

Advanced Hardware Design
ECE - 6372
Connected Home System

Table of Contents

Introduction.....	3
Bill of Material.....	4
System Architecture.....	5
Design Consideration	6
Code	7
Demonstration	13
Conclusion.....	13

1. Introduction

The IoT based Connected Home System is becoming quite popular across the world. One can monitor the activities inside and outside of his house at any time with a single tap on phone or click on Computer in real-time.

In this project, we have simulated a small-scale home automation system in which we have taken certain parameters as inputs of the system to get the real-time data from these devices and integrated it with our Google firebase database and Android Application which can not only be used for monitoring of those parameters and detection of motion but also can be used for controlling those parameters. The two parameters we are focused on in this project are motion, and temperature and humidity. Temperature and humidity are grouped into one category as their parameters are tied to one function in the system. While both parameters are used as monitoring tools to determine if there is motion detected or what the temperature and humidity readings in the room are, both have a separate function that adds to the automation of the house. The motion detection can also turn on lights when detected, and the temperature and humidity reading gets sent to the database to be stored for monitoring purposes.

The components we used include Raspberry Pi 3B+ as the primary controller with three different sensors, namely Motion Sensor, Temperature & Humidity Sensor which are generating digital and analog inputs for our controller. We are also using LED light to detect motion and glows when temperature & humidity goes above the specified range. Connections are made on breadboard with jumper wires. The detailed architecture is explained in Section 3.


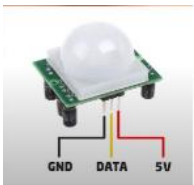



2. Bill of Material - BOM

The selection of components for this project is based on the following factors:

- Cost
- Application Size
- Connectivity
- Resolution

We used the following components shown below in Table 1.

Table 1: Bill of Materials

Sr #	Description	Image	Qty
1	Raspberry Pi 3B+		1
2	Motion Sensor		1
3	Temp. & Humidity Sensor		1
4	Bread Board		1
5	Jumper Wires		Pack

3. System Architecture

Our project consists of three main portions. There is the hardware portion with a raspberry pi and the corresponding peripherals, the Google Firebase database which holds the data for our system, and the android app which provides the user with more control over the system. The figure below shows a broad view of the system.

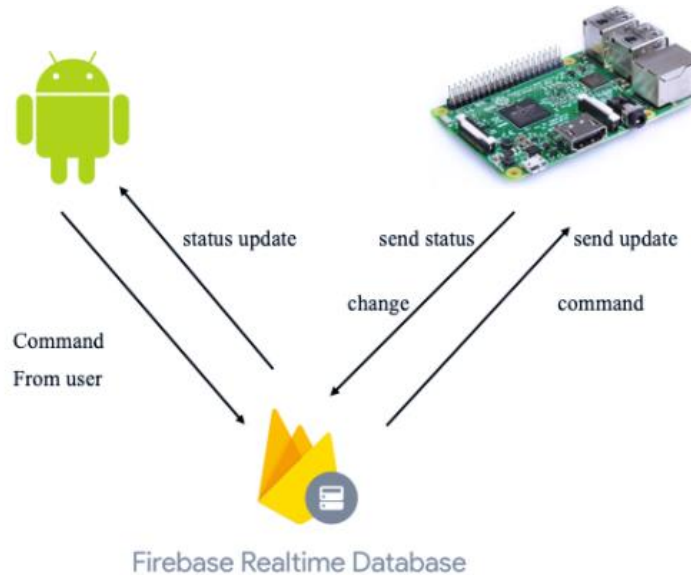


Figure 1: Broad overview of System, and the communication between systems

In terms of system architecture, the hardware components, the raspberry pi, the motion sensor, and the temperature and humidity sensor, are connected together as shown below in Figure 2.

