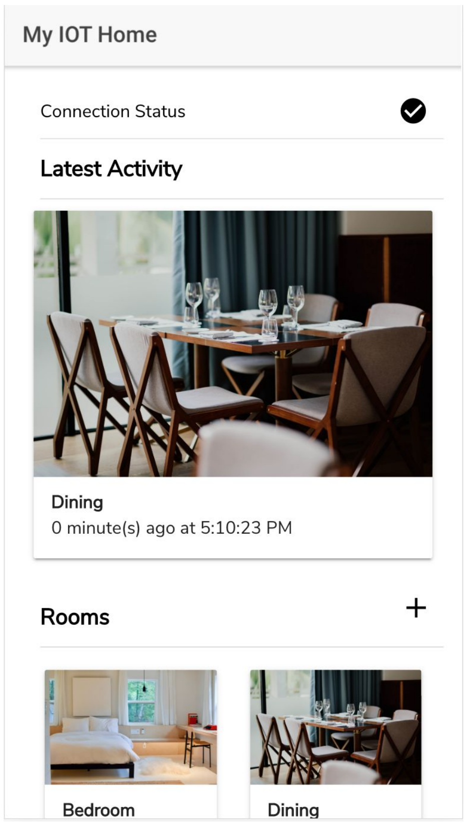
Brice Wilbanks
Assignment 3 Report
October 7, 2019

App UI/UX Decisions

UI Overview

Since the mobile app has limited functionality, the UI is important to present the limited amount of information in an intriguing and useful way. To achieve this, the mobile app uses images and other elements to add eye-catching components to the mobile app. Additionally, with changing data, the presentation method of this app had to also dynamically change the UI. There were multiple UI considerations for these special cases.

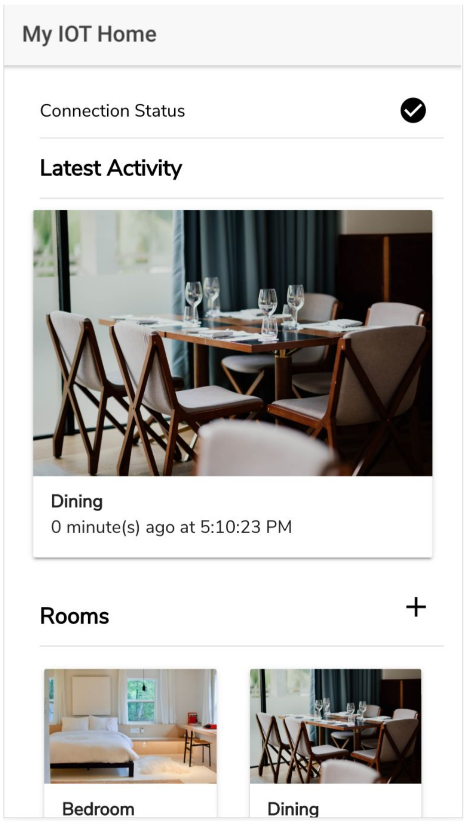
UI Decision One - Single Presentation Screen

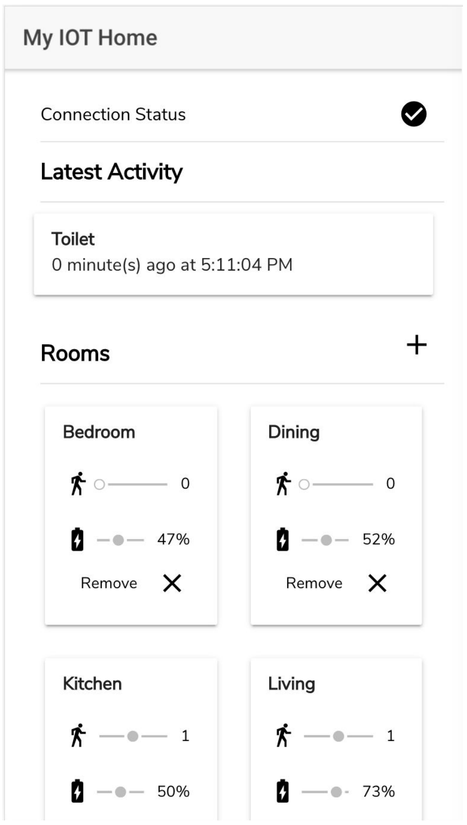


Current UI
Room Images (Image 1a)

