ESE 543 Final Project Report

May 19thth, 2020

Smart Hub

Asif Iqbal, Roni Das

Faculty: Shan Lin

Smart Hub is an android application that is designed for smart home devices. This application provides centralize control for all home smart devices such as smart light bulbs, thermostats, smoke detector, garage door, security etc. Smart home devices are considered IoT devices for their capability of connecting to online services. Smart Hub provides an endpoint for these data in a GUI based application. User can add new IoT device to the application home and have control to all their essential wireless devices.

Problem statement includes today's smart devices are vendor dependent. Meaning, every IoT device has its own software solution. For example, Philips Hue has an app named Hue. Similarly, Google's nest uses Google's proprietary software. This poses a problem when user have multiple such smart devices from different vendors. One must switch back and forth between several dozen applications. Although, it might make sense for business end, where companies forcing customers to adopt their ecosystem. On the other hand, customers has less option to choose from. Smart Hub solves this problem by combining all smart things into one. All smart devices have standard protocol, such as MQTT, which they use to communicate to local or online servers. Since, the communication methods are standard, there is little to no barrier for different smart things to be controlled by one application. Smart Hub provides a getaway for this.

Project specification includes many well-defined ideas along with constraints. Few such constraints are budget and time. Smart Devices, especially if new on market, are prohibitively expensive. To mitigate this problem, we devised a solution where we make our own IoT device on a budget. In order to accomplish this goal, we required a cheap Wi-Fi Chip, such as ESP32 WROOM, along with the sensors.

This IoT device consists of multiple sensors. The purpose of them is to collect raw data. Using a Wi-Fi module, this data is then sent to the cloud. In this case, the cloud provider is AWS. Using MQTT and AWS's IoT Core service, getting the data to the cloud was successful. Using a java-based phone application, that data is hen presented to the user in real time. The parts used in this project are Wi-Fi module, Combustible Gas/Smoke Sensor. Motion Sensor. An application to communicate with these devices, Smart Hub, was designed in Android studio. Note that the raw cost of the parts above comes out to be a little of $30, making our system low cost.

Now let's look at the design of the system.

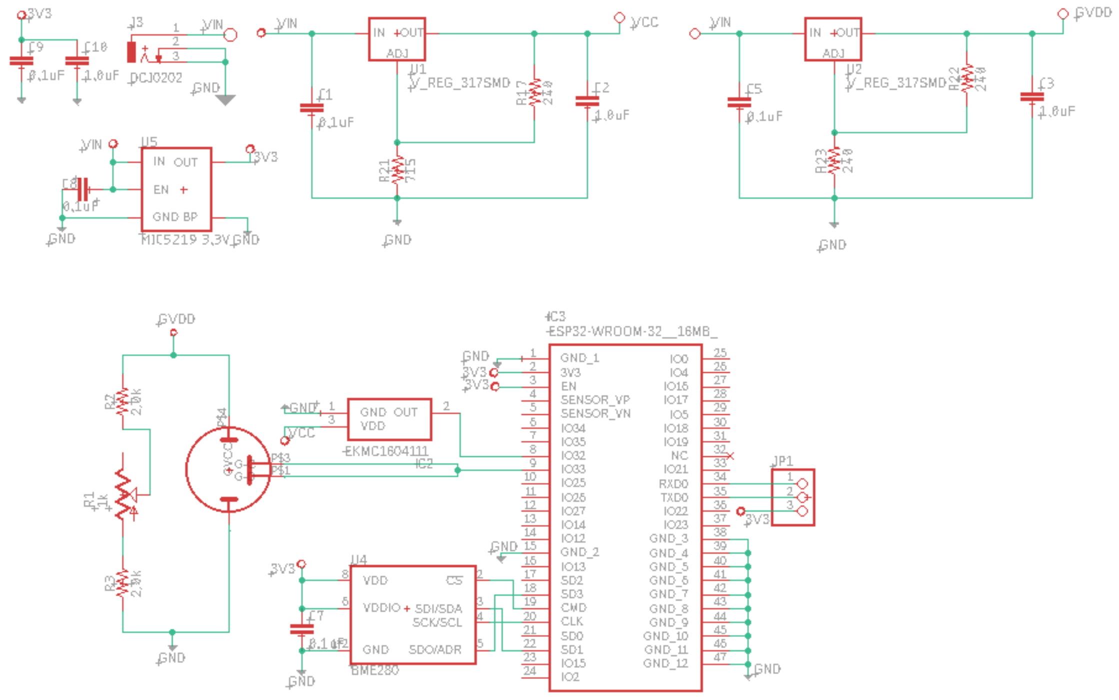


Fig. 1 : Top Level Schematic for IoT Sensor Node.

