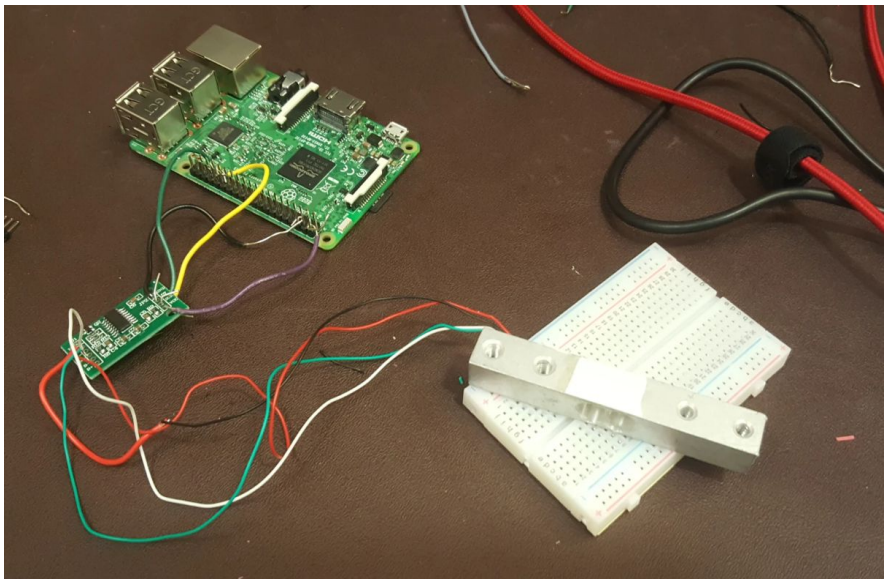
Sparti

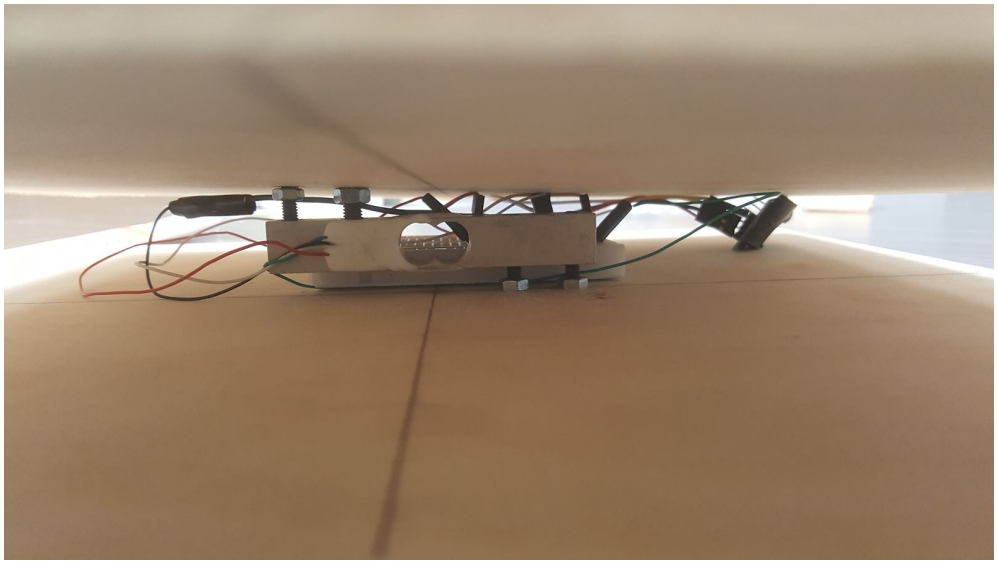
Members:
Kavan Mally and Raghav Rao

Description
The goal of the project was to turn a desk into a connected ‘hub’ of information with data that would be useful to a student at a glance. This was largely inspired by David Rose’s concept of Enchanted Objects and Amber Case’s Calm Technology. The challenge was to make something fully configurable without being difficult to learn. This meant getting as much data from context as possible, to reduce the amount of explicit information a user needed to provide. Therefore the shuttle times and laundry were mapped to different light gradients, and an android app was developed to determine when the user had woken up. A load cell was connected to another raspberry pi to communicate the weight of a laundry basket with the desk lights.