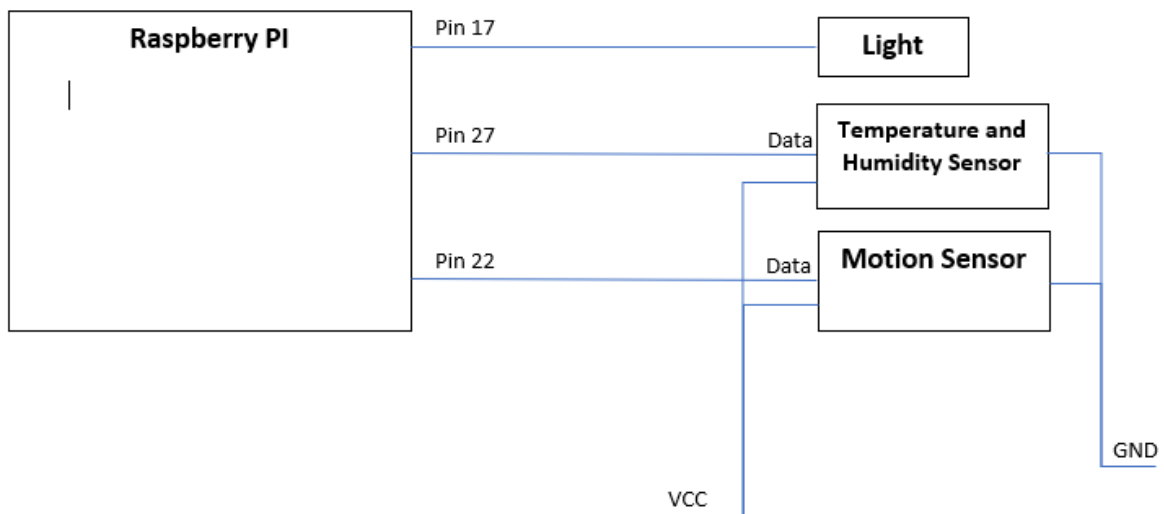


Figure 2: Pin Connections between Raspberry pi, Motion Sensor, and Temperature and Humidity Sensor

Pin 17 functions as an output, while pins 27 and 22 function as inputs to the raspberry pi. The temperature and humidity sensor (TH sensor) and the motion sensor's data pins connect to the raspberry pi's input pins to deliver the data relevant to the system. The TH sensor and the

motion sensor are also connected to VCC and GND to power the peripherals.

Since the system is a real time system, the data inputs are checked in real time and updated in the system consistently. This ensures that the system can react in time to update the status of the system in a reasonable time frame. For example, this allows the lights to be turned on in a short time after the motion sensor sends an on signal signifying that motion is detected.

4. Design Consideration

While the specifications of the assignment imposed certain requirements for the technology and methods being used in our project, there were still many design considerations that were decided by the team. Using a raspberry pi, android studios, and Google Firebase were all decided beforehand due to the requirements of the project. Despite that, however, these choices still provided very valuable design considerations. The raspberry pi is a very powerful tool for a smaller cost than something that is more cutting edge, especially when in regards to our project. The raspberry pi is also multifunctional, in that because it's such a widespread piece of technology, it is very flexible in its functionality.

Some of the design considerations that were conscious choices from our team included the sensors that we are using. We needed sensors that were compatible with the raspberry pi, provided the proper functionality, were cost effective, and most importantly, available now. One of the real world challenges that we were faced with was the global chip shortage. This provided a real look into what sorts of challenges design teams could face unexpectedly. Due to the global pandemic and the resulting supply chain issues around the world, many of the devices we looked into were out of stock or unavailable for an extended period of time. Because of this issue, we had to use what was available and compatible.

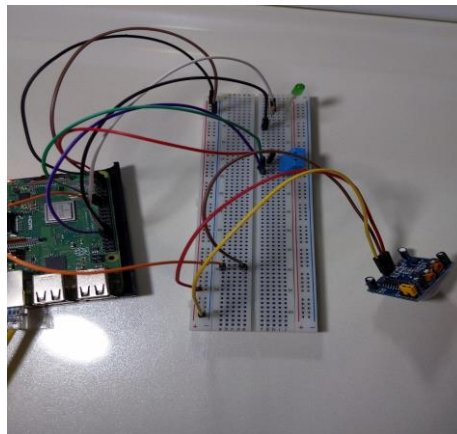


Figure 3: Picture of Materials used in the System

5. Code

5.1. Back-End

For the back-end code for this project we utilized python. Our code can be found in the src folder with the name Python_RaspberryPi.py. This portion of the code is responsible for connecting the raspberry pi to Google Firebase while also providing the logic to the microcontroller for its functionality. There are 2 sensors being used, a motion sensor and a temperature and humidity sensor. These are serving as inputs to the system that provide conditions for the outputs to turn on or off automatically.

