Lessons Learned

Our top three

1. From a design standpoint, we could have implemented another design in the layers under the "overall" MVC design. It was late to implement the design afterwards due to us having to alter the code that then became unfit for another design.
2. The coding of the system could have gone smoother if we had a better plan for parsing the data, but overall, everything worked and are encapsulated. The testing of the system should focus more on integration tests rather than unit tests as our method logics are mostly sound but the problem occurred when they were put together.
3. The top lesson learned is to start the project small and build up on a working version first. We coded too much prior to putting everything together, resulting in our group having spent too much time on debugging the code everyone put together. Other lessons are set deadlines earlier than the actual deadlines so we can comfortably finish and having a better design/idea of the backend.

Documentation of System Before/During vs. After

Team Pear found that documenting system design before/during system implementation had several merits over documenting after system implementation. Our team did both; we documented our system design before implementation started and added to the documentation as we coded. Documenting in this way helped us to stay organized, create maintainable code, and stay flexible. Before we started coding, our team met and discussed design patterns. This allowed us to have a starting point to work from, and helped us to map out the different areas of our system. We also started drawing out UML class diagrams and wrote out the header files we needed. Documenting design in UML class diagrams and header files were incredibly helpful in creating modular, maintainable code with high cohesion.

Our team members were able to work on separate files without losing track of how the different classes interact. As we were coding, we kept adding to our documentations depending on how our program changed. Code implementation will inevitably deviate slightly from the initial design as issues previously not thought of arise. We followed the big design decisions made at the start such as the design patterns and most of the classes. We added to our UML documentations as we added to the classes from Stage 1 to Stage 2. Documenting system design during the implementation allowed us to have flexibility while keeping our system organized. Documenting system design after system implementation would not be as helpful as what we did because it would have no impact on how we code. We would be disorganized and our code would not be as modular and maintainable.

Project and Team Analysis

Our team organization was well structured. Our project managers were quite clear with the expectations of what was to be completed at the start of the next weekly team meeting. Everyone was held accountable for their role; as a result, everyone performed. Our project performance emphasized consistency. Consistently meeting our self-imposed deadlines and also our ability to adapt to unforeseen problems was excellent.

Interactions with the customer went.

Interactions with the customer we're smooth. We found that they were available for potential questions and were quick to reply via email. The only hiccup occurred during demo submissions. We had some difficulties getting the executable to run on other computers. After we determined the root of the problem, fixes were applied and updated versions were sent. We appreciated their understanding while debugging these issues. The customer was timely with their demo evaluations. In fact, they provided some meaningful and guiding suggestions that helped shaped our final project.

Our Recommendations

For the next project, student should have regular meetings with the TA. The TA can then advise keep the group on task and assist with their project. Students should also have in class tutorials related to the software tools that they will learn and use in their project. There should also be a tutorial time where students can learn or use as a meeting time. A forum should also be use on owl where student can post question. Class material should also be posted on owl. The next project should also be targeted with a new client where students get paid or have reference letters.

Learning experience reflection

This project has been a learning experience in many ways. First and foremost, working with a group of 10 has been particularly challenging; finding group meeting times, and organizing who is working on what, were two issues that could be troublesome. After code was written, integrating everyone's work always posed problems when it came to compiling. Learning to overcome these obstacles was a great experience. Secondly, it was a great experience to have so many opinions to work with when finding the best solution to a problem. Overall the group project was important to learn group dynamics, collaboration and task allocation.