This UI decision was the question on whether to include another screen for users to monitor their devices (motion sensors). Under the assignment specification, the requirement described two screens, one for the latest activity and another for a list of rooms/devices with specific device information. However, before development, it was planned that this mobile app would separate these elements into two separate components on the landing page of the app. The upper screen of the mobile app was setup to show where the last activity of the elder was recorded. This primary feature of the mobile app does not require a whole screen, and if placed on its own screen would create unused whitespace. To address this whitespace, additional app functions were placed on the same screen. For example, if the user would like to use the apps secondary and tertiary functions, they can scroll down to reveal more information about the rooms and devices, including battery and historical tracked activity. Hence, the one screen design worked well for this specific mobile app. There has been one screen throughout the apps lifecycle and therefore there is no alternate UI screenshot.

UI Decision Two - Room Images vs No Room Images





Current UI
Room Images (Image 2a)

Alternate UI
No Images (Image 2b)

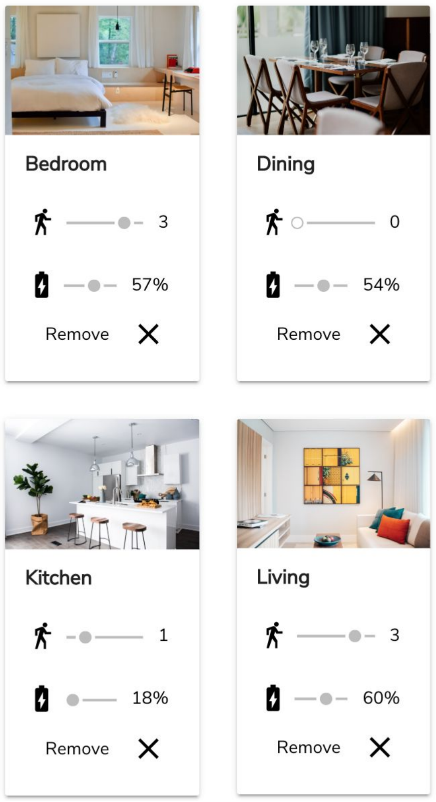
Discussion of Current UI - Images / Non-Images
One major UI consideration was the implementation of room images for the data. The mobile app has a minimalistic material design feel, hence the coloring and design of the mobile app needed elements to make it more alluring to the user. Images for each room added this necessary color and feel to the app. There were multiple practical reasons to include the images. First, users were now able to recognize the rooms where the elder had motion quicker. Users tend to recognize images first over text, and with this modification, users are able to skim with “Rooms” section of the app and see the activity occurring. Second, the images add an increased sense of comfort to the users of the app. Since the elder is living in the house pictured, the user of the app feels like they are actually in the house monitoring the senior. The pictures add a sense of realism not conveyed through text and numbers. Third, and finally, the images add

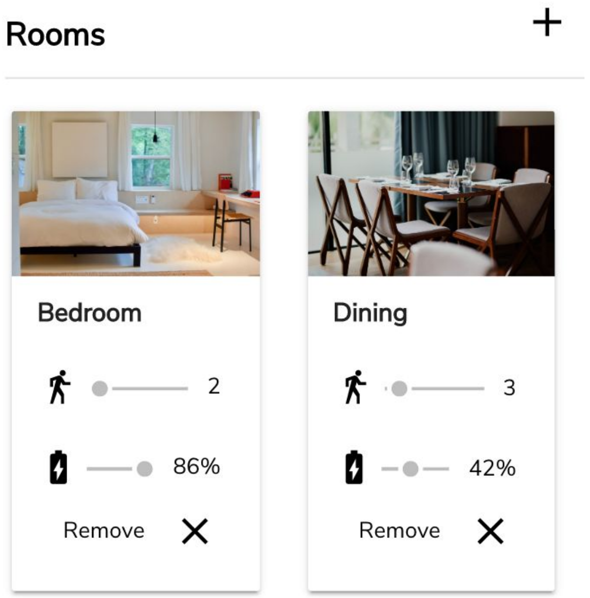
personalization to the app. In a complete build of the app, users would be able to upload their own images for each room, adding to the personal feel. The app is built around the user and not the data.