When motion is detected by the motion sensor the light functionality in our system is turned on. Once motion is no longer detected, the system turns the lights off. The temperature and humidity readings are taken in real time and stored in the database. The readings are available for the end users to monitor and maintain the desired environment.

Both functions have calls to update the Google Firebase database upon any change in the system. This is to ensure that the proper status of the system is recorded at all times, as there are ways to override the automatic nature of the system that will be further elaborated upon in the front end portion of this project.

5.2. Firebase (Database)

For many Internet of Things (IoT) projects a message queuing system like MQTT (Message Queue Telemetry Transport) is all that is needed to connect sensors, devices and graphic interfaces together. If however you require a database, with sorting, queuing and multi-media support then there are some great cloud storage platforms that are available. One that is definitely worth taking a look at is Google Firebase.

Like any IoT solution, Google Firebase can have inputs and sensors sending data directly into it, and a variety of client applications to view the data (below diagram), but Google Firebase also offers other features such as: file storage, machine learning, messaging, and server side functions.

First of all, we have connected our Raspberry Pi to Firebase. There are a number of Python libraries to access Firebase. The pyrebase library has some added functionality such as: queries, sorting, file downloads and streaming support. The pyrebase library only support Python 3, and it is installed by:

```
sudo pip3 install pyrebase
sudo pip3 install --upgrade google-auth-oauthlib
```

Figure 4: Code to install Firebase

Config settings in the code as below is the integration point between Database and Raspberry:

```

2 import pyrebase
  #import the pyrebase module to communicate with the firebase
3 import time
  #import the time modulde to add delays
4 import Adafruit_DHT
  #import DHT sensor libraries
5 from gpiozero import MotionSensor
6 from gpiozero import LED
7
8 config = {
  #define dictionary named config with several key-value pairs that configure the
  connection to the firebase database
9     "apiKey": "AIzaSyDwWRVMPeHzXoB4pYt_d2JipGyR6Qszlpg",
10    "authDomain": "project-7700667.firebaseio.com",
11    "databaseURL": "https://project-7700667-default-rtdb.firebaseio.com",
12    "storageBucket": "project-7700667.appspot.com"
13 }
14
15 firebase = pyrebase.initialize_app(config)
  #initialize communication with the firebase database
16 #GPIO Setup for GPIO numbering, choose BCM
17 GPIO.setmode(GPIO.BCM)
18 GPIO.setwarnings(False)
19

```

Figure 5: Config code shows the Firebase connection

As a second step, we have created a Firebase project to add Firebase to our Android App. We have listed the steps below detailing how we connected the system to Firebase.

Step 1: Create a Firebase project

Before we can add Firebase to our Android app, we have created a Firebase project to connect to the Android app.

Step 2: Register the app with Firebase

Step 3: Add a Firebase configuration file

1. the configuration file is available once we enter the Android app package name, with **google-services.json** the connection parameters is ready.
2. To enable Firebase in the app, we add the **google-services plugin** to the Gradle files. (Below the plugin details can be found)


```

buildscript {
    repositories {
        // Check that you have the following line (if not, add it):
        google() // Google's Maven repository
    }

    dependencies {
        // ...

        // Add the following line:
        classpath 'com.google.gms:google-services:4.3.10' // Google Services plugin
    }
}

allprojects {
    // ...

    repositories {
        // Check that you have the following line (if not, add it):
        google() // Google's Maven repository
        // ...
    }
}

apply plugin: 'com.android.application'
// Add the following line:
apply plugin: 'com.google.gms:google-services' // Google Services plugin

android {
    // ...
}

```

Figure 6: Code to be added to gradle plugin to ensure Firebase connection to Android

Step 4: Adding Firebase SDKs to the app

Using the Firebase Android BoM, declare the dependencies for the Firebase products that you want to use in your app. Declare them in your module (app-level) Gradle file (usually app/build.gradle).

