Research Article
Open Access

An IoT-enabled smart elderly living environment

Abstract

With the advancement of smart sensing technologies, embedded systems, open source OS, wireless networks, and IoT gateways, elders will have better quality living environment with self-monitoring health care. The solution we proposed will help monitor aging family members and eliminate the concerns of safety from family members who have a vested interest in their aged loved ones. An open interoperable IoT based platform architecture is a good choice for family members, relatives, and caregivers to remotely monitor elder's sensation data in real time, such as monitoring elders' living environmental quality and vital body sign in real time. Our technological platform prototype will involve pulse rate sensor, indoor environmental temperature and humidity sensor, Arduino and Raspberries Pi embedded systems, Firebase from Google, firebase real-time database, and web App. The developed prototype can collect individual sensational data, store and sync data among all registered users in real time. The experiment has been setup to test the IoT based platform with two sets of embedded computers with the biosensors installed at different locations. Multiple users, including local users and remote users, can concurrently receive the sensational data via our developed web in real time. The information received can inform people about the health and living conditions of the elders who live at the home independently.

Keywords: IoT, arduino, raspberry pi, data acquisition, biosensors, Google firebase, web application

Volume 6 Issue 2 - 2020

Yu Wang, Sunghoon Jang
Department of Computer Engineering Technology, New York City College of Technology, USA

Correspondence: Dr. Yu Wang, Department of Computer Engineering Technology, New York City College of Technology of the City University of New York, Brooklyn, NY, 11201, Tel (718)260-5893, Email ywang@citytech.cuny.edu

Received: December 31, 2019 | **Published:** March 06, 2020

Introduction

The world's older population continues to grow at an unprecedented rate. According to World Population Prospects 2019 (United Nations, 2019),[1] one in six people in the world will be over the age of 65. The number of people above age 80 years is growing even faster than the number above age 65. In 1990 there were just 54 million people aged 80 or over in the world, a number that nearly tripled to 143 million in 2019. Globally, the number of persons aged 80 or over is projected to nearly triple again to 426 million in 2050 and to increase further to 881 million in 2100. Many elderly citizens live in nursing homes or have personal nurses to help them with daily tasks, but for those who are used to be independence, such restraint of freedom can be devastating. It is crucial to develop modest support systems for the elder to live healthily and safely in an independent living environment. With the advancement of smart sensing technologies, embedded systems, wireless communication, and IoT-enabled service, elders will have enhanced quality of their living environment in self-monitoring or remote family-monitoring of health care. The research in this area is in wide range from low-level data acquisition by sensors to high-level data integration,[2–4] including sensing and raw-data acquisition, the communication hardware and software to relay data locally or remotely, and data analysis and learning/reasoning methods to identify activities and provide caregivers and experts with useful and significant information.

The different sensing technology and communication networks have been proposed.[5–7] For example, wearable sensors can be used to acquire data to generate activity patterns for the forgotten complex activities in smart elderly living environment. Wireless and IoT (Internet of things) service can be used to send information remotely to doctors and caregivers for early detection and prevention of depression and diabetes. Accelerometers and an electrocardiogram (ECG) sensor can be used to monitor heart attack. Near-field imaging floor sensor and noise recognition patterns can be used to detect falls. Electromyography (EMG) can be used to help detect neuromuscular abnormalities. A wide range of open-source hardware that includes healthcare sensors and low-cost single-board computers are readily available on the consumer market.[8] Both Raspberry Pi and Arduino are equally popular open source hardware but using different boards. This allows us to improve and contribute to the hardware and software design under an open-source license. An IoT driven health monitoring system using Raspberry Pi was proposed in.[9] The Raspberry Pi (RPi) connects to the sensors of heartbeat, temperature, and accelerometer via GPIO, I2C bus, and extra ADC (analog to digital converter) since there is no analog channel available from the RPi. Arduino is a very low-cost popular hardware/software platform. There are a number of libraries available in the Arduino community to enable the developer easily to access various sensors and modules that are reacting to external digital and analog signals. The hardware of ESP 8266 with Arduino board is considered in.[10] The analog reading from Arduino is shared to the web database via ESP 8266 Wi-Fi Module that can be accessed by the patients or registered doctors. Our previous work combined the advantages of Arduino and RPi,[11,12] in which both hardware development boards are chosen to acquire and relay the sensational data to the Google Firebase. Our current improved cross-connected system is constructed around Firebase from Google, which provide application development platform as Backend-as-a-Service (Baas). We conducted the experiment with two sets of the hardware and software components. Each set of hardware includes one Arduino microcontroller, one RPi minicomputer, the pulse rate sensor, and DHT22. Each set of hardware is used to send a group of acquired data to real time database via Wi-Fi. Arduino programming, Python programming, and JavaScript are used in the IoT enabled cross-connected platform development.