The images are currently borrowed royalty-free from the Unsplash.com photo library. To make demoing and testing the app easier, the set of five images were used and linked to through an exported images class. As mentioned above, in a full build and a production release of the app, users would be able to upload their own room images to personalize the app.

Discussion of Alternate UI (Image 2b)
In this version of the app, there are no images displayed for any of the devices or rooms. This alternate UI shows what can be achieved with just text and numbers being fed from the IOT broker. While the mobile app looks sufficient without the images, the images add the feel and personalization needed for the mobile app. Under the textual UI, the only indication that a room has activity under the "Latest Activity" component is the text "Toilet". This is a small indicator for the latest activity and users might miss its message. The image adds space to the component for the caregiver to quickly recognize which room the elder was last using. Scrolling down to the "Rooms" component of the app, the same dilemma occurs; it is hard to quickly recognize which room the battery and activity level is referring too. The image added the needed recognition.

UI Decision Three - Activity Slider Max Value





Current UI
Dynamic Activity Slider (Image 3a)

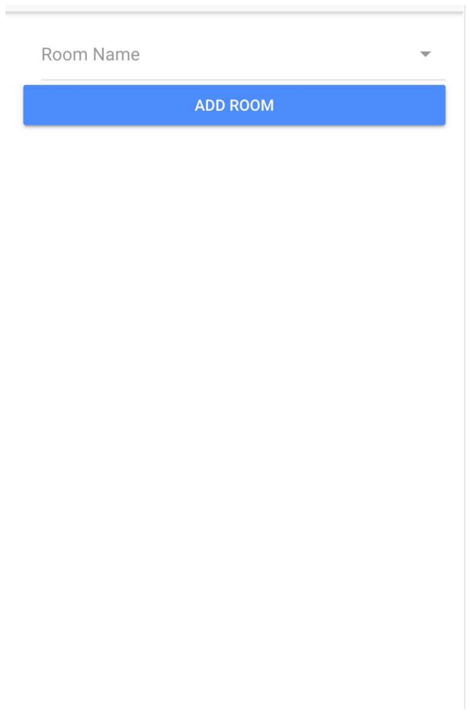
Alternate UI
Max Activity Slider (Image 3b)

Discussion of Current UI
In this UI decision, the max value of the individual room activity monitor was considered. Looking specifically at the activity section of the room cards at the slider with the person walking icon, the max value of this range changes dynamically based on how much motion has been detected in total. For example, in image 3a, the most activity depicted has been three motion detections in the bedroom and living room. This has been accurately displayed by showing the slider range about 75% of the way full. The max value accepted by the slider is the room with the most activity plus one. In another example, if there were 30 movements detected in the living room, the max slider range for all rooms would be 31. Under this dynamic max range, the user could quickly see how much activity a room had received without manually comparing the activity numbers. This acts almost as a histogram, however, each datapoint is contained under the rooms individual card, making for a cleaner UI.

Discussion of Alternate UI (Image 3b)
Under this alternate UI, the max slider was set to some predetermined max value. This would be useful on a case-by-case basis where some app users may want to know how much a room is being used compared to some predetermined value. However, for this mobile app, having a preset max value made the