Below, the related dependencies can be found:

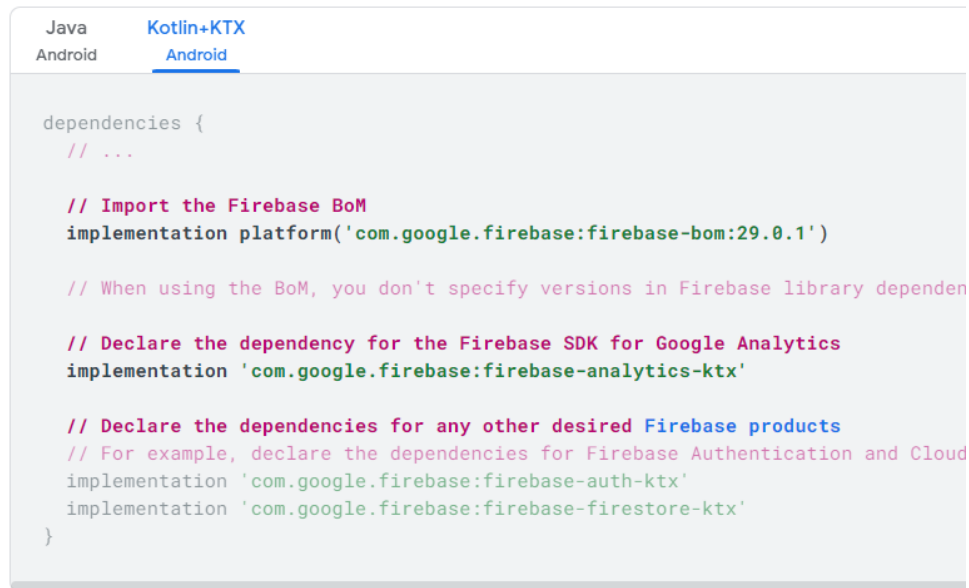


Figure 7: Firebase dependencies

Realtime Database screenshot shows our root directory and childs:

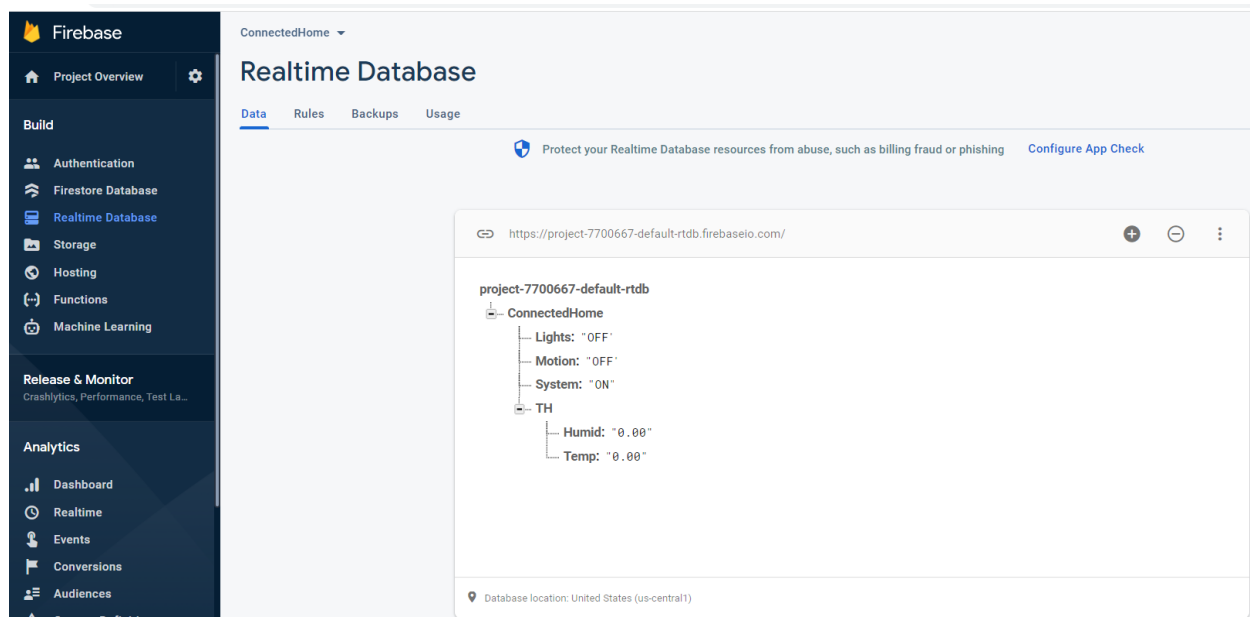


Figure 8: Database Directories

The security is configured in the Database -> Rules menu option. To keep things simple for testing read and write security is set to true for public access