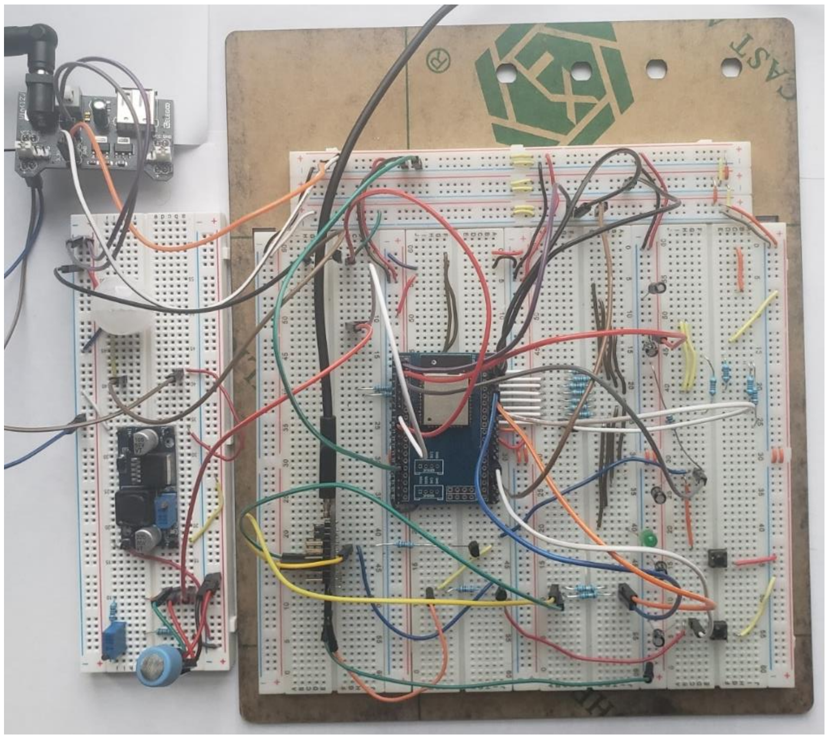


Fig. 2 : Prototype of top -level Schematic.

Both Figures above, describes smart Sensor Node for data entry. The purpose of this node is to connect to AWS Cloud services in transferring collected data.

Let's get into the application design. Before anything, we need to understand how IoT Core in AWS works and why it is idea for IoT devices. The way to make IoT work is by creating what is called a "IoT Thing". It is basically an IoT object in AWS IoT Core environment. This is a necessary step in order to link an IoT device to AWS. The Thing needs a "Policy" attached to it. Policy dictates what this Thing can do. For example, the IoT Thing connect with other devices, can it subscribe and/or publish to other devices? These rules are set by the Policy. In our case, the properties for our Policy is as follows: connect, subscribe and publish. "Connect" allows any device to connect to our IoT thing. This this case it would be the phone application and Wi-Fi module. "Publish" allows the Wi-Fi module to send data to AWS via MQTT protocol. "Subscribe" allows AWS to send data, in real time, to Smart Hub phone application using MQTT protocol. This protocol is network protocol which allows devices to communicate with each other via messages.