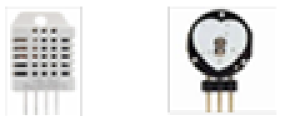


Figure 1 DHT22/AM2302 and pulse sensors.

Multiple clients, including local and remote users, can concurrently receive two groups of sensational data from two sets of registered hardware via our developed web app in real time.

Sensor principle and applications

Temperature and relative humidity (RH) measurements are often collected as part of an indoor environmental quality investigation. Most people feel comfortable when the air temperature is between 68°F (20°C) and 80°F (27°C) and the relative humidity ranges from 35% to 60%. The ASHRAE guidelines recommend 68°F to 74°F in the winter and 72°F to 80°F in the summer. The guideline also recommends relative humidity of 30 to 60 percent.[12] Poor environmental conditions will have side effect to people health. Cold indoor temperatures have been associated with increased blood pressure, asthma, and depression of elders. A very hot environment often leads to elderly heatstroke and muscle cramps. Relative humidity can affect the incidence of respiratory infections and allergies. If it is too dry, the relative humidity is below 30%. The likelihood of the spread of cold and flu increases. On the contrary, the likelihood of fungi and bacteria grows quickly if the relative humidity is higher than 60%. The heart rate is a vital body sign. It is important to identify if the elders have normal range of the heart beats. The heart rate may be affected due physical exercise, sleep, anxiety, stress, and illness. An average heart rate is from 60 to 100 beats per minute (BPM). Arrhythmia can cause heat to beat slower than 60 BPM (Bradycardia) or faster than 100 BPM (Tachycardia).

The DHT22/AM2303 and pulse sensors (Figure 1) are used in our experiment setup. The DHT22/AM2302 is a temperature and humidity sensor[13] used to monitor the living environment. The sensor is made of two parts, a capacitive humidity sensor and a thermistor. It is capable of measuring 0–100% relative humidity with 2-5% accuracy and -40 to 80°C temperature readings ±0.5°C accuracy. It is simple to use, but requires two seconds sensing period to grab the data. The sensor's output signal can use a customized digital format which can be connected to the digital input pin of Arduino Nano. The pulse sensor[14] can obtain heart beating data from the users by placing the sensor on their fingertips. When a heart beats, the blood flow makes absorption of the wavelength of green light. The reflected green light will be sensed by the light photo sensor APDS-9008. When blood pumps through tissues, the converted voltage output will change due to green light absorption. The converted voltage signal then is connected to an analog input channel of Arduino platform.