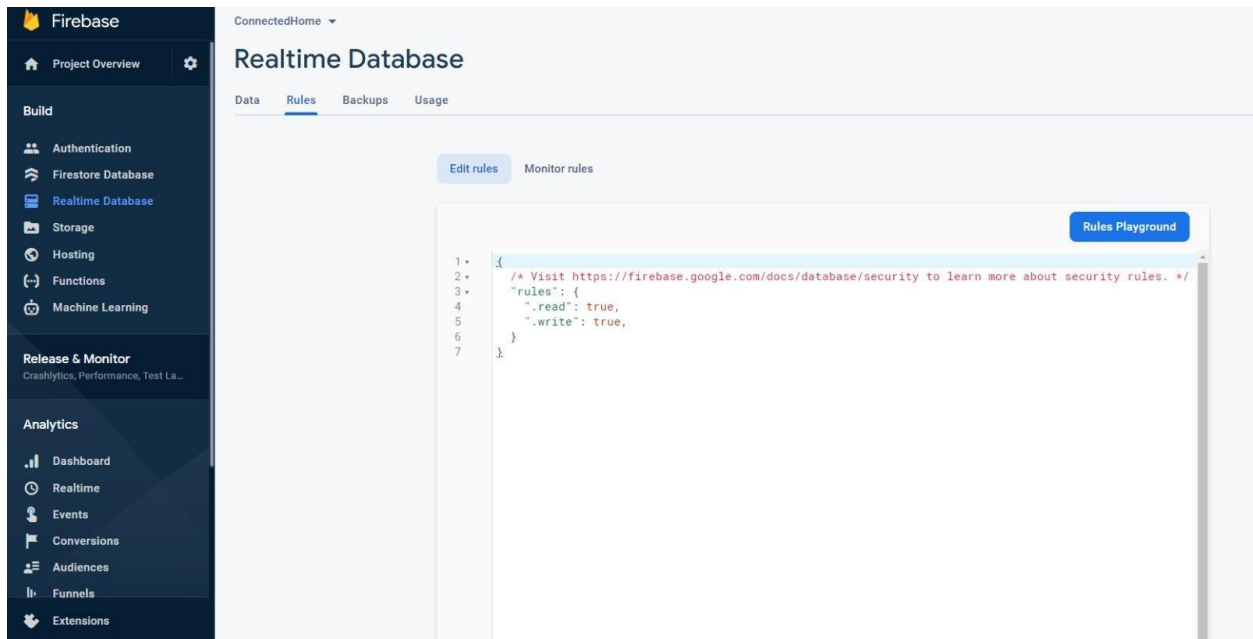


Figure 9: Read and Write access permissions

Below two graphs show how Firebase provides Analytics functionality to follow up the historical activities in the system:

