点非常多，可以灵活配置扩展的实现类

```
IUserService=com.itheima.travel.service.impl.UserServiceImpl
```

- 其次，创建工厂类FactoryUtil.java，代码如下

```

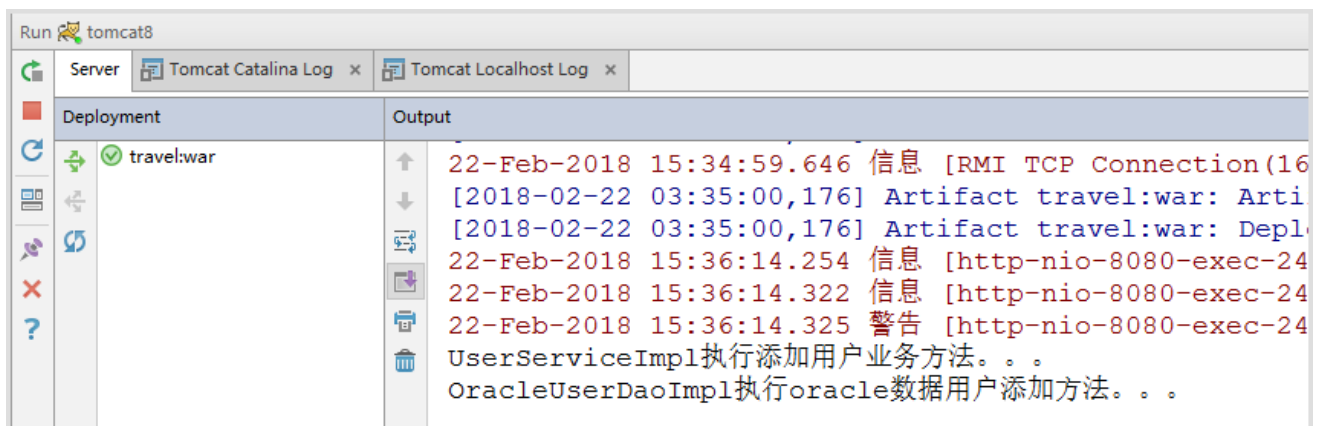
public class FactoryUtil {
    private static ResourceBundle resourceBundle;
    static{
        resourceBundle = ResourceBundle.getBundle("impl");
    }
    public static Object getImplObject(String itfName){
        try {
            String className = resourceBundle.getString(itfName);
            return Class.forName(className).getConstructor().newInstance();
        } catch (Exception e) {
            e.printStackTrace();
            throw new RuntimeException(e);
        }
    }
}

```

- 优化web层的UserService里面创建实例IUserService接口对象定义，达到隐藏实现类的目的

```
private IUserService userServiceItf =  
(IUserService)FactoryUtil.getImplObject("IUserService");  
private void addUser(HttpServletRequest request, HttpServletResponse response)  
    throws ServletException, IOException {  
    try {  
        userServiceItf.addUser();//调用业务方法  
    }catch (Exception e){  
        e.printStackTrace();  
    }  
}
```

运行项目，打开浏览器运行"<http://localhost:8080/travel/user?mmethod=addUser>",控制台输出信息：



The screenshot shows the Eclipse IDE's Run console for a Tomcat8 server. The 'Server' tab is active, showing 'Tomcat Catalina Log' and 'Tomcat Localhost Log'. The 'Deployment' section on the left shows 'travel:war' with a green checkmark, indicating successful deployment. The 'Output' section on the right displays the following log messages:

```
22-Feb-2018 15:34:59.646 信息 [RMI TCP Connection(16  
[2018-02-22 03:35:00,176] Artifact travel:war: Arti  
[2018-02-22 03:35:00,176] Artifact travel:war: Depl.  
22-Feb-2018 15:36:14.254 信息 [http-nio-8080-exec-24  
22-Feb-2018 15:36:14.322 信息 [http-nio-8080-exec-24  
22-Feb-2018 15:36:14.325 警告 [http-nio-8080-exec-24  
UserServiceImpl执行添加用户业务方法。。。  
OracleUserDaoImpl执行oracle数据用户添加方法。。。
```