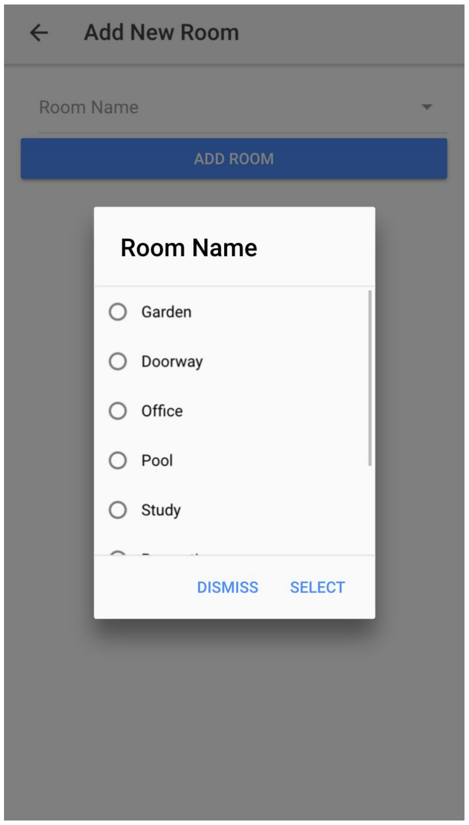
comparison between rooms tricky. The user generally wants to know how much a room is being used compared to other rooms in the house, not an arbitrary value. Additionally, this arbitrary value would need to be changed based on the individuals use. For example, if a particular elder is more active than another, the same max value would have the activity slider range maxed out for one elder and under utilized for another elder. There would be no max value that would work for all use cases. Hence, the dynamically changing slider was used.

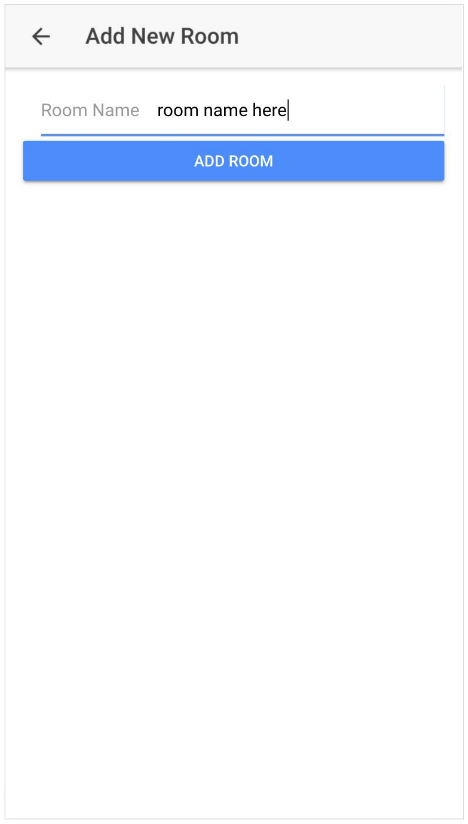
Self-Designed Screen: Add Room (2)
For the self-designed screen requirement, the mobile app contains an “Add Room” screen. This screen allows the functionality for the user to add rooms and devices to the elder’s smart home app. Users are able to select a room and add it to the “Rooms” component on the homepage. While this feature is purely aesthetic since the data stream from broker cannot be modified, it shows what a full production version of this mobile app could look like. While the room page is simple, there were several UI considerations taken into account.



Screenshot of the Current Add Room UI

List of Rooms / User Defined Rooms





Current UI
Selection of Rooms (Image 4a)

Alternate UI
User Defined Rooms (Image 4b)

Discussion of Current UI
In the current version of the mobile app, users are able to select a room name from the dropdown menu of common rooms and spaces found in homes. The reason a drop down was selected for this mobile app was a pure functional consideration. In order to make the rooms match the rest of the app, they also needed room images. However, since all the room images are predefined for this preproduction mobile app, these rooms would also need predefined images. In a full build of the app, users would be able to upload an image from their device that is a photo of the room the sensor is in. However, this is not the case for this assignment based, non-production mobile app. Therefore, a list of common rooms and associated images had to be used. Using a text-based, user defined approach would have caused issues for linking to rooms. First, images could be sourced from an images API, however the issue with this is searching the API for relevant photos considering user input. The API may not have image results for specific user inputs. Additionally, the mobile app would need to use text-scrubbing for security. For the time allotted and expected in the assignment specification, these features would not have been feasible. Because of these reasons, a dropdown selection UI was used.