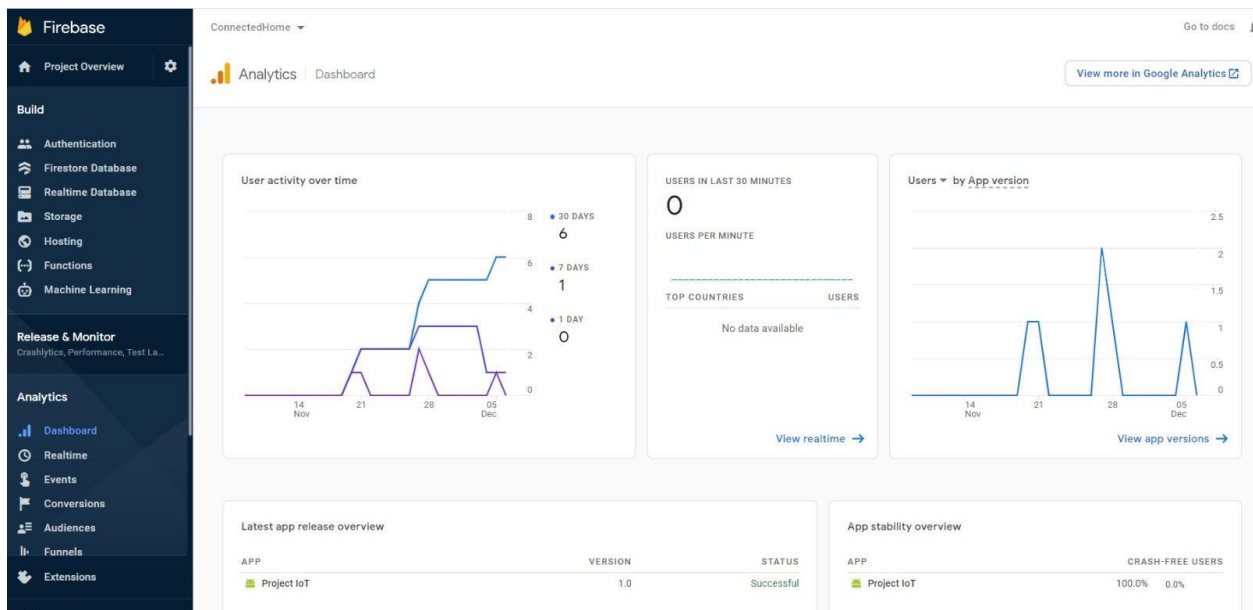


Figure 10: Firebase Analytics

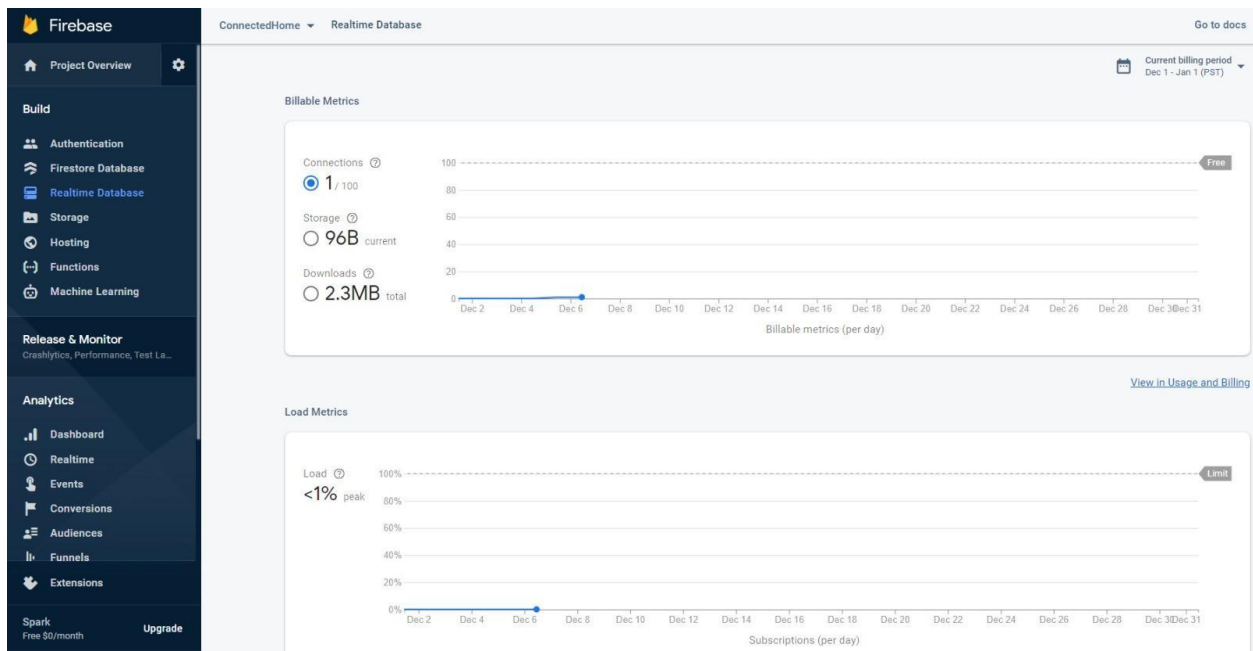


Figure 11: Firebase Analytics Continued

5.3. Front End

The android app is the front end of our project. We developed functionality and actions with Google Kotlin. For the GUI development we used Android Studio's XML development environment. It will mirror the state of the system through a connection to Google Firebase to ensure that it shows the accurate state of the system. The app works as both a monitoring device through its Firebase connection and as a method of overriding the current functionality of the raspberry pi's automated program. The finished UI of the app can be seen below in Figure 12.

