My senior design project is based around a technological void that currently exists within the caving community. Specifically, my project will be about designing a surveying device that can be used in the harsh environments of caves, while providing more functionality and features than other solutions. One aspect of both my education, as well as professional experience through co-ops, that I have enjoyed is designing software that is very low-level, and fills a very specific niche. Many modern software solutions attempt to solve too many issues all at once, and can often lead to scope-creep and lackluster features. This project will be about solving a very real and specific issue, providing a useful device for surveying caves, that seeks to remain simple yet robust. Particularly, allowing for long-range distance measurements, intuitive angle calculation, a simple calibration system, and integration with cardinal directions for map orientation.

While Computer Science courses tend to focus a bit more on high-level languages and concepts, there are also many courses which I believe have prepared me for this kind of project. Namely, CS 2011 (Introduction to Computer Systems), EECE 2060C (Digital Design), EECE 4029 (Systems Programming), as well as CS 5127 (Requirements Engineering). For the first two, these classes provided a starting point for programming and interacting with electronics and low-level software, and gave some insight into how to efficiently design around hardware limitations. While these are fairly surface-level courses that are meant to give a broad overview, these courses sparked excitement in working outside of the scope of high-level software applications. Furthermore, Systems Programming was a class that while not explicitly focused on micro-controllers and electronics, still provided a lot of useful information on how APIs are developed in real-world operating systems. Since this project revolves around building an API that can interact with many different sensors, I believe this class has prepared me for some of the challenges I may face. Finally, Requirements Engineering has taught me a lot about the best practices when it comes to planning out projects, specifically around generating test cases, system requirements, and many other aspects in the planning and implementation phases.

As for my co-op experiences, all of my rotations have been with the Idaho National Laboratory where I have worked as a software engineer. While my work there mostly deal with high-level software applications, I still believe that there are many skills that I have cultivated during these experiences that will be helpful for this project. First and foremost, my

co-ops have exposed me to Agile and Scrum methodologies, which has helped structure the way in which I both plan and develop software. The idea of incremental implementation is crucial in developing robust and powerful software, and I will definitely be keeping this in mind when working on this project. Furthermore, I believe that working in a professional software engineering environment has exposed me to new ways of thinking when it comes to designing software, and will help with planning this project out. While this project may stray a bit from what I have done at my co-ops, I believe many of the concepts that I have learned in practiced during these experiences can be applied to any sort of software engineering problem.

As for motivation to choose this project, there are a few things that inspired me to choose that specific topic. Firstly, during my very first rotation at the INL I was tasked with creating an API that could communicate with older balances that were programmed in a non-standard language, and allow for API calls to be made through a web application. While not necessarily the most design-heavy work, I found it extremely fascinating and exciting, as I was getting to bridge the gap from the physical world to the digital world. It can be easy to lose sight of what we do as computer scientists when most of our work is just running on a computer, so seeing this interaction between low-level electronics and modern high-level applications piqued my interest in a project that would have a similar effect. Secondly, I was having a completely unrelated discussion with a friend who is an avid caver, when we got on the topic of surveying caves. In short, the gold-standard device was known as the DistoX, but it was only able to perform all the needed functionality if you installed a firmware chip that was made by a hobbyist. While this was sufficient for a while, the DistoX was finally discontinued and given a new model, which no longer worked with this firmware chip. All in all, I found this to be an interesting challenge to tackle, as it has a very real application in a field that is not directly related to Computer Science.

From a preliminary perspective, I believe designing a device from the ground-up is paramount in order to truly solve this problem. While designing the device in its entirety would be the ultimate goal, I believe splitting this goal up into smaller achievements is important in order to reduce this large problem into many smaller ones. For example, identifying the sensors that will be used, designing the physical connections between these sensors, developing parts of the API for each sensor, designing the calibration system, and finally designing the physical

device are all smaller goals that will allow for gauging the progress being made. Furthermore, I can already expect having to go through many iterations of this device due to the harsh conditions that are common within caving. While the device may be able to work in a bright, dust-free environment, I expect that many re-designs will be necessary to deal with the dark, wet, dusty, and overall brutal environmental factors that push the physical hardware its limits. Because of this, I believe that this device will have to undergo a lot of testing and revisions that may manifest into new requirements that were not originally expected.