Discussion of Alternate UI (Image 4b)
Under the alternate, the mobile app would allow users to choose their own room names and upload a photo for that room. Additionally, since the users would be using their own broker and IOT home, the user would also have to connect the device to the mobile app via this page. The technology and UI would look similar to now by having users use a form to input information. All-in-all, the main reason this UI was not used was the photo consideration. In order to make the added rooms work with the mobile app feel, the images needed to be added.

Discussion of Possible Issues (3)
There are multiple issues that specifically can affect this smart, elder-care home app. These issues stem both from issues with the IOT devices themselves and from specific mobile use issues that are present in all mobile apps, regardless of IOT integration.

Device Connectivity Issues
Since IOT devices are limited in their function, the loss of connection one or multiple devices to the broker can deem the units useless. In this ender care mobile app, the app depends on constant data coming from the units and the broker to ensure the safety of the elder. If the units are offline, the broker may respond to the server, and associated mobile app, that there is no motion, even though this is not the case. This could be addressed with a protocol change as discussed in part four of this report, but assuming we use MQTT, we need to account for this issue. The sensors are programmed to only return either one or zero, indicating whether the room is in motion or not. With this limited data delivered from the sensor, the broker must make assumptions to fill the gaps. In the case of an unexpected disconnect or initial connection issues with the sensor, the broker may assume that no response is the same as a zero response, indicating no motion. In the eldercare example, this would be used to trigger a 'no-motion' alert, even though the sensors are at fault. Therefore, this must be considered and failsafes must be designed to prevent one faulty, offline sensor from affecting the mobile app results.

Issues with a No-Motion Response
If the broker makes the assumption that there is no motion in a specific room, this contributes to the data of all the rooms. In the eldercare mobile app, if there is no motion in all rooms after a specified time interval, the mobile app triggers a notification indicating the lack of movement. If a sensor is offline, there is no way of knowing whether or not the room is occupied, thus affecting the results as a whole. The solution to this problem is relaying to the app users that the sensor is in-fact offline and not factoring the offline sensor's data into the motion calculations. While some users might find it obnoxious that the sensors are not properly connected, this would prevent false positives or negatives given to the user. With a notification indicating the loss of connection, the user does not receive a false message stating that the elder is safe or unsafe. While not ideal, this solution handles the issue of a lack of connection to the sensor.

IOT Device Battery Issues
Since many of IOT sensors and beacons are small and low-profile in design, they may lack the space to have a sufficient battery or power connection. These devices usually use a coin-battery with a short life. Because of this, the actions of the beacon and sensor need to be limited in order to conserve battery life. This relates directly back to update rates of data being sent from the sensor. There is a trade-off between battery life and sensor accuracy that must be observed. In order to use the sensor to its full ability and use, the battery life would be depleted at an increased rate. With an increased battery depletion rate, the sensors batteries would need to be switched out more often, leading to additional cost and frustration experienced by the elders and caregivers. Developers and IOT System Architects must keep this in mind when designing an elder's smart home. If developers request updates from the sensors too much, the battery will be drained, but if they do not request updates from the sensors enough, the data will not be

sufficient enough to feed the mobile app. The best way to solve this problem is finding a sensor options that allows a reliable and steady data stream from the sensors without substantially draining the battery.

Sensitivity & Other Motion
Another issue that IOT sensing devices, specifically motion sensors face is false positives or negatives for motion. Take an example of animals in the house: motion sensors sense all types of motion, regardless if it's natural, animal, or human. Because of this, the sensors will relay positive motion data back to the broker for all types of movement sensed. One possible solution to minimizing the effect of non-human motion being detected is changing the sensor sensitivity settings. By decreasing the sensitivity, smaller moving objects, such as dogs, cats, and house-pets, may not be detected. Additionally, movement caused by air conditioning/heating drafts would generally not be detected. On the downside, however, movement by the elder may not be recognized under certain conditions. Slow movements by the elder may not trigger a motion notification due to the sensitivity being turned down. System architects and developers must take this into account. Another solution to this is using a combination of other sensors to determine whether the movement was a human movement. For example, noise sensors may also be used to determine if the movement detected by the motion sensor is also making noise. If both are true, the movement detected is more likely to be a human movement. The basis of the internet of things is the ability to gather large amounts of data and determine some outcome or conclusion. In this case, using multiple home sensors to determine the elders health is preferable. Other examples of sensors may be door sensors or temperature sensors. Sensors are discussed more in length below.

