JavaWeb项目总结

一、项目的功能

本次项目，一共有四大功能，增、删、改、查（逃  
 话虽如此，对需求的把握是十分重要的，凡是做实事都要有目的性，有了目的再做规划，才不至于乱七八糟甚至南辕北辙。在软件开发中，这就是需求获取、需求分析。由于需求基本都已经实现，这里就仅说明功能。  
 这个小项目包含前台管理系统、后台管理系统两个子系统。前台主要包括用户模块、主页显示模块、购物车模块、订单模块；后台主要包括分类管理、商品管理、系统用户管理、注册用户管理、订单管理。值得注意的是：后台作为管理员权限是能获取前台所有数据的，反之则不行。  
 后台的增删改查往往是针对单一实体，所以实现比较简单；而前台是面向用户的，所以就需要分解成子功能，涉及到多个实体的数据增删改查。比如，用户模块的登录注册需要异步校验、邮箱验证，订单模块的生成订单包括验证库存（查）、添加order（增）、迁移cartItems到orderItems（查增）、删除cartItems（删）、更改库存（改）、查询orderItems（查）这一连串功能。

二、项目的核心思路

我想自问自答一个问题：这个项目功能实现的核心思路、流程是什么？最主要的功能就是增删改查，而实现就是三层架构。  
 表示层（web层：jsp、servlet）：前端用户肯定要输入一些数据，叫做参数，然后servlet接收这些参数，进行校验，为了减少方法的参数，往往要把参数选择性地封装到实体对象里，然后调用下一层的方法，返回用户需要的数据或结果（重点，表示层只拿数据，并不掺和业务逻辑和数据的事，这就是三层架构的精髓），然后把这些结果放到一个域里面，jsp页面通过EL表达式取出数据。  
 业务逻辑层（service层）：如果要做的操作比较复杂，则需要在service层进行分发、组合，这个“复杂”主要是因为Dao层的一个方法往往只能做增删改查的一种操作、而且只能操作相关联的表，比如“生成订单”实际上调用了增删改查所有的功能，所以必须在service层进行分发。显然，如果只用做一种操作，那往往转发给Dao层就行了。  
 数据访问层（dao层）:直接以sql语句对数据库进行增删改查。然后将结果经过service层返回给web层。

三、编程套路的问题

（1）三层架构。应该说三层架构、MVC的思想就是一种核心编程套路，它实现了解耦，体现了多态，体现了软件工程里很多原则。  
 （3）实体类的设计，对实体进行抽象的能力。抽象、封装、继承、多态是面向对象最主要的特点，增删改查的就是实体对象的封装与解封装，所以说实体是核心也不为过。对“购物车”“订单”这个实体的封装就很能说明问题，关键是首先得确实能用言语表达清楚要抽象的实体是什么，比如说“购物车”和“购物车列表”在人家嘴里说来可能都是那个页面，但是用集合抽象就不能“一词多义”了。如果把cart抽象为List<cartItem>，那么cartList的命名就不伦不类了，要么叫cart，要么叫cartItemList。  
 （2）数据表的设计。这个问题跟松哥讨论过，我感觉建表这块不太懂套路，做cartItem表只留了uid, pid的外键，字段少了需要外键联查，各层封装解封装也麻烦；我担心字段多了出现冗余和扩展性，但是松哥说根据前端展示建表就行了，最后的order表我就索性就多建字段，总体确实方便很多。

四、其他心灵鸡汤

松哥曾说，tomcat帮你做了70%的事。的确，有了request/response, getParameter()，好像天下我有了，干起活来也变成了重复性搬砖。但是时常想起而焦虑，就算不用你写底层，就算JAVAEE的底层也没人用了，那http、servlet、request/response、cookie/session的原理能很流畅地讲出来吗？尽管知乎上金旭亮老师说，懂不懂原理和API用得熟不熟根本没关系，但如果只会拿着API干活，也不是真的程序员吧，我觉得对技术的敬畏是程序员的立身之本。而自己对底层也掌握地不是很好，相信很多人跟我一样。听说Spring的“底层”更加复杂，我又该如何去面对呢。  
 说到“底层”，我不愿称其为底层，我愿称为“下层”，因为JAVAEE的下层还有JAVASE/数据结构与算法/离散数学——JAVA虚拟机/编译原理——操作系统/计算机组成原理——高数/线代/英语...直接说某层的底层是“底”到哪一层呢？就面试而言，JAVASE/数据结构与算法是最重要的“底层”，我觉得一定要另外花时间去理解和深入，往境界说，这也是知识分子的追求。