The IoT enabled cross platform design

Our current research makes use of early IoT prosthetic framework. In our improved IoT enabled cross-connected platform for smart living environment shown in Figure 2, multiple smart objects are connected to Firebase. Arduino Nano is a low cost microcontroller with a crystal oscillator of frequency 16 MHz and is not running an operating system. There are a number of libraries available in the Arduino community to enable the developer easily to access various sensors and modules that are reacting to external digital and analog inputs. With a Nano board development platform, the overall size of the prototype is greatly reduced. The Arduino Nano is approximately 85% smaller than the size of the Arduino Mega. Raspberry Pi 3 Model B+ (RPi3B+) could reach 1.4 GHz and Raspberry Pi 4 (RPi4) can reach 1.5 GHz on the CPU with low cost. They are faster than Arduino by almost 90 times in clock speed. Both RPi3B+ and RPi4 are 64 bits single-board computers built-in with Wi-Fi and Bluetooth features. They can serve miniature Linux servers to handle things like user interface, application, data processing, network connections, web servers, etc. Our approach is to directly connect the Arduino Nano board to the Raspberry Pi via serial communication USB cable. Since the low cost Arduino Nano easily reacts almost immediately to a raw-data acquisition at its digital or analog channels, we connect sensors to Arduino Nano and then signal the RPi about the sensing data changes. The acquired data will be processed and relayed by the miniature computer RPi to Firebase via Wi-Fi network connection. Our system is constructed around Firebase from Google, which provide application development platform as Backend-as-a-Service (Baas). Firebase can provide developers with a variety of tools and services, such as real time database to store data and website hosting tools to develop quality applications. We can collect and store data in Firebase real-time database as JSON (JavaScript Object Notation) objects. The data stored as JSON is synchronized to all connected clients in real-time. Our developed frontend hosting applications will retrieve stored sensational data from the real-time database to allow relevant parties (doctors, family members, etc.) to monitor the living conditions in real time from a mobile device or a website.

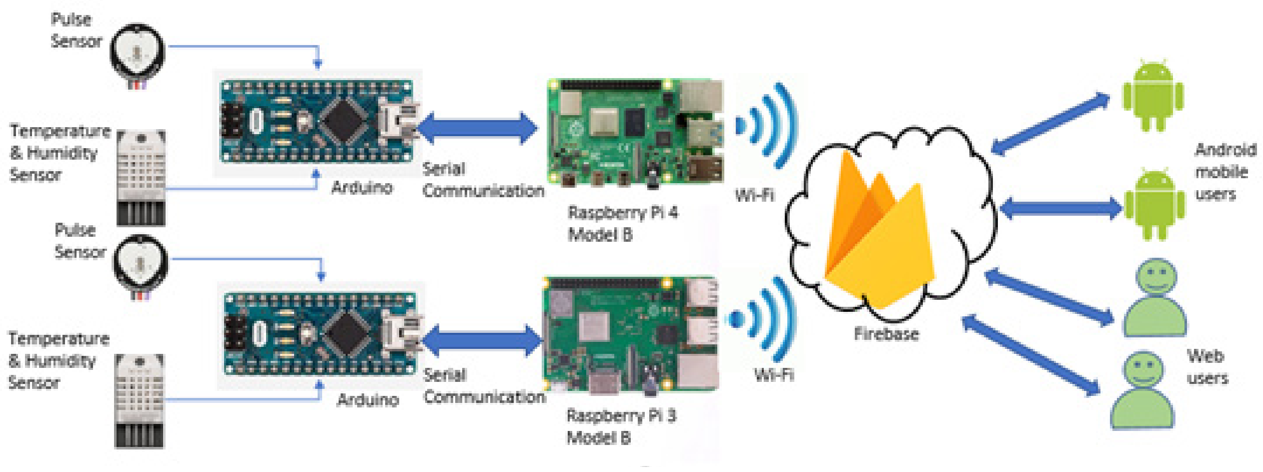


Figure 2 IoT enabled cross-connected platform for smart living environment.

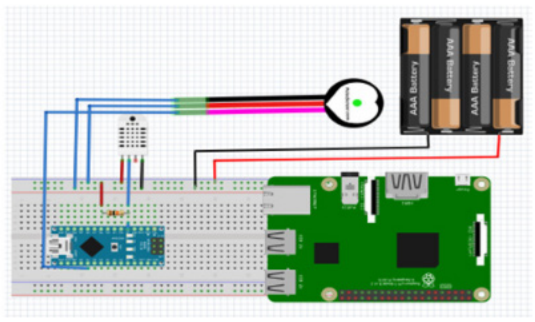
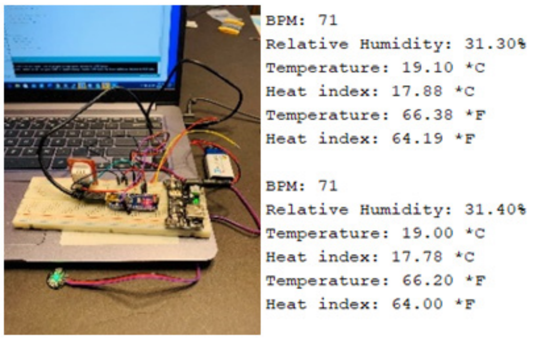


Figure 3 Schematic of data acquisition system.