Placement of Sensors
A major consideration that IOT architects must take into account is the placement of sensors in their respective places. Poor sensor placement can lead to false or incorrect data being sent to the broker. In the eldercare smart home example, a poorly placed sensor may not pick up on certain motion of the elder. This can pose the same issue presented above where the sensor sensitivity is set too low. False and inaccurate data can lead to issues in the design of the software. Poor sensor placement is especially an issue for heat and temperature sensors. If a temperature sensor is placed in a naturally cold or hot part of a room, the overall room temperature rating will be false. For example, placing a temperature sensor near the doorway may cause it to read colder while placing the sensor in the ceiling may cause a hotter reading. If the temperature is controlled with this data in mind, the false readings would cause the heating or cooling system to improperly heat/cool the room. This has been a common criticism with the Google Nest Thermostat when it is placed poorly.

Lack of Other Sensors
The final issue presented in this report is the lack of other sensors present in this specific eldercare smarthome. Relying on just motion sensors comes with the major issues presented above. Additionally, with just motion sensors, the main purpose of the app, ensuring the safety of the elder, is not realized. Specifically, several points of the elders day-to-day life are not considered. First, if the elder leaves the house, there is no specific indication of this in the architecture. The elder being away from home is the same as inactivity according to the current system. One solution to this is door sensors. Adding sensors that send data to the broker when a door is opened could indicate whether or not the elder is actually home. The second issue with the current architecture is the lack of ability to see when an elder is sleeping.

Similar to the issue with elders leaving the residence presented above, when the elder is sleeping, there is no motion being streamed to the broker. The solution to this, is again, adding an array of different sensors to the residence. In this sleeping case, a noise sensor may be used to detect snoring or a temperature sensor may be used to detect body heat in the bedroom. More of these sensors makes the eldercare smart home come to fruition. However, without these sensors, the functionality of the current architecture is limited. Adding additional data would help the system become fully useful.

MQTT Critique and Alternative (4)

MQTT Overview

When looking at Internet of Things architecture, one popular protocol used for message transmission between IOT devices is MQTT (Message Queuing Telemetry Transport). The protocol transfers byte sized data packets over low-bandwidth connections. The minimum size of a packet is only two bytes. This combination of low-bandwidth connections and small data packets makes the MQTT protocol seem perfect for IOT devices. Under MQTT, the small payload of the message can contain data to control, request, or change data from an IOT beacon, sensor, or device. Additionally, MQTT has specific features to subscribe to data streams from a specific broker. Subscribing to a data stream allows the IOT device to constantly emit information and allows the developer to use this data stream. Subscriptions specifically came in handy with the sample eldercare mobile app in this assignment. Another key benefit of the MQTT protocol is its scalability and flexibility. MQTT works with a small and large number of nodes. Because of its lightweight nature, MQTT is incredibly scalable to thousands of devices. One MQTT broker can subscribed to a number of devices. With these benefits, MQTT has become the ISO standard for IOT device data transfer.

MQTT Critique

However, MQTT is not without its major flaws and security vulnerabilities which cause issues for large scale and secure IOT projects. John Rinaldi, Chief Strategist and CEO of Real Time Automation, an IOT expert, described a few major issues with IOT in his article posted in 2018[1].

