网上商城实战篇

今日内容介绍

- 订单管理/我的订单
- 订单管理/订单详情
- 订单管理/在线支付
- 权限过滤器

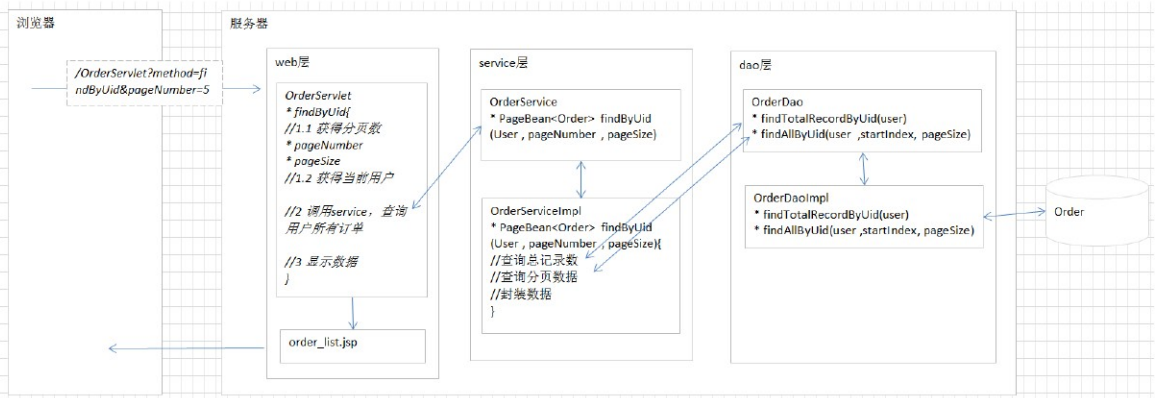
今日内容学习目标

- JavaWeb 知识巩固

第1章 订单管理

1.1 我的订单

1.1.1 分析



1.1.2 代码实现

- 步骤 1：修改 header.jsp 页面，查看“我的订单”

```
<a href="${pageContext.request.contextPath}/OrderServlet?method=findByUid">我的订单</a>
```

- 步骤 2：修改 OrderServlet，添加 findByUid()方法

```
//我的订单
public String findByUid(HttpServletRequest request, HttpServletResponse response)
        throws Exception {

    //1.1 获得当前页
    int pageNumber = 1;
    try {
        pageNumber = Integer.getInteger(request.getParameter("pageNumber"));
    } catch (Exception e) {
    }
    int pageSize = 3 ; //暂时固定值
    //1.2 获得当前用户
    User loginUser = (User) request.getSession().getAttribute("loginUser");

    //2 通过业务层查询
    OrderService orderService = new OrderServiceImpl();
    PageBean<Order>     pageBean     =     orderService.findByUid(loginUser     ,
pageNumber,pageSize);

    //3 显示数据
    request.setAttribute("pageBean", pageBean);

    return "/jsp/order_list.jsp";
}
```

- 步骤 3：修改 OrderService，添加 findByUid()方法

```
//接口
//通过用户查询订单
PageBean<Order> findByUid(User loginUser, int pageNumber, int pageSize)
   throws Exception;
```

```
//实现类
@Override
public PageBean<Order> findByUid(User user, int pageNumber, int pageSize)
   throws Exception{
    //1 查询总记录数
```

```
        int totalRecord = orderDao.findTotalRecordByUid(user);
        //2 封装数据
        PageBean<Order> pageBean = new PageBean<>(pageNumber, pageSize, totalRecord);
        //3 分页数据
        List<Order> data = orderDao.findAllByUid(user , pageBean.getStartIndex() ,
pageBean.getPageSize());
        pageBean.setData(data);
        return pageBean;
    }
```