Plan
For our project, we tried doing tasks by descending complexity. We started by working on a complete walking skeleton for the midpoint feasibility. This meant we had to get a minimum working scraper for laundry and the shuttles, get mosquitto to work on android, and to communicate with the the pi. Past the midpoint feasibility, we began with trying to solder together the load cell and figure out the code to read the weight and publish. We then worked on the two web scrapers for shuttle times and laundry, and integrating them with mosquitto and the rest of the system. Finally, we worked on replacing the lamp_ui with our weather UI and implementing the mosquitto listener to work with the previous components

Process/methods
- Load Cell:



    - The first iteration was prototyped by connecting the load cell, load cell amplifier and raspberry pi via breadboard and just used sample code from a github repo
- 
    - We then worked on modifying the code to work with mosquitto and calibrating it for our specific load cell, soldering together most of the parts. We then used ThinkBox to create a platform to measure the weights, since placing items directly on a load cell meant gave wildly varying results depending on how off center it was.

- Kivy:
    - We repurposed the original UI to display weather related icons instead of the lamp sliders. We settled on a four panel design to display the four most relevant questions one might ask on weather data: “Do I need a coat, umbrella or boots today?” and “What is the coldest it will get?”. We used the openweather api to get the weather data, and clipart off Google to indicate what apparel the user might need. We used two sets of images, active and inactive that were selected depending on whether hard-coded criteria for each was met.
- Laundry:
    - We created a web scraper that goes through LaudryView’s website on a hard coded room and fetch the number of available washing machines. The website is a javascript site however which made it harder to parse. Still, we were able to produce a list of available machines. However, due to how the site is setup, we had trouble extracting estimated times from the website.

- Weather:

    - We created a script that called on Open Weather's web api. These web calls give us an overview of the current weather and allow us to tune our microservices accordingly.
- Bus:
    - We created a script that called on a web api that gives us a json dictionary of all the bus stops on Case campus as well as their estimated arrival time. We parsed the data and created a lighting script that listens to when the bus is coming and changes the lampi lights based on how long the bus will take.
- Android:
    - To interact with our services, we wanted an android application to start the process. To simplify the process, we decided to utilize an already existing open source alarm app and inject MQTT calls to an awaken topic in the dismissAlarm routine. This would signify to us that the user is awake and ready to receive information. Due to an open bug from 2017, we had to manually maven build the Java version and import as a dependency to the project.
- MQTT infrastructure:
    - With so many microservices, we need a way for them to communicate with each other. To that end, we had our scripts publish and subscribe on an agreed upon topic list and had appropriate services listen in on specific topics. We also prepend these topics with MAC address to avoid collision with potentially other devices.

Results

- We had mixed results. The software sections of the projects worked successfully, but the hardware was variable. Due to finicky nature of the circuit we've had days where it would work perfectly and other days where it wouldn't give us meaningful data. This interfered with our microservices that relied on its data to trigger them and operate normally. This unfortunately interfered with our goal of a seamless enchanted experience. Otherwise our microservices worked as expected.

- Load Cell:
    - Unfortunately the screws were two sets of metric screws M4 (bottom) and M5 (top), and the ThinBox donated screw sets did not have too many metric options. Both sets of screws are 20 cm long, but to get proper cantilevering, longer screws are needed. We could not find the right screwdrivers since M4 falls somewhere between 1/8th and 7/64th allen keys which was the closest we could find. Time was lost using needle nose pliers to tighten the screws which damaged the wood and the loadcell. We also drilled one of the holes slightly off center and to fix the stubborn screws, replaced one of the M5's for an M4, leading to wobbly results.

Conclusion

Kavan: I learned how flexible and usable MQTT is for projects such as these and how easy it is to include it in these. I also learned more about how much effort goes into makes these isolated components work together with each other. Finally I learned about some of the frustrations of open source community such as Android's Paho package, which still has an breaking bug from 2017.

Raghav:

The most important lesson was never to pick a hardware project for a final. I learned how to solder, read schematics, debug hardware and to manipulate an existing project to fit my needs. In hindsight I should have given up on the loadcell earlier and focused on polishing and integrating the rest, possibly adding alexa integration.