1. First, he described MQTT's issue with data formatting and lack of interoperability. Under the current MQTT protocol, data is transferred via an array of bytes. This is all the data that is transferred. The issue, however, is not the form the data is transferred but rather the encoding. There is not a standard encoding method used to format the data. Because of this, there is also no encoding method common for subscribers to that data. This becomes an issue when transferring data between the sensor and the hub. Looking at the eldercare mobile app, the payload which contains a timestamp and an integer may have issues if the mobile app subscriber does not properly format that data. The subscriber may expect a format undeliverable by the MQTT protocol and the data is therefore uninterpretable. Therefore, data encoding management becomes an issue for MQTT.
2. Another major issue with MQTT is the lack of status messages delivered by the devices. This means that subscribers to the data are unaware of the status of the devices changes. If the devices comes offline, comes back online, is permanently removed, or is congested, the subscriber has no way of knowing if the device is still active. If a user or device subscribes to an offline device, there is no notification to indicate the devices status, making debugging and connection challenging to determine. To make matters worse, in some cases, the last message sent by the device is stored and sent to any new subscribers, making the device appear to be online even if it has been offline for weeks. MQTT does not have a standard for ensuring that subscribers are properly subscribed to a topic.

[1] Article: https://www.rtautomation.com/rtas-blog/whats-wrong-with-mqtt/

3. As for security, since MQTT is a TCP Application layer protocol, it requires TLS encryption. While TLS is not a poor security protocol, it has its drawbacks with used with MQTT. TLS requires a lot of bandwidth to initiate and confirm the handshake between two devices. This added bandwidth negates the benefit of MQTT being lightweight and low in bandwidth. TLS adds latency to the process of sending messages, which works against MQTT lightweight nature[2].
4. Lastly, MQTT does not allow subscription discovery. This means that there is no way to see the streams of data coming from a specific device. Since there is no way to see the different subscription options of a device, the subscriber must rely on knowing the topic level and topic name to subscribe. These strings must be shared between all subscribers in order to receive data from the data stream. This method is archaic and forces the setup of the device to know a specific, public string. Data discovery could allow subscribers to be dynamic in changing subscription topics.

MQTT Alternative

A proper MQTT alternative would take the best parts of MQTT and address the downfalls efficiently. The best parts of MQTT mostly relate to its lightweight, low-bandwidth nature. Its flexibility and scalability come as secondary related benefits to the lightweight primary benefits. With that, an MQTT alternative would need to have these two features. To address the downfalls, an MQTT alternative would have a lot to tackle.

Data Packets

As mentioned above, a new framework would keep the lightweight and low bandwidth nature of MQTT. In order for this to happen, the data packet design would have to be similar in size to the current MQTT architecture. This means that data packets must be at minimum two bytes or less in size. This two byte size would ideally also contain all the control information, such as origin, status, and device information. With this new data packet being developed, we can solve one of the major drawbacks of MQTT: the lack of status messages. With a few more pieces of information on the data packet sent from the device, we can include device information and status, such as a MAC address, subscription options, and status messages. For example, a data packet can now contain information about whether the device is online or offline and its uptime over a past period of time. All of these data points can be used by the subscriber and the devices themselves can be serviced if necessary. While this data requires additional bytes of data, we can minimize or condense the information sent therefore minimizing the overall packet size. Additionally, even if this data requires more bytes of data, faster internet connections being currently developed and installed would be able to handle slightly larger data packets.

Payload Formatting

Along with the base data packets being changed, the formatting and encoding for the payload portion of the packet would need to be altered. For an example, we can look at the Advanced Message Queuing Protocol (AMQP)[3]. This protocol is an alternate to MQTT and tackles the drawback of a lack of interoperability between clients and brokers. Under their protocol, AMQP translates data payloads on

[2] TLS Article: https://www.hostingadvice.com/how-to/tls-vs-ssl/
[3] Overview of AMQP: https://www.digitalocean.com/community/tutorials/an-advanced-message-queuing-protocol-amqp-walkthrough

both the client and broker ends. This translation at so called "message brokers" allows any device to communicate with each other. The major drawback observed with MQTT was the encoding of data between two devices. With this solution provided by the AMQP, different IOT devices can now be connected to each other, regardless of encoding.

Security

The last MQTT drawback to consider would be the security of this new protocol. Since TLS is industry standard and is quite secure, it may be considered more insecure to move away from a well established security protocol, regardless of additional latency. In this new protocol, the drawbacks of moving away from TLS outweigh the benefits gained by lack of latency. Therefore, we would keep the same security protocol.