- 步骤 4：修改 OrderDao，提供 findTotalRecordByUid() 和 findAllByUid() 两个方法

```
    //接口
    /**
     * 通过用户查询订单总记录数
     * @param user 指定用户
     * @return
     */
    int findTotalRecordByUid(User user)throws SQLException;
    /**
     * 查询用户订单分页数据
     * @param user 指定用户
     * @param startIndex 开始索引
     * @param pageSize 每页显示个数
     * @return
     */
    List<Order> findAllByUid(User user, int startIndex, int pageSize)throws Exception;
```

```
    //实现类
    @Override
    public int findTotalRecordByUid(User user) throws SQLException {
        QueryRunner queryRunner = new QueryRunner(JDBCUtils.getDataSource());
        String sql = "select count(*) from orders where uid = ?";
        Long count = (Long) queryRunner.query(sql, new ScalarHandler(), user.getUid());
        return count.intValue();
    }

    @Override
    public List<Order> findAllByUid(User user, int startIndex, int pageSize) throws
Exception {
        QueryRunner queryRunner = new QueryRunner(JDBCUtils.getDataSource());
        String sql = "select * from orders where uid = ? order by ordertime desc limit ?,?";
        List<Order>         list         =         queryRunner.query(sql,         new
BeanListHandler<Order>(Order.class), user.getUid(), startIndex,
```

```
                pageSize);
        // 遍历获得每个订单:
        for (Order order : list) {
            // 进行查询每个订单的订单项:
            sql = "SELECT * FROM orderitem o,product p WHERE oid = ? AND o.pid=p.pid";
            // Array ArrayList Map MapList Bean BeanList ColumnName Scalar Keyed
            List<Map<String, Object>> oList = queryRunner.query(sql, new
MapListHandler(), order.getOid());
            for (Map<String, Object> map : oList) {
                // 将属于订单项的封装到 OrderItem 中.
                OrderItem orderItem = new OrderItem();
                BeanUtils.populate(orderItem, map);
                // 将属于商品的数据封装到 Product 中.
                Product product = new Product();
                BeanUtils.populate(product, map);
                orderItem.setProduct(product);
                orderItem.setOrder(order);

                order.getList().add(orderItem);
            }
            order.setUser(user);
        }
        return list;
```

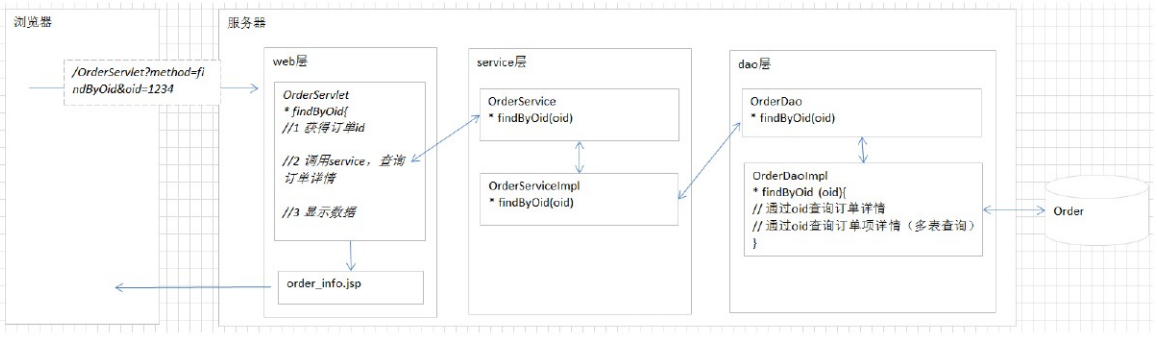
- 步骤 5：jsp 页面显示表单数据

```
    <%--遍历订单 --%>
    <c:forEach items="${pageBean.data}" var="order">
    <tbody>
        <tr class="success">
            <th colspan="5">订单编号:${order.oid} </th>
        </tr>
        <tr class="warning">
            <th>图片</th>
            <th>商品</th>
            <th>价格</th>
            <th>数量</th>
            <th>小计</th>
        </tr>
        <c:forEach items="${order.list}" var="orderItem">
        <tr class="active">
            <td width="60" width="40%">
                <input type="hidden" name="id" value="22">
                <img
src="${pageContext.request.contextPath}/${orderItem.product.pimage}"      width="70"
```

```
height="60">
            </td>
            <td width="30%">
                <a target="_blank">${orderItem.product.pname}</a>
            </td>
            <td width="20%">
                ￥${orderItem.product.shop_price}
            </td>
            <td width="10%">
                ${orderItem.subtotal}
            </td>
            <td width="15%">
                <span class="subtotal">￥${orderItem.subtotal}</span>
            </td>
        </tr>
        </c:forEach>
    </tbody>
    </c:forEach>
```

1.2 订单详情

1.2.1 分析



1.2.2 代码实现

- 步骤 1：显示“我的订单”时，显示订单状态 state 对应的文字提示

```
订单编号:${order.oid}, ${order.state},

<c:if test="${ order.state == 1 }">
```

```
<a href="${pageContext.request.contextPath}/OrderServlet?method=findByOid&oid=${order.oid }">付款</a>
    </c:if>
    <c:if test="${ order.state == 2 }">
        等待发货
    </c:if>
    <c:if test="${ order.state == 3 }">
        确认收货
    </c:if>
    <c:if test="${ order.state == 4 }">
        订单结束
    </c:if>


    <font color="#b00">总金额：${order.total}</font>
```