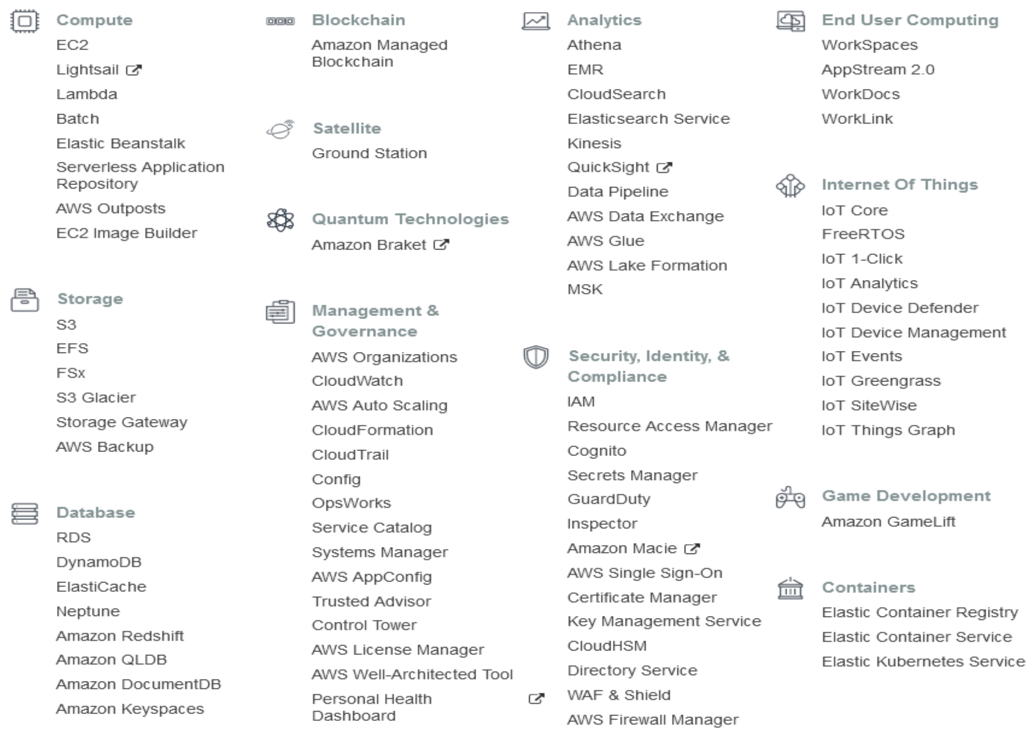


Fig. 3: Services offer by AWS Cloud.

In order to connect to AWS services, few configurations need to be made. Figure 3, shows all the services offered by AWS. Under Security, Identity & Compliances, configuration of IAM Role is made along with providing Cognito. IAM roles defines who has access to the smart device and what are the permissions. Cognito provides access to AWS services without user Authentications, meaning user do not require sign up. This option is selected to operate data processing in serverless mode.

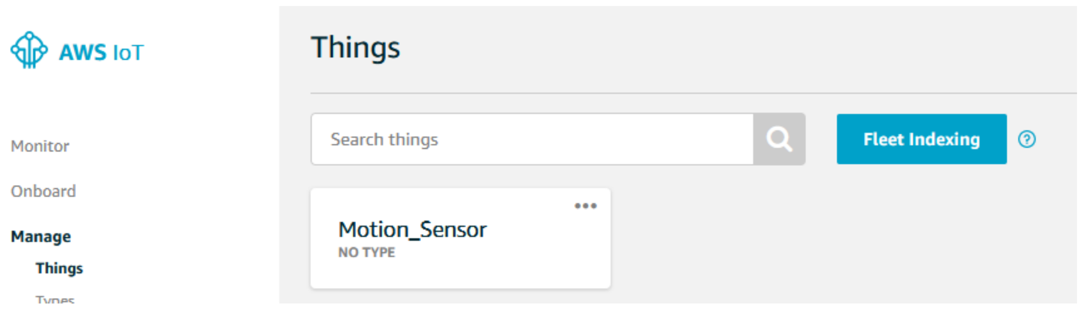
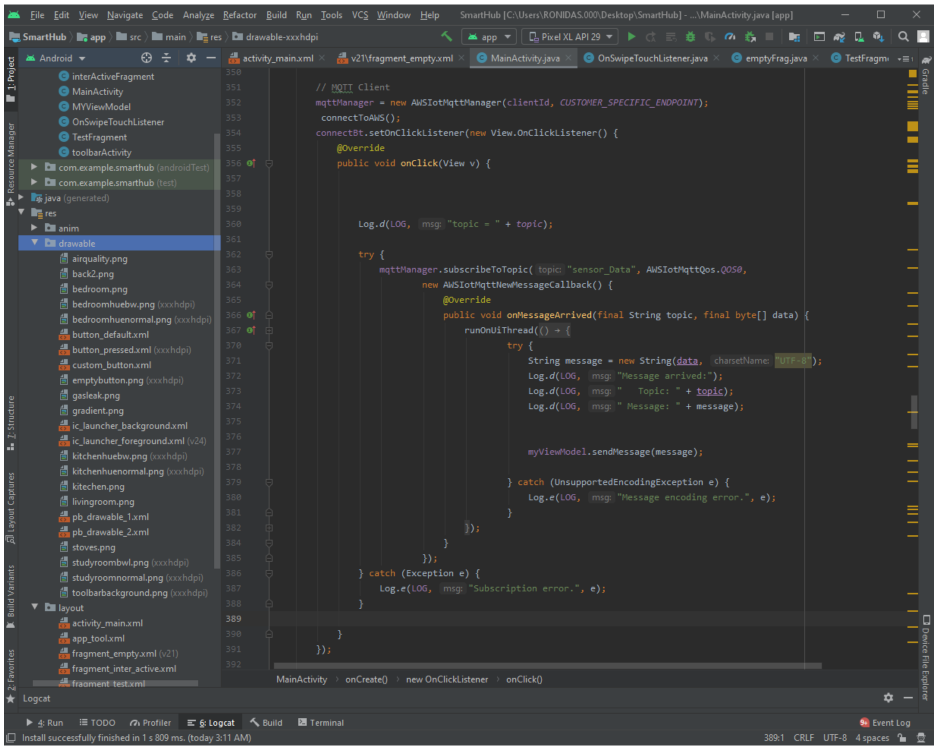


Fig. 4: A Smart Things “object” in AWS cloud.

