梁怡萍项目收获

1.协调数据库设计的时候对于考试，试卷，试题，考生几个元素之间的关联设计比较混乱，最后认为存留冗余的情况能使得映射关系更清晰，但是数据量大的时候其实还应该更精简；

2.对于模块的划分应该更明确界限，否则调用传参时有时会逻辑模糊；

3.软件并不只是代码工作，更多的是设计，管理与维护，写好文档对于自己和其他成员都是一种方便，并且代码风格和注释也是一种规范，大家按同一种规范开展项目更容易一些

4.在环境配置的时候，要尽量统一所有配置，在这次项目的实现过程中，我将Myeclipse编码方式改为了UTF8，而另外一个小伙伴却使用的GBK，在最终合并所有功能的过程中，产生了极大的不方便，每次后端代码部分的中文都会变成乱码，因此下次前期工作应当更加充分，环境配置尽量一致

5.还有一个最大的感受就是，如果一味的追求完美，代码效率最高，实现的最精炼，可能会导致实现时间超长，实现过程异常艰难，所以也应当适当的放弃一些效率，从而大幅度提高编码速度，最终项目基本完成之后再去考虑优化，这样会让整个项目推进的更加顺利

吴渊博项目收获

1.数据库设计方面的问题

感觉数据库设计是整个项目体系构建的基础，因为所有操作的本质都是对数据库的增删改查，如果数据库没有设计完善合理的话，所有的操作都会增添很多麻烦。想要将数据库分模块管理，降低冗余度，但是这样设计使得在进行操作时设计的表越来越多，对数据库的访问难度和复杂度都会变高。最后导致自己的那个模块整个经历过一次推翻重写的过程。

2.学会了Git的使用

因为技术还不成熟，之前写代码的时候，经常出错，特别是在增添新的部分时，有时原来一部分已经测试通过了，但是由于加上了新的内容，而这些内容可能涉及到很多方面的修改，这样就会发生很多错误。而在使用Git之后，就能够非常清晰地看到自己新修改的部分，同时在项目报错时，也能够及时地进行版本回退，避免到最后整个项目都不能运行的问题。同时Git在对项目不同模块的合并时也有智能化的优势，只需解决冲突，大大降低了工作量。

3.前端玄学

因为之前自己是开发后端的，对于前端技术没有很深的了解，在这次开发中学到了一些前端技术，之前想要使用bootstrap框架，但是发现，这个框架也是要基于大量H5以及CSS的知识，所以最后还是使用基本的技术。同时决定前端开发和后端开发最大的不同在于后端开发如果出现错误就直接报错，具有逻辑关联，但是前端技术并不是，前端即使没有达到设计的效果，也能正常运行，但是就会变得很奇怪，觉得自己这方面的知识太过薄弱。

4.服务器内容

因为之前自己就有服务器，就想把项目部署到自己的服务器上面，但是之前也没有接触过，通过这个过程学会了WAR包的使用，Xftp连接远程服务器安装Tomcat、Java环境,Xshell对于Linux系统的服务器进行命令行操作等。同时在这个过程中遇到的最大的困难就是项目访问远程服务器的MySQL以及本地Navicat访问远程服务器的MySQL，一直以来的执念就是，项目放在服务器项目肯定不是使用“localhost”来访问，但是最后查了很多博客之后发现就是直接使用localhost来对MySQL进行访问，之前自己无法访问的原因是端口设置错误，以及服务器防火墙设置的问题。

5.好的IDE的重要性

之前自己一直使用的是MyEclipse,但是发现这个IDE经常出现卡顿现象，而且进程突然被锁住无法打开，导致花费很多时间在解决IDE问题上面，这样非常延误项目进度。觉得以后还是要使用新的软件工具来辅助开发（例如IDEA），增强效率。

阎娜项目收获

1.我之前主要把学习中心放在前端。由于对后端的不了解，在合作时难免因为各种原因对合作的小伙伴产生一些抱怨，产生消极心理，影响项目的进行。通过这次项目，对后端进行了简单的上手操作，发现后端的难度确实很大，所以我能更加理解彼此的难处，能更加和谐的合作，然后也能更加有效的沟通交流，提高项目效率。  
 2.合作一定要提前沟通好各个细节。这些不只有项目的前期构想，数据库设计，还包括很重要的前端页面设计的工作，这不止是前端的事情，还包括后台的事情。如果后台要的功能前端页面没实现或者实现的方法不一致，那么综合的时候还得重修删改，同理，如果前端的页面功能后端没实现，那么前端做的这个页面用不上，这些废掉的页面的心血和时间就浪费了，所以，提前进行有效的沟通十分重要。

赵梦宇项目收获

1.协调数据库设计的时候对于考试，试卷，试题，考生几个元素之间的关联设计比较混乱，最后认为存留冗余的情况能使得映射关系更清晰，但是数据量大的时候其实还应该更精简；

2.对于模块的划分应该更明确界限，否则调用传参时有时会逻辑模糊；

3.软件并不只是代码工作，更多的是设计，管理与维护，写好文档对于自己和其他成员都是一种方便，并且代码风格和注释也是一种规范，大家按同一种规范开展项目更容易一些

4.原本设想的是不用框架，完全用原生代码实现功能，加深自己对web项目的理解，但是实现过程中的复杂程度远远超出了我们的想象，非常深刻的理解了框架的优越性和存在的必要性，就比如说数据库交互层的实现，如果用hibernate框架，虽然配置起来可能比较繁琐，但是用框架两三行代码就可以实现的功能，要用原生代码写个十几甚至几十行

5. 同一种功能可能会有不同的实现方式，比如考试题目的设定，可以直接限定试题数，也可以动态添加，在考虑和决定最终方式的过程中，也可以感受到不同的思维方式碰撞出来的火花