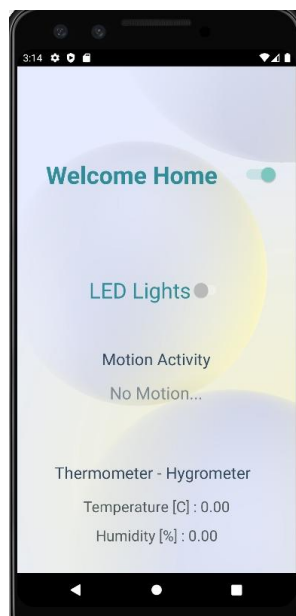


Figure 12: Android App UI

The android app is only connected to Google Firebase and not the raspberry pi, but the overall system updates due to the connection from Firebase to the raspberry pi. In this case, Firebase acts as the go between for the android app and the hardware system made up of the raspberry pi and the sensors, making it seem as though they are connected.

6. Demonstration

The demonstration has been done separately for each part, namely Raspberry Pi and Front & Backend programming. While we have created videos for demonstration that can be found at the end of this section, a brief summary of demonstration can be found below.

We have connected our field devices with the Raspberry Pi 3B+ model using jumper wires and breadboard as it is more feasible for making modifications and re-use of the equipment. The motion sensor is connected to the breadboard using three wires:

- Red wire is connected to 5 volts
- Brown wire is a signal wire
- Yellow is connected to the ground

For the temperature and humidity sensor, the connections are as follows:

- Red wire is connected to 5 volts.
- Green wire is for the signal
- Purple wire is connected to the ground.

The LED with resistor connections are as follows:

- One end of the resistor is connected to the 32nd pin of the microcontroller which is also providing signal
- The other end is connected to the ground.

The built-in Bluetooth and Wi-Fi module on the controller has been used primarily to send the real-time data to our database which is connected with our Android app. We used the ethernet port of the controller for downloading the program and back-end programming is done in Python.

When the system is turned on, it will show the Temp. & humidity readings and if there the motion is detected, it can be observed at the center of the android app GUI under the 'Motion activity'.

6.1. Links

6.1.1. Introduction to Project and Bill of Materials – BOM

https://drive.google.com/file/d/1lo2O_A4PvsAA1c0bHBUnPBQaniaUjAhJ/view?usp=sharing

6.1.2. System Functionality

<https://drive.google.com/file/d/1nnHPvasuVR4ZFkl2HQlwH7F2f0Wck5Tq/view?usp=sharing>

6.1.3. Hardware, Software (Android App) and Database

<https://drive.google.com/file/d/10inpFLU-8eGoc9lipmEwVhzvYewXnQ2W/view?usp=sharing>

6.1.4. Motion Sensor and Lights

<https://drive.google.com/file/d/1Z8nZV3OhsXTrQDCjsdgpmmfVwwznqr9DS/view?usp=sharing>

6.1.5. Temperature and Humidity Sensor

https://drive.google.com/file/d/15kX1SR3CljL6Q5_4yqfhMm-JdLYL1C18/view?usp=sharing

7. Conclusion

In this project we successfully created a system that was capable of monitoring aspects of the home as well as changing them based on inputs. The temperature and humidity of a home was monitored and stored in Firebase, while the motion sensor could detect movement in the home while also turning on the lights in rooms where movement is detected. The system connects the hardware to Google Firebase and Google Firebase connects to the android app, creating a seamless interaction between the hardware system and being able to control it through a phone.

We were successfully able to create this project due to the skills and concepts we learned in our AHD course, and we believe that we were able to take advantage of these skills expertly.