自我介绍：

您好：

我叫陈红利来自河北石家庄，14年大学毕业后进入一家公司做java开发，到目前为止参与开发过几个java的项目，这些项目我主要使用的底层是SSH、SSM框架。最近做的一个项目做的是网购类的项目，项目名是生鲜快购，是我们公司承接其他公司的外包项目，这个生鲜网项目主要分为网站前台和后台，后台又分为运营后台和商家后台。我参与了运营商后台其中的运营商登录和商品管理功能。

（我叫陈红利，来自河北石家庄，从事java行业已经三年多的时间了，在这期间，我参与开发了有五个项目，在这些项目中主要使用的有SSH、SSM框架。最近做的一个项目做的是电商项目，为了提高开发效率，我们在项目中前端用的Angular+BootStrap、后端用的SSM+Dubbo、权限验证和登录使用的SpringSecurity、缓存用的redis、商品评论用的MongoDB。）

运营商登录：

录首先我们需要创建一个实现了接口的实现类，复写这个方法，在该方法中获取登录的用户名，根据用户名去查询用户的密码，然后把用户名和密码通过SpringSecurity与我们输入的用户名和密码进行匹配。如果匹配成功跳转到指定的页面。我们使用BCrypt算法实现密码加密。当我们添加用户信息的时候，创建BCrypt密码加密的对象，然后调用加密encode方法实现加密放到数据库。最后还需要在springsecurity的配置文件中指定加密算法。

商品管理：商品录入、图片上传、商品三级分类（三级分类前台用了面包屑导航和搜索）

(录入)新增商品信息

新增商品要同时操作两张表，商品表和商品扩展描述信息表，首先创建一个包装类里面包含商品的基本信息和商品的扩展信息两个属性。当点击保存的时候把商品的基本信息和商品的扩展信息添加包装类中，然后把商品的名称、价格、副标题等（一些基本信息）插入商品表中，通过返回插入数据的id，以这个id为外键向商品扩展信息表中插入商品的描述信息。

商品介绍的内容是通过富文本编辑器获得，需要导入富文本编辑器的css和js，在页面中通过js的edit.Html()用来获取编辑器的内容。

商品图片上传

我们录入的图片信息都放在FastDFS文件服务器上管理。图片上传的第一步是在工程中添加文件上传的依赖和工具类，然后在商家后台配置文件服务器地址和图片上传路径，最后在springMVC中上传解析器。

第二步在前端js页面通过AngularJS实现上传，这里面需要设置两个属性（Content-Type header、transformRequest: angular.identity），当点击上传的时候，后端通过FastDFS的客户端上传图片并获得上传地址，然后在添加商品描述对象的时候，把图片地址插入到商品扩展信息表里面存放图片地址的字段中。

【这里面需要设置两个属性（Content-Type header属性，这个属性的默认值是application/json。我们如果要上传图片，需要把这个属性设置为null，这样就会使用 multipart/form-data值。另外需要设置transformRequest: angular.identity，这个属性主要是上传对象序列化。）】

商品的三级分类显示

根据分类的父id查询分类，分类页面默认显示顶级分类，当我们点击顶级分类的时候根据这个顶级分类的id查询二级分类，当点击某个二级分类根据二级分类的id去查询三级分类。然后在后台定义一个根据父id查询分类列表的方法，在前端的js页面的controller层接收传递过来的父id调用后台的方法，当进入三级分类页面的时候默认显示顶级列表。（使用ng-init在页面初始化的时候调用js的controller层，通过service获得后台查询到的parent\_id=0的数据的集合如果成功获得就将数据添加到$scope的变量中，使用ng-repeat循环集合，最终使用angularjs的表达式显示。）

商品三级分类前台用了面包屑导航

实现面包屑导航分三级，首先定义一个分类的级别默认是1。当点击”查看下级分类”的按钮时候，获取到当前分类对象，并把当前分类对象的名称显示在面包屑导航中。首先设置面包屑导航的分类级别，然后根据不同的分类级别设置内容。使用ng-if判断当前的分类级别，如果当不等于3的时候才显示“查询下级”按钮。

商品三级分类前台用了搜索

我们网站的搜索使用了solr。Solr是一个实现搜索的开源服务。它本身是一个war包，可以放到tomcat中，去配置中文分词器并定义业务域，首先指定实体类中的属性和索引库域的对应关系，然后查询数据库中所有需要导入的数据，把数据导入到索引库中去创建一个搜索对象和一个条件对象并设定查询的域和关键字，最后调用工具类的查询方法执行查询。

购物车管理

在详情页点击“加入购物车”或者“查看购物车”的时候我们做了一个判断，假如用户没有登录，我们会将用户加入到购物车中的商品信息存入到cookie中，然后在用户点击"结算"时提醒用户进行登陆，并将购物车中的信息和用户关联起来存入到redis中，如果用户已经登陆则用户将商品加入购物车时直接将信息存入redis中，并设置过期时间，这样做的目的是为了防止用户频繁操作购物车增加系统压力，而redis可以看做是一个持久化的缓存。用户填写完收获地址信息以及选择支付方式，点击“提交订单”时会将reidis中的信息取出来插入到数据库的订单表中，并清除reidis中该购物车的信息，同时将订单表中该条记录的状态。

评论管理

mongodb是一个nosql数据库，我们在项目中通常用它来存储评论信息，

【评论id,商品id,标题，评分，内容，评论人信息，评论的发布时间】

因为每个商品都会有评论信息，而且某些热门商品的评论信息可能是数千条，mongodb正好适用于这种大数据量、高并发、弱事务的互联网应用。考虑到mongodb的高可用性我们搭建了3台mongodb数据库来实现副本集，这样不仅可以达到故障自动转移的特性而且也可以通过读写分离提高性能，即便主服务器宕机了，还会通过投票选举出下一个主服务器继续提供服务。为了控制留言，我们留言的界面设置订单状态，状态为5，也就是交易成功收货后才能评论，并在评论成功后将订单状态改为6。

Redis、Nginx + Tomcat负载均衡等工作：

我们在做这个项目时，考虑到服务器性能的问题，最开始想到使用纵向扩展来增加硬件的配置提高其性能，但这样做比较耗费资金，而且服务器内存空间也是有限的；所以后来就使用横向扩展,当时我们使用nginx+3个tomcat进行负载均衡，没进行负载均衡之前，所有的请求都由一台tomcat处理，这样会使我们的tomcat所承受的压力增大，而进行负载均衡之后，同样数量的请求经过nginx将其分发到多台tomcat进行处理，从而降低每台tomcat所承受的压力,而且当其中一台机器宕机时，其他机器还可以继续提供服务，保证服务不间断。

当时项目在部署完成后，遇到这么个问题，用户登录输入验证码的时候，明明验证码输入的正确，但总是提醒说验证码不正确从而不能正常登录，经过分析后发现第一次请求被发送到t1上，那么放在session中的验证码就被放到了t1上，当用户输入验证码点击登录时，新发送的请求有可能被发送到t2上面，这样在进行对比时就肯定会不一致从而提示验证码输入错误，后来我就考虑使用ip\_hash这种负载均衡策略来代替默认的轮询策略，虽然解决了验证码错误问题，但是在后续的测试中发现如果用户在使用过程中突然一台服务器宕机了，那么因为session在这台服务器上存储着，所以就会提示用户重新登录，这样用户的体验度非常不好，最后就通过将session信息保存到redis服务器中从而在多台web服务器中实现session共享，这样就解决了上面所说的那些问题。