**Final Thoughts Survey:**

**What went well over the course of the project? Why?**

The project was a huge learning experience for me. It was the first time that I worked collaboratively on a coding project. I learned some Python, JavaScript, HTML, Git commands, and how to write a LATEX report. These are all skills that will be very useful to me as a move on in my software engineering path.

**What could have gone better? Why?**

Team communication and load distribution. We did not take an active enough role in distributing responsibilities amongst team members – especially during the first half of the project. Though only 1 team member had the skill to implement the back-end, instead of finding a design that allowed more collaborative programming, we left the back-end entirely in the hands of 1 team member. Though the back-end was developed successful, the goal of the project aside from collecting temperature, is to learn. By not participating on back-end development, the majority of the team still lacks that experience of back-end development.

**Favorite element of the project? Why?**

Documentation and logistics. My favorite part of the project is bringing it all together. I enjoyed most sewing together the work of different members, organizing the source code on GitHub, and creating the report that documents our process.

**Least favorite element? Why?**

Webpage development. I am a stronger Java, C / C++ programmer than I am a JavaScript / HTML developer. There were many JavaScript language specific nuances that impeded the forward progression of the project. For example, the largest problem was computing and returning the average of all the temperatures for a given room. The data retrieval, the algorithm (sum and divide), the response all worked individually. However, when attempting to get all 3 elements to work in conjunction, we came across issues value returned always being 0, or only working with static data – not useful.

**What are you biggest takeaways from this project? Why?**

Planning is key. For example, one of the residents who had agreed to allow a Raspberry Pi to collect data in their room unplugged the device simply because the ‘…the LED on the power adapter was too bright.’ This could have easily been resolved with an inch of duct tape. Also, another resident simply accidently knocked the power adapter out of the wall and did not notice until weeks later. As a result of the devices being unplugged, there was weeks’ worth of unrecoverable data that was never recorded. Lack of data can compromise the integrity of all of the data, and the project as a whole. Had we planned better for human error, we may have placed the Raspberry Pi computers differently, or created a simple flag that notified us of a Raspberry Pi that failed to report back for a given period of time.

**If you could do this project over again, what would you do differently? What would you keep the same? Why?**

In regards to the way that the Raspberry Pi computers communicated with the computers, I would have made it through the world-wide-web, rather than SUNY New Paltz’ intranet. We relied on static IP addresses, a team members job as the network administrator, and several other “hot-fixes” that allowed us to deploy the sensors faster. However, this is disadvantageous in more ways than it is advantageous. Firstly, without a SUNY New Paltz authorized network admin, gaining static IP addresses is impossible – therefore rendering this implementation unrealistic. Secondly, this did not allow other team members to participate in the back-end development of the project. This is not only is an academic disservice to the students who cannot participate, but also unfair to the only member of the team capable of resolving the networking issues of the project. This decision however, was made with earliest deployment time as first priority, for the purpose of collecting the maximum amount of data.

If I were to keep anything the same, it would be the format of the data, both locally on the Raspberry Pi’s SQLite3 database and the remote server. Returning the data as a JSON object was useful in processing the data becomes a simple task. Most programming languages can parse JSON, and many libraries, such as Google Charts, can easily take that JSON data and represent it nicely via graphs and charts.

**If another team were asked to complete the same task, what recommendations would you give them to increase their chances of completing a successful project?**

I would emphasize to that team the importance of organized planning. Although collecting, logging, and graphing data may seem like a simple process, organized planning allows for better preparation for unexpected events.