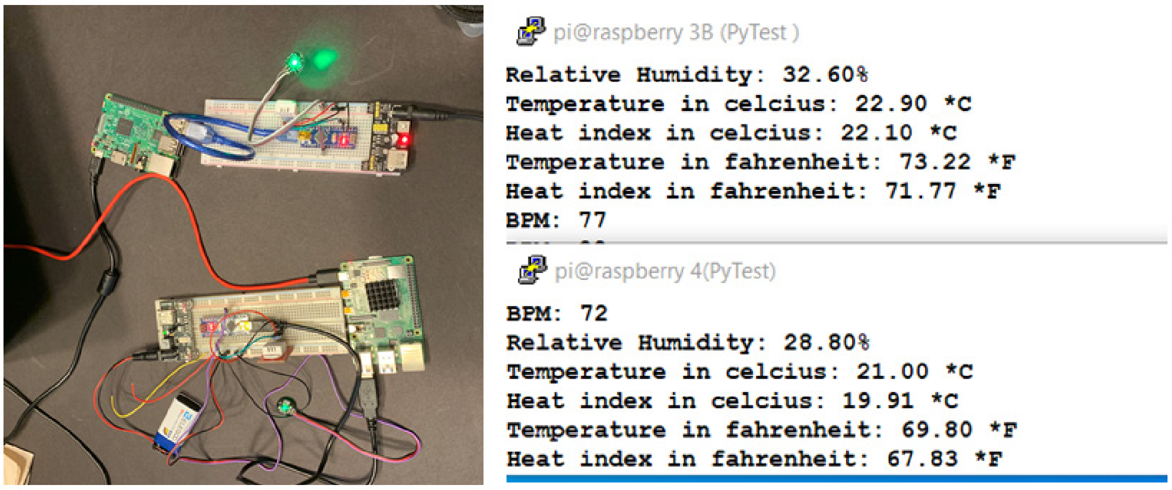
```
BPM: 71
Relative Humidity: 31.30%
Temperature: 19.10 *C
Heat index: 17.88 *C
Temperature: 66.38 *F
Heat index: 64.19 *F

BPM: 71
Relative Humidity: 31.40%
Temperature: 19.00 *C
Heat index: 17.78 *C
Temperature: 66.20 *F
Heat index: 64.00 *F
```

Figure 4 Raw data acquisition test using Arduino Nano.

Implementation, test, and analysis

Raspberry Pi does not have a standalone module for the converter from analog signal to digital signal. It only can accept the digital signal at its GPIO pins. Therefore, we choose the Arduino Nano for sensor digital or analog data acquisition. The sensed data is transmitted to Raspberry Pi from Arduino through the serial communication. The sensor DHT22/AM2303 is used to acquire the sensational data of temperature in Celsius (°C) and Fahrenheit (°F), and relative humidity (RH). The heat index is computed based on relative humidity and temperature. It referred to as the temperature felt by the body. It can be shown as heat index in °C and °F. For example, if temperature is 19°C/66.2°F and RH is 31.4%, the heat index is 18°C/64°F. The body feels colder than 19°C/66.2°F. With the temperature of 28°C/82.4°F and the RH of 40%, the heat index goes 28°C/82°F. However, if the RH goes 80% with the same temperature of 28°C/82.4°F, the heat index will increase to 32°C/90°F.[15] The body feels very hot and uncomfortable. It is a warning sign for elders to stay in safe to avoid heat exhaustion even the temperature itself is only 28°C/82.4°F.

