This document covers the challenges faced by the team at UltraLite Coding, and the lessons we learned from them. There are three sections of challenges:

* Teamwork, Communication, Methods & Practices
* Software Design, incl. Test-Driven Development & Agile
* Software Development, Commit Log

*Teamwork*

The challenges we faced with teamwork were primarily about how well we followed our plans. We always broke up our work for each iteration in a way we all agreed was fair – the issues we faced were with people not following through with the plans that were made. This problem was especially critical during Project 4, leading up to the presentation. Some group members had not completed (or even started) their work until the weekend before it was due. The lesson we learned was to more clearly break up our work and make it very clear when everything needs to be done (creating milestones that could show definite progress) – instead of just assigning things at the start of the increment and hoping everyone did their parts from there on out.

*Communication*

The chief challenge we faced with communication was with some team members not appropriately communicating their needs and their situation – we had some people wait far too long, until it was practically too late, to ask for help. We also had people waiting on other people to finish their work before they could do their own work (which was reasonable), but we were not aware that they were waiting so we couldn’t accommodate them. Overall, the biggest lesson we learned is that we all need to stay in better communication with each other and more clearly (and consistently) communicate where we are at.

*Methods & Practices*

The challenges we’ve faced with our methods and practices have been concerning how we have broken up our work and gone about connecting the various pieces each of us have worked on. We struggled to break up the work in a way that would lead to the overall success of our group -perhaps if we had changed some of our methods or practices we may have had some more success. Of course, hindsight is 20/20.

*Software Design*

The biggest struggles that we’ve had, as they concern the design of our software, has been in accommodating the ways we wanted to design our program with the real-world limitations of having other classes, jobs, and commitments to family and friends outside of this class. We would have loved to have gone completely Agilewith how we developed our software, but instead we had maybe once-a-fortnight meetings and somewhat regular contact via text – instead of proper scrum meetings/etc. The biggest struggle with design specifically was with test-driven development: we wanted to develop our software in according to that design pattern but unfortunately it was not always possible. This is mainly due to the massive sprint we had to get the project done before P4 that simply didn’t allow time for building tests first. It was also especially hard to build tests first when we didn’t have a good idea of what our code would even *look* like. The biggest lesson we learned was that we should have designed our system from the top-down before we even wrote any code, and after we had done that (but before we had implemented any functionality) we should have written all our tests.

*Software Development*

The biggest problem we faced with the development of software was the translation of lessons learned in class to actual coding practices. We felt that so much class time was spent on design, but at the end of the day no matter how many different methodologies and design patterns we learned we ended up only being able to accommodate some of them, especially given that the hardest part of this project for us was not designing the code but instead knowing what code actually had to be written (given our almost-complete lack of experience with UI’s, graphics, databases, and servers). The lesson we learned from this was that we should have begun implementing our system (i.e.; writing our code) long before P3 – that would’ve been the only way we could have actually implemented the lessons learned in class in a full and wholesome way.

*Commit Log*

The commit log for our repository is an interesting one – it shows a consistent pattern of rises and falls, with most commits/week being made the week before the iterations were due, and with the least commits/week being made the weeks in which we were presenting. However, the most immediately attention-grabbing thing in our commit log is the massive spike in the week of the November 12th. All other weeks average somewhere around 20 commits/week, but that week saw 156 commits. This was due to the problems I’ve hinted at above, where we had group mates not even begin their parts until that weekend – and their parts were the most critical parts for that iteration. Ultimately, as a team, we surmounted this challenge and went on to create a product of which we could all be proud.