- 步骤 2：修改 OrderServlet，添加 findByOid 方法，点击“付款”，根据订单 id 查询看订单详情。

```
//通过订单 id 查询详情
public String findByOid(HttpServletRequest request, HttpServletResponse response)
        throws Exception {

    //1 获得当前页
    String oid = request.getParameter("oid");

    //2 通过业务层查询
    OrderService orderService = new OrderServiceImpl();
    Order order = orderService.findByOid(oid);

    //3 显示数据
    request.setAttribute("order", order);

    return "/jsp/order_info.jsp";
}
```

- 步骤 3：修改 OrderService，添加 findByOid()方法

```
//接口
//通过 oid 查询订单详情
Order findByOid(String oid);

//实现类
public Order findByOid(String oid) {
    return orderDao.findByOid(oid);
}
```

- 步骤 4：修改 OrderDao，添加 findByOid()方法

```
//接口
/**
 * 通过 oid 查询详情
 * @param oid
 * @return
 */
Order findByOid(String oid) throws Exception;
```

```
//实现类
@Override
public Order findByOid(String oid) throws Exception {
    QueryRunner queryRunner = new QueryRunner(JDBCUtils.getDataSource());
    String sql = "select * from orders where oid = ?";
    Order order = queryRunner.query(sql, new BeanHandler<Order>(Order.class), oid);
    // 封装 Order 中 list 集合的数据
    sql = "select * from orderitem o,product p where o.pid = p.pid and o.oid = ?";
    List<Map<String, Object>> oList = queryRunner.query(sql, new MapListHandler(),
oid);
    for (Map<String, Object> map : oList) {
        OrderItem orderItem = new OrderItem();
        BeanUtils.populate(orderItem, map);

        Product product = new Product();
        BeanUtils.populate(product, map);
        orderItem.setProduct(product);
        orderItem.setOrder(order);

        order.getList().add(orderItem);
    }
    return order;
}
```

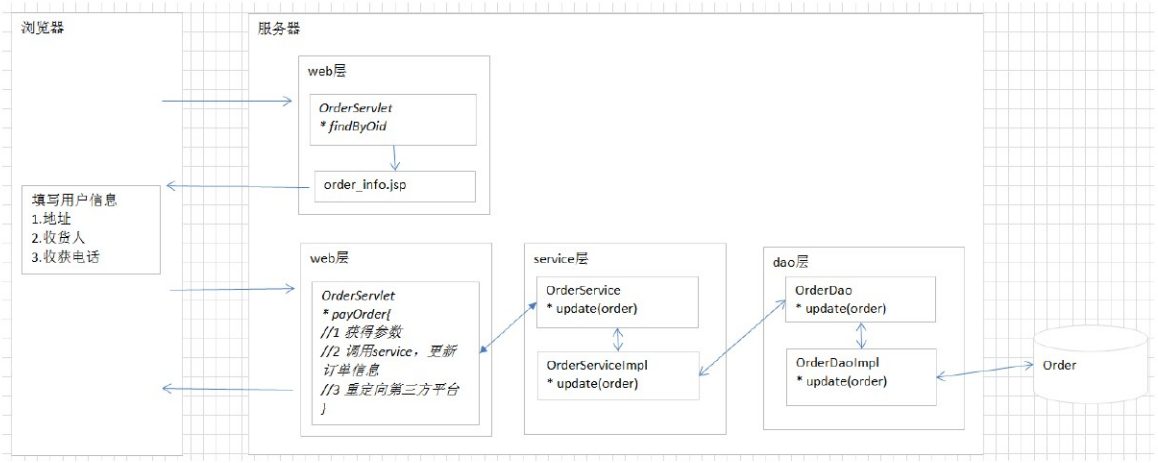
- 步骤 5：修改/jsp/order_info.jsp 页面，显示详情，准备付款。