In our experiment, the data pin of DHT22 is connected to Arduino Nano digital pin 2. The 5V DC power battery or adjustable breadboard DC 5V power supply is provided to VCC pin of DHT22. Place a 10 KΩ resistor between VCC and the data pin to act as a medium-strength pull up on the data line. *The* "DHT.h" for DHT22 is download to be included in Arduino library. The methods of the DHT object can get the values of humidity, temperature in Celsius and Fahrenheit , and heat index in Celsius and Fahrenheit by readHumidity(), readTemperature(), computeHeatIndex(), respectively. To use a pulse sensor to measure the heart beats per minute (BPM), the data line of pulse sensor is connected to the analog channel A_3. The VCC line is connected to DC 5V power supply. We follow the same procedure used for the installation library to use the DHT22 sensor. The "PulseSensorPlayground.h" is downloaded to the Arduino library directory. After a declaration of the instance of PulseSensorPlayground, the methods of getBeatsPerMinute() and sawStartOfBeat() can be accessed to acquire the data of the latest beats-per-minute (BPM) and the detected status of heartbeat pulse, respectively. All acquired data with associated strings then can be sent to Arduino Nano serial port at a baud rate of 9600. The USB cable is used to transmit the acquired sensational data from Arduino Nano to RPi. The schematic of data acquisition design is shown in Figure 3. The raw data acquisition test using Arduino Nano is shown in Figure 4, where the two groups of data in BPM, Relative Humidity, Temperature, and Heat index are obtained in the interval at least 2 seconds to satisfy DHT22 sensing time requirements. The differences of the two group data in relative humidity, temperature in Celsius, heat index in Celsius are $31.40\% - 31.30\% = 0.10\%$, $19.00°C - 19.10°C = -0.10°C$, and $17.78°C - 17.80°C = -0.10°C$ .They fall in the accuracy range 2-5% of relative humidity and ±0.5°C accuracy in temperature readings.



```
pi@raspberry 3B (PyTest )

Relative Humidity: 32.60%
Temperature in celcius: 22.90 *C
Heat index in celcius: 22.10 *C
Temperature in fahrenheit: 73.22 *F
Heat index in fahrenheit: 71.77 *F
BPM: 77
```

```
pi@raspberry 4(PyTest)

BPM: 72
Relative Humidity: 28.80%
Temperature in celcius: 21.00 *C
Heat index in celcius: 19.91 *C
Temperature in fahrenheit: 69.80 *F
Heat index in fahrenheit: 67.83 *F
```

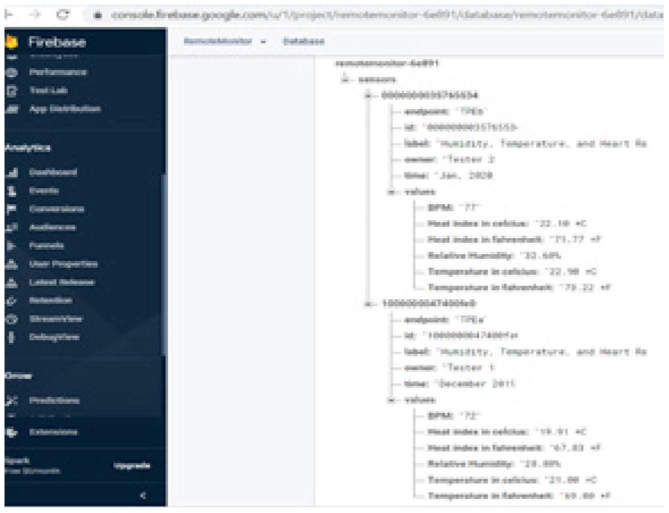
Figure 5 Experiment setup with two sets of hardware and key-value pairs.