Application development for contains many processes which were undertaken throughout this semester. Figure below, shows a snapshot of Android Studio view during app development process. There are 7 Java Classes and 5 Layouts and their functions very by context. Figure 6 shows the user interface for the App. Mainly, 4 cards displayed on screen are for separate fragments classes with respective views. They each look similar, rectangular button style, however with different content inside. Main Activity implements lister Interface to communicate with fragments. Figure 5 displays a subscribing method in connecting to AWS server. Method subscribe to a Topic named “sensor_Data”, meaning any IoT home device publishes to this topic will broadcast their raw data to Smart Hub.



```
        // MQTT Client
        mqttManager = new AWSIotMqttManager(clientId, CUSTOMER_SPECIFIC_ENDPOINT);
         connectToAWS();
        connectBt.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {



                Log.d(LOG,  msg: "topic = " + topic);

                try {
                    mqttManager.subscribeToTopic( topic: "sensor_Data", AWSIotMqttQos.QOS0,
                            new AWSIotMqttNewMessageCallback() {
                                @Override
                                public void onMessageArrived(final String topic, final byte[] data) {
                                    runOnUiThread(() → {
                                            try {
                                                String message = new String(data,  charsetName: "UTF-8");
                                                Log.d(LOG,  msg: "Message arrived:");
                                                Log.d(LOG,  msg: "   Topic: " + topic);
                                                Log.d(LOG,  msg: " Message: " + message);


                                                myViewModel.sendMessage(message);

                                            } catch (UnsupportedEncodingException e) {
                                                Log.e(LOG,  msg: "Message encoding error.", e);
                                            }
                                    });
                                }
                            });
                } catch (Exception e) {
                    Log.e(LOG,  msg: "Subscription error.", e);
                }

            }
        });
```

Install successfully finished in 1 s 809 ms. (today 3:11 AM)

Fig. 5: A Smart Things “object” in AWS cloud.

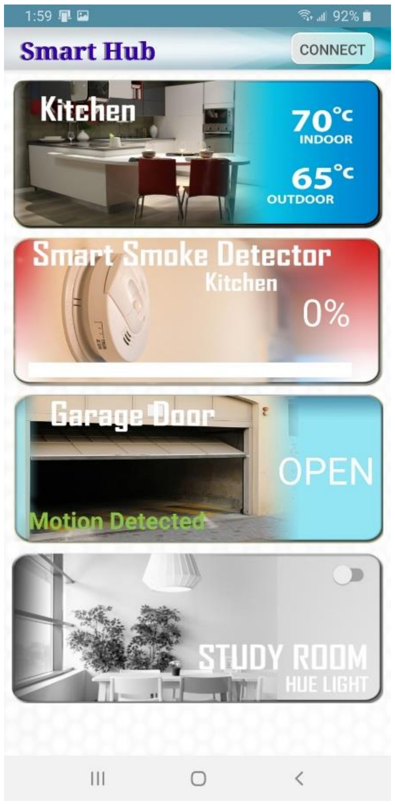


Fig. 6: User Interface.

Smart Hub supports gesture control such as swipe left or right. Additionally, user can potentially add new cards on view. This represents adding a new IoT device to the system. Cost reasons prohibits us to commit to that extent.

In Conclusion, the goal of this project is to determine the necessary steps required to develop a centralize application for all smart home devices. Although, smart devices are expensive and vendor specific, it is possible to test the idea with off the shelf-components, on a budget. This project does exactly that. A custom IoT platform was created to gather raw data. Using MQTT protocol and AWS cloud services, data was transferred to a serverless online service, IoT Core. Android application Smart Hub uses the same protocol to collect data from online. Separate view groups expended on collected information and neatly present it to user for further control. At the end, any smart home device uses MQTT protocol can connect to Smart Hub android application.

Additional Resources:

YouTube App Demo	: https://youtu.be/9qFteB_qdyw
Android APK	: https://bit.ly/2LJr5G8