A lot has changed since our first design proposal. Our initial reaction to working from home was that we should continue with our original project idea, and have somebody at school ship us the hardware that we ordered. This idea is impractical because we do not know if it is possible to get our hardware delivered to us, and if so, how long it would take. This will leave two thirds of the team without the platform they need to build the system on, which we feel is too big a risk.

Web application controller:

A web application will be used to monitor and control the system. Users will be able to add a new device to the network, examine system state, and define device interactions. The web application will not deviate much, if at all, from the original plan. The backend will still communicate to the databases on the devices to poll system state and declare system interactions.

Interaction Layer:

The interaction layer consists of data storage and communication between the devices. Given that this is largely abstractable from the hardware, and interfaces through the hardware with an interface (see the hardware layer discussion below), the interaction layer can function largely the same as we had initially planned.

One change to the system is that we will now be using qemu (a processor simulator) running on an AWS instance to simulate the raspberry pi's. The reason for this is that our team has some familiarity with qemu, and qemu provides an excellent way to simulate hardware for free. We chose AWS because while AWS, Azure, and GCP all provide comparable functionality and cost, we are most familiar with AWS.

The main changes to the interaction layer are our requirements.

- Device commissioning: Since we no longer can have all devices on the same physical network, and will be using virtualized devices in the cloud, we need to hardcode the IP addresses of the devices. As such, we will not be developing a "commissioning" process to add new devices to the system.
- Internet resiliency: Since we are simulating "local" devices in the public cloud, we can't manually disconnect them from the internet and show that the functionality remains. While cloud service providers like AWS offer subnet abilities to simulate a local network and disconnect it from the rest of the public internet, we feel that we can demonstrate this requirement without using subnets. Instead, we can demonstrate that all incoming / outgoing network calls are from / to devices in our "network".
- Sensor input to device action latency: We previously required this to be <100ms. However, given that the public cloud has unpredictable latency, we will also be removing this requirement. In particular, as everything moves to virtual during the pandemic, public cloud services could see greater strain, leading to potentially more unpredictable latencies.

Hardware Layer:

Since acquiring hardware will be uncertain with classes in this state, we plan on virtually simulating hardware using a webapp. This will encompass a software interface for the interaction layer to connect with and a webapp that is connected to that interface. The interaction layer will use the same function calls as it would to the hardware previously, but through this virtual interface. This will allow us to do all of the demos and testing virtually so that we can all work remotely.

Since we will no longer be able to accurately test latency, we are removing latency as a requirement for the hardware.

We are adding interface and web app requirements to the hardware part however. Once we further delineate the interaction and hardware layers, the requirements for those two are very clear. For example, if the interaction layer calls the gpio library, the requirements for the hardware interface would be all of the functionality of the gpio library.

In the hardware webapp, we will need the ability to have intuitive interactions with these devices, display multiple devices on the page at once, and easily control the virtual “home environment” around the devices.

We will also look into ordering smart devices and using their APIs as interfaces for our virtual devices. This would allow us to have the physical demo without having to develop circuitry remotely.

New sources of risk:

A major risk of switching to the smart devices and using their APIs would be that we might not have a way to get the smart devices with the current situation, meaning that we would not be able to use them. To mitigate this, we are planning on developing all of this with the goal of a hardware webapp but feel out how long it would take for these devices to get shipped to us.

Revised Gantt Chart:

Team A6 Capstone Project Schedule

Rip Lyster, Richard Deng, Niko Gupta

Project Start Date: 1/13/2020 (Monday) | Display Week: 10

WBS	TASK	LEAD	START	END	DAYS	% DONE
1	**Research Design Elements**					
1.1	Research Decide on Devices	Rip	Mon 2/03/20	Fri 2/07/20	5	0%
1.1.1	Decide what interactions those devices will have	Rip	Sat 2/08/20	Sun 2/09/20	2	0%
1.2	Research Storage System for Devices	Richard	Mon 2/03/20	Fri 2/07/20	5	0%
1.2.1	Decide on Storage System	Richard	Sat 2/08/20	Sun 2/09/20	2	0%
1.3	Research Interaction System for Devices	Niko	Mon 2/03/20	Fri 2/07/20	5	0%
1.3.1	Decide on Interaction System	Niko	Sat 2/08/20	Sun 2/09/20	2	0%
2	**Draft Main Systems**					
2.1	Hardware Design and Order for MVP Devices	Rip	Mon 2/10/20	Mon 2/24/20	15	0%
2.2	Webapp Framework Design Decisions	Everyone	Mon 2/10/20	Wed 2/12/20	3	0%
2.2.1	Webapp Framwork Demo	Richard	Thu 2/13/20	Mon 2/24/20	12	0%
2.3	Interaction Layer Design Decisions	Everyone	Mon 2/10/20	Wed 2/12/20	3	0%
2.3.1	Interaction Layer Demo	Niko	Thu 2/13/20	Tue 2/25/20	13	0%
2.4	Design Presentation Work	Everyone	Fri 2/21/20	Sun 2/23/20	3	0%
3	**Build / Integrate Main Systems**			-		
3.1	Test Hardware Devices	Rip	Mon 2/24/20	Wed 2/26/20	3	0%
3.2	Write Hardware Interaction Layer	Rip	Thu 2/27/20	Sat 2/29/20	3	0%
3.3	Integrate Webapp and Interaction Layer	Niko & Richard	Mon 2/24/20	Sat 2/29/20	6	0%
3.4	Integrate Hardware and Interaction Layer	Everyone	Sun 3/01/20	Wed 3/04/20	4	0%
	PIVOT POINT					
0.1	Build hardware web application	Rip	Mon 3/23/20	Sat 4/04/20	13	0%
0.2	Build hardware interface	Rip	Mon 3/23/20	Sun 3/29/20	7	0%
0.3	Design and Build Interaction Layer	Niko	Mon 3/23/20	Wed 4/01/20	10	0%
0.4	Build Web Application and dockerize	Richard	Mon 3/23/20	Wed 4/01/20	10	0%
1	**Second Integration Phase**			-		
1.1	Integrate New Hardware with Interaction Layer	Rip	Thu 4/02/20	Wed 4/08/20	7	0%
1.2	Integrate Commissioning to Webapp	Niko	Thu 4/02/20	Wed 4/08/20	7	0%
1.3	Create Demo Interactions	Richard	Thu 4/02/20	Wed 4/08/20	7	0%
1.4	Free Space, Extra Time	Everyone	Thu 4/09/20	Sun 4/12/20	4	0%
2	**Time Padding**			-		
2.1	Working on Project Report/Presentation	Everyone	Mon 4/13/20	Sun 4/26/20	14	0%
2.2	Work on Finalizing Demo	Everyone	Mon 4/13/20	Sun 4/26/20	14	0%
2.3	Work on Added Features	Everyone	Mon 4/13/20	Sun 4/26/20	14	0%