Both RPi3B+ and RPi4 are single-board computers built-in with Wi-Fi and Bluetooth features. We connect an Arduino Nano to the USB port of RPi. The Python serial module encapsulates the access for the serial port. The imported serial module provides backbends of RPi for Python running through the serial communication. The serial communication with the baud rate 9600 can be established between Arduino programming function Serial.begin(9600) and Python programming method serial.Serial('/dev/ttyUSB0', 9600). RPi receives the sensational data by calling Python code serial.readline() and stores the data in key/value pairs in the Dictionary data structure. Figure 5 shows two sets of Arduino Nano and RPi minicomputers with the sensors are setup to run the system experiment. One set uses RPi4 with Arduino Nao and sensors of DHT22 and pulse rate, another set to use RPi3B model instead of RPi4. Both RPi3B+ and RPi4 computers are running individually at different locations to relay the data in key/value pairs to Firebase real-time database via the Wi-Fi mobile hotspot. The two groups of pairs of key and value (key, value) are stored at the Dictionary. The first group of pairs of key/value $\text{RPi3B+}_{(\text{key,value})}$ is from the RPi3B+, where $\text{RPi3B+}_{(\text{key,value})}$ = {(Relative Humility, 32.60%), (Temperature in Celsius, 22.90*C), (Heat index in Celsius, 22.10*C), (Temperature in Fahrenheit, 73.22 *F), (Heat index in Fahrenheit, 71.77*F), (BPM, 77)}. The second group of pairs of key/value $\text{RPi4}_{(\text{key, value})}$ is from RPi4, where $\text{RPi4}_{(\text{key, value})}$ = {(Relative Humility, 28.80%), (Temperature in Celsius, 21.00*C), (Heat index in Celsius, 19.91*C), (Temperature in Fahrenheit, 69.80 *F), (Heat index in Fahrenheit, 67.83*F), (BPM, 77)}. All of key/value pairs $\text{RPi3B+}_{(\text{key, value})}$ and $\text{RPi4}_{(\text{key, value})}$ are shown in real-time. The key/value pairs will be uploaded to a Firebase database via Wi-Fi and then distributed to the chosen endpoints.

Access to firebase

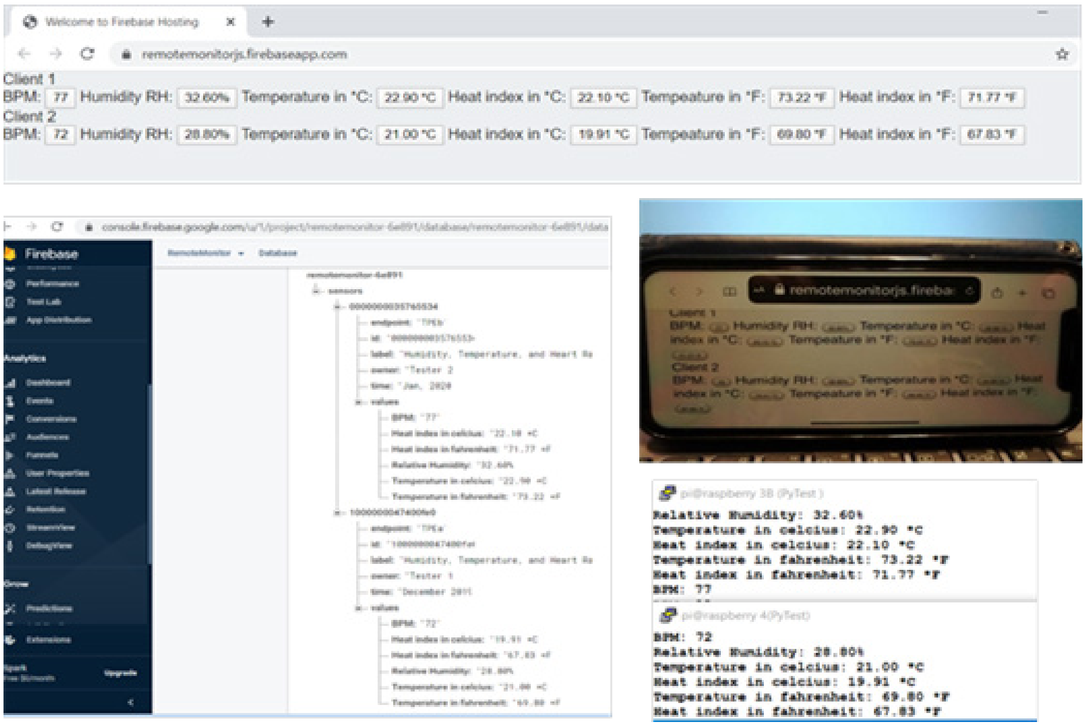
Firebase is built on Google infrastructure and is a cloud-based services. Backend as a Service (BaaS) is currently used in Firebase. *BaaS* provides the *backend* for mobile and web applications to integrate with their application *backend*. A server-side rule was created when register to Firebase project. The rule adds the user object to the JavaScript Object Notation (JSON) database. Only authenticated object monitors can read or write data to interact with the database. The sensor and rule models are generic objects. The database can store health and environmental data as JSON tree. Data stored as JSON is synchronized to all connected clients in real-time. Firebase provides tools to allow the access of the real time database by a number of programming languages, including Python and JavaScript. The Python program running on RPi is used to receive the data from Arduino platform and transmit them via Wi-Fi to Firebase. After the Firebase project is created with service account, we can install the Firebase Python SDK and import firebase_admin to read and write real-time database data with full admin privileges, or limited privileges. Each RPi have unique CPU serial number that can be obtained by imported function open('/proc/cpuinfo','r'). When fetching the CPU serial number at the location in the database, we can retrieve all of its associated child nodes. Even though a JSON data tree can support thirty-two levels deep in each path, in practice, we opt to keep the data structure as flat as possible. This allows faster data access. Figure 6 shows JSON objects from the Firebase real-time database. The CUP serial number $RPi3\_id = 0000000035765534$ is for RPi3B+ model and $RPi4_{id} = 1000000047400fe0$ is for RPi4. When fetching RPi3_id, it can retrieve all child nodes where the values are acquired by its set of sensors. $RPi3\_id$ child nodes show that owner is Test 2 in Jan 2020 and the pairs of key/value is the same as $\text{RPi3B+}_{(\text{key, value})}$. When fetching RPi4_id, the app can retrieve all child nodes where the values associated with its sensational data. $RPi3\_id$ child nodes show owner Test 1 in December 2019 and the pairs of key/value is the same as $\text{RPi4+}_{(\text{key, value})}$.

