零售商页面实现思路

1.登录功能：将前台页面传过来的用户名和密码在代码中进行处理，代码会去调用数据库，验证用户输入的是否正确。同时，登录的时候会验证用户的身份是管理员还是零售商。（这里角色分别是管理员、一级、二级、三级零售商）。代码中，同时会添加一条登录日志到数据库中

2.修改密码功能：首先在前台页面上使用js进行逻辑验证（a.三个输入框是否大于8位b.第二三个框输入的是否一致），如果前台验证都通过，就把原密码传到后台中和数据库中进行匹配。如果相同，就可以更新数据数据库字段。如果不相同，就返回消息到前台提示与原密码不相同。

3.查看货源功能实现。信息展示：跳转到这个页面的时候，代码中做处理，将有关商品的信息保存到request的属性中，然后在jsp页面中用EL表达式渲染出来。商品的查询功能实质就是使用了sql的where语句来进行查询。分页功能使用mybatis的pageHelper插件实现。点击查看详情的时候，这个详情按钮上有一个属性是商品id，会根据这个id向后台发送一个ajax请求，返回的是商品的详细信息，然后在模态框中呈现出来。添加购物车，实质上就是向数据中添加一条字段，同时对添加的数量进行处理(a.改不能负数.b比库存小)

4.销售情况的实现。主要就是使用了百度的开源图表工具echarts。信息展示页面加载的时候会向后台发送请求，从数据库拿到数据，然后在前台用js进行一些运算（比如当天的利润和、当天的销售情况），然后在echarts上配置一下，在jsp上呈现出来。按日期选择也是发送请求到后台，拿到数据，然后呈现。上传销售情况的处理其实和表单是相同的，不过这里使用js获取页面上的值，然后发送请求，请求体中的数据会被保存到数据库，下次看见的就是最新上传的了。

5.发送信息给管理员。这里是发送给roleId为0（就是管理员）的用户。这些信息会被存到数据库中。有个from和to字段，from字段就是发送人，to就是接收人。消息列表就是点击的时候后台会去处理数据（和商品展示功能一样），然后同样在前台渲染出来，注意，这里展示出来的消息应该是to=username的值，就是接收人是当前用户名的消息。

6.退出登录。实质就是将所有的session清除，并且跳转到登录界面。

管理员的功能：

查看消息列表显示出来的信息应该roleid=0（代表管理员）的信息。（其他大体相同）

管理货源。这里最开始的也是显示一些货源，但是多了一个编辑和删除按钮。编辑按钮中有一个商品id，点击的时候会向后台拿到这个商品的具体信息。然后填充到输入框中。（和查看详情按钮原理是一样的），修改后提交就将表里的数据传到后台，后台会将这些数据封装（组装）成一个实体，然后去数据库中进行更新。

管理订单。订单分成了四个状态。“确定”这个按钮中有保存订单id，点击下去的时候其实就是将下拉框的值发送到后台，然后后台根据这个id去查找这个订单记录，并且进行更新。当选择到取消订单或者审核未通过的时候，会弹框。这里是在js中进行判断这个下拉框值是什么，如果是这二个值，就弹出模态框。查看详情的按钮也保存这个订单id的值。点击的时候发送请求然后返回信息，然后填充进去（和查看详情，编辑商品是一样的）

管理零售商。添加零售商上，其实就是将前台的值传到后台，后台进行封装，然后插入一条记录到user表中。

编辑零售商按钮中有存一个用户名，然后吧这个用户名发送到后台，后台去查找这个用户，然后返回信息给前台，前台就可以展示了。点击确认修改的时候，就是将输入框中的数据传到后台，后台根据用户名查找出一条记录，然后进行更新。禁用按钮中也存了一个用户名id，然后后台根据这个id去找出这个用户名，并且更新这个用户的status字段。

套路都是一样

1.当点击一个链接的时候，内容会被显示，这个会跳转到后台。后台就做两个操作：一个是去数据库中查找对应的内容，然后设置到request属性中，一个是跳转页面。页面用预留了EL表达式的值，预留占位符一样，然后就可以被填充（存到request属性中就可以被填充）。

2.当点击一个按钮的时候，就涉及到发送请求（给后台一些数据），后台根据这些数据去做对应的处理（也可以不做处理直接返回给前台），然后返回到前台，前台的一些输入框，或者个人信息就可以被展示。有的地方是使用js来处理的。

触发js的点击事件（和按了按钮一样），点击时候发送一个请求到后台，然后拿到数据，然后在指定位置的元素进行填充。基本是同理的。

系统中最关键的技术：

在技术上，最关键的使用到的技术是属于SpringMVC和AmazeUI的使用了。SpringMVC作为一款出色的MVC的框架，能够大大减少我们的开发时间，同时我们映射各式各样的URL，我们只需要在开始阶段简单的配置，随后开发的速度变大大提升。它解决了我们本身代码过于耦合的想象，在程序中贯穿了面向接口的编程思想。同时，前端使用了基于扁平化风格的自适应框架，amaui，好处有二，其一，其风格简洁大方，其二，能够适用不同分辨率的屏幕。作为一个以后端开发为主的程序，这个框架能够让我制作出给美观的界面。

软件实现的难点：

软件实现的难点在于我在设计数据库的时候采用的第三范式的数据表设计，但是却采用java语言。java是一门面向对象的编程语言，无法将java中的类和数据库中一个表的字段一一对应，这就涉及到了复合属性的映射，只有当对业务逻辑充分理解的情况下，才能做好这种映射逻辑。同时，前台展示的数据也是需要做处理的，这就需要编写一些复杂的业务逻辑代码来完成。总的来说，就是从最底层拿到是数据，需要对它们进行一层层封装、一层层解析，才能拿到我们想要的数据，这个我认为是软件实现的难点。