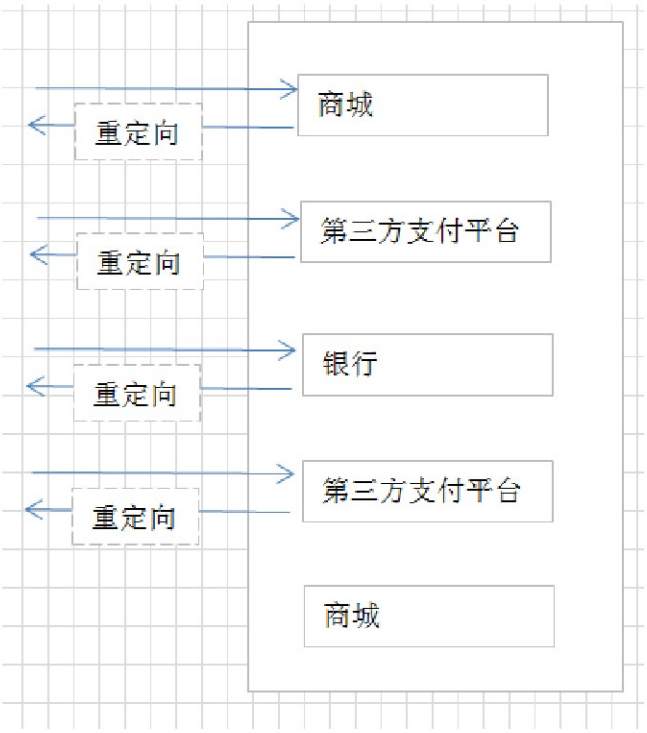
```
订单编号:${order.oid}
<tr class="warning">
    <th>图片</th>
    <th>商品</th>
    <th>价格</th>
    <th>数量</th>
    <th>小计</th>
</tr>
<c:forEach items="${order.list}" var="orderItem">
```

```
    <tr class="active">
        <td width="60" width="40%">
            <input type="hidden" name="id" value="22">
            <img
src="${pageContext.request.contextPath}/${orderItem.product.pimage}"      width="70"
height="60">
        </td>
        <td width="30%">
            <a target="_blank">${orderItem.product.pname}.</a>
        </td>
        <td width="20%">
            ￥${orderItem.product.shop_price}
        </td>
        <td width="10%">
            ${orderItem.count}
        </td>
        <td width="15%">
            <span class="subtotal">￥${orderItem.subtotal}</span>
        </td>
    </tr>
    </c:forEach>
    商品金额: <strong style="color:#ff6600;">￥${order.total}.00 元</strong>
```

1.3 在线支付(了解)

1.3.1 分析





1.3.2 代码实现

- 步骤 1：修改/jsp/order_info.jsp ，完成支付。表单摘要如下：

```
<form id="orderForm" action="${pageContext.request.contextPath}/OrderServlet" >
    <%--隐藏字段 --%>
    <input type="hidden" name="method" value="payOrder"/>
    <input type="hidden" name="oid" value="${order.oid}"/>
    <input name="address" placeholder="请输入收货地址">
    <input name="name"  placeholder="请输收货人">
    <input name="telephone" placeholder="请输入联系方式">
    <input type="radio" name="pd_FrpId" value="ICBC-NET-B2C" checked="checked" />
工商银行
    <a href="javascript:document.getElementById('orderForm').submit();">
</form>
```

- 步骤 2：修改 OrderServlet，添加 payOrder 方法

```
// 订单付款
public String payOrder(HttpServletRequest request, HttpServletResponse response)
        throws Exception {

    //1 获得请求参数
    String oid = request.getParameter("oid");
    String name = request.getParameter("name");
```

```
        String address = request.getParameter("address");
        String telephone = request.getParameter("telephone");
        String pd_FrpId = request.getParameter("pd_FrpId");

        //2 调用业务层，查询订单，并修改订单状态
        OrderService orderService = new OrderServiceImpl();
        Order order =  orderService.findByOid(oid);
        order.setAddress(address);
        order.setName(name);
        order.setTelephone(telephone);

        orderService.update(order);

        //3 重定向到第三方平台
        String p0_Cmd = "Buy";
        String p1_MerId = "10001126856";
        String p2_Order = oid;
        String p3_Amt = "0.01";
        String p4_Cur = "CNY";
        String p5_Pid = "";
        String p6_Pcat = "";
        String p7_Pdesc = "";
        String                                  p8_Url                                  =
"http://localhost:8080/"+request.getContextPath()+"/OrderServlet?method=callBack";
        String p9_SAF = "";
        String pa_MP = "";
        String pr_NeedResponse = "1";
        String                                  keyValue                                =
"69cl522AV6q613Ii4W6u8K6XuW8vM1N6bFgyv769220IuYe9u37N4y7rI4Pl";

        String hmac = PaymentUtil.buildHmac(p0_Cmd, p1_MerId, p2_Order, p3_Amt, p4_Cur,
p5_Pid, p6_Pcat, p7_Pdesc, p8_Url, p9_SAF, pa_MP, pd_FrpId, pr_NeedResponse, keyValue);

        StringBuffer                    sb                    =                    new
StringBuffer("https://www.yeepay.com/app-merchant-proxy/node?");
        sb.append("p0_Cmd=").append(p0_Cmd).append("&");
        sb.append("p1_MerId=").append(p1_MerId).append("&");
        sb.append("p2_Order=").append(p2_Order).append("&");
        sb.append("p3_Amt=").append(p3_Amt).append("&");
        sb.append("p4_Cur=").append(p4_Cur).append("&");
        sb.append("p5_Pid=").append(p5_Pid).append("&");
        sb.append("p6_Pcat=").append(p6_Pcat).append("&");
        sb.append("p7_Pdesc=").append(p7_Pdesc).append("&");
        sb.append("p8_Url=").append(p8_Url).append("&");
```

```
        sb.append("p9_SAF=").append(p9_SAF).append("&");
        sb.append("pa_MP=").append(pa_MP).append("&");
        sb.append("pd_FrpId=").append(pd_FrpId).append("&");
        sb.append("pr_NeedResponse=").append(pr_NeedResponse).append("&");
        sb.append("hmac=").append(hmac);

        // 使用重定向:
        response.sendRedirect(sb.toString());

        return null;
    }
```