Frontend hosting applications to retrieve JSON nodes from the Firebase real time database allows relevant parties (doctors, family members, etc.) to monitor the living environment and health in real time from a mobile device or from a website. A website can be constructed using tools provided by Firebase to build a html/JavaScript file to display the stored data in a real-time database. Figure 7 shows an example of the webpage that displays sensor data updated in real time, in synchronization with NoSQL real-time database and the sensor data received by the Python program running on Raspberry Pi. In Figure 7, the real-time database with JSON nodes is shown at the lower left side of the figure. The webpage, with URL in firebaseapp.com domain shown at the top as well as the middle right side, is from a web browser on a PC laptop and cell phone, respectively. The lower right side of t shows Raspberry Pi 3 Model B+ and Raspberry Pi 4 running a Firebase web server. The pairs of key/value $\text{RPi3B+}_{(\text{key, value})}$ and $\text{RPi4}_{(\text{key, value})}$ are the same across the Firebase real-time database, webpage shown on Laptop and phone, and the python program that receives the sensor data from Arduino. They all displays the five data values in two groups: the heart beat per minute (BPM), the relative humidity (Humidity RH), the temperature in Celsius (Temperature in *C), the heat index in Celsius (Heat index in *C), the temperature in Fahrenheit (Temperature in *F), and the heat index in Fahrenheit (Heat index in *F). Therefore, when the running Python program receives the sensor data from Arduino, the changes of sensor data values in real time are captured in the browser. Data stored as JSON is synchronized to all connected clients in real-time. An authorization object monitors and manages the ability of users to interact with the database. With the communication hardware and software components and Firebase and website, the sensing data can be sent locally or remotely to provide caregivers and experts useful and significant information.



```
remotemonitor-6e891
  sensors
    0000000035765534
      endpoint: "TPEb"
      id: "0000000035765534"
      label: "Humidity, Temperature, and Heart Ra
      owner: "Tester 2"
      time: "Jan, 2020"
      values
        BPM: "77"
        Heat index in celsius: "22.10 *C"
        Heat index in fahrenheit: "71.77 *F"
        Relative Humidity: "32.60%"
        Temperature in celsius: "22.90 *C"
        Temperature in fahrenheit: "73.22 *F"
    1000000047400fe0
      endpoint: "TPEa"
      id: "1000000047400fe0"
      label: "Humidity, Temperature, and Heart Ra
      owner: "Tester 1"
      time: "December 2019"
      values
        BPM: "72"
        Heat index in celsius: "19.91 *C"
        Heat index in fahrenheit: "67.83 *F"
        Relative Humidity: "28.80%"
        Temperature in celsius: "21.00 *C"
        Temperature in fahrenheit: "69.80 *F"
```