- 步骤 3：修改 service

```
    //接口
    //更新
    void update(Order order)throws SQLException;
```

```
    //实现类
    public void update(Order order) throws SQLException{
        orderDao.update(order);
    }
```

- 步骤 4：修改 dao

```
    //接口
    /**
     * 更新订单
     * @param order
     */
    void update(Order order)throws SQLException;
```

```
    //实现类
    public void update(Order order) throws SQLException {
        QueryRunner queryRunner = new QueryRunner(JDBCUtils.getDataSource());
        String sql = "update orders set total=?,state=?,address=?,name=?,telephone=?
where oid = ?";
        Object[] params = { order.getTotal(), order.getState(),
                            order.getAddress(), order.getName(),
                            order.getTelephone(), order.getOid() };
        queryRunner.update(sql, params);
    }
```

- 步骤 5：回调程序，修改 OrderServlet，添加 callback()方法

```
    //付款成功，回调方法，修改订单状态
```

```
    public String callBack(HttpServletRequest request, HttpServletResponse response)
            throws Exception {

        String r6_Order = request.getParameter("r6_Order");
        String r3_Amt = request.getParameter("r3_Amt");

        OrderService orderService = new OrderServiceImpl();
        Order order = orderService.findByOid(r6_Order);
        order.setState(2);
        orderService.update(order);

        request.setAttribute("msg", "请！您已经付款成！订单号为："+r6_Order+" 支付的金额为:"+r3_Amt);
        return "/jsp/msg.jsp";
    }
```

第2章 权限过滤器

2.1 需求

我们已经完成用户管理、商品管理、分类管理、订单管理等功能模块，订单模块和购物车模块必须确保用户登录后才能访问，我们会在 servlet 中编写相同的校验代码，我们可以通过权限过滤器统一处理。

2.2 代码实现

- 步骤 1：实现类

```
    public class PrivilegeFilter implements Filter{
        @Override
        public void init(FilterConfig filterConfig) throws ServletException {
        }

        @Override
        public void doFilter(ServletRequest request, ServletResponse response,
FilterChain chain)
                throws IOException, ServletException {
```

```
            HttpServletRequest req = (HttpServletRequest) request;
            User loginUser = (User) req.getSession().getAttribute("loginUser");
            if(loginUser == null){
                req.setAttribute("msg", "您还没有登录！没有权限访问！");
                req.getRequestDispatcher("/jsp/msg.jsp").forward(req, response);
                return;
            }
            chain.doFilter(req, response);
        }

        @Override
        public void destroy() {
        }

    }
```

- 步骤 2：web.xml 配置，只对订单操作、购物车操作进行登录权限校验

```
    <filter>
        <filter-name>PrivilegeFilter</filter-name>
        <filter-class>cn.itcast.store.web.filter.PrivilegeFilter</filter-class>
    </filter>
    <filter-mapping>
        <filter-name>PrivilegeFilter</filter-name>
        <url-pattern>/jsp/order_info.jsp</url-pattern>
        <url-pattern>/jsp/order_list.jsp</url-pattern>
        <url-pattern>/OrderServlet</url-pattern>
        <url-pattern>/jsp/CartServlet</url-pattern>
    </filter-mapping>
```