Figure 6 JSON objects from Firebase real-time database.



Client 1
BPM: 77 Humidity RH: 32.60% Temperature in °C: 22.90 °C Heat index in °C: 22.10 °C Tempeature in °F: 73.22 °F Heat index in °F: 71.77 °F
Client 2
BPM: 72 Humidity RH: 28.80% Temperature in °C: 21.00 °C Heat index in °C: 19.91 °C Tempeature in °F: 69.80 °F Heat index in °F: 67.83 °F

```
remotemonitor-6e891
  sensors
    00000000357655334
      endpoint: "TPEb"
      id: "00000000357655334"
      label: "Humidity, Temperature, and Heart Ra
      owner: "Tester 2"
      time: "Jan, 2020"
      values
        BPM: "77"
        Heat index in celcius: "22.10 *C"
        Heat index in fahrenheit: "71.77 *F"
        Relative Humidity: "32.60%"
        Temperature in celcius: "22.90 *C"
        Temperature in fahrenheit: "73.22 *F"
    10000000047400fe0
      endpoint: "TPEa"
      id: "10000000047400fe0"
      label: "Humidity, Temperature, and Heart Ra
      owner: "Tester 1"
      time: "December 2011"
      values
        BPM: "72"
        Heat index in celcius: "19.91 *C"
        Heat index in fahrenheit: "67.83 *F"
        Relative Humidity: "28.80%"
        Temperature in celcius: "21.00 *C"
        Temperature in fahrenheit: "69.80 *F"
```

Client 1
BPM: Humidity RH: Temperature in °C: Heat index in °C: Temperature in °F: Heat index in °F:
Client 2
BPM: Humidity RH: Temperature in °C: Heat index in °C: Temperature in °F: Heat index in °F:

```
Relative Humidity: 32.60%
Temperature in celcius: 22.90 °C
Heat index in celcius: 22.10 °C
Temperature in fahrenheit: 73.22 °F
Heat index in fahrenheit: 71.77 °F
BPM: 77
```

```
BPM: 72
Relative Humidity: 28.80%
Temperature in celcius: 21.00 °C
Heat index in celcius: 19.91 °C
Temperature in fahrenheit: 69.80 °F
Heat index in fahrenheit: 67.83 °F
```

Figure 7 Sync data with NoSQL cloud database across all of the clients in real time.

Conclusion and future works

The project deployed IoT enabled cross-connected platform for a smart living environment. The multiple platforms communicate each other to send sensing data via serial communication to Raspberry Pi from Arduino Nano, then have the Raspberry pi Wi-Fi relay sensing data to the Firebase real-time database, which is accessible through our developed website. The cloud-based solution has no restrictions to a geographical location. The real-time information can be monitored locally or remotely. The system has multiple platforms to satisfy user needs. However, we found out the pulse rate sensor has not been working properly such as inconsistent BPM readings. Our further work will include to calibrate the sensational information to remove the noises, develop a mobile Android app with low energy Bluetooth connection, add more user-friendly vital sign sensors into the platform, such as non-contact infrared temperature sensor to acquire body temperature, and apply the learning model to forecast and predict health conditions.

Acknowledgments

This research is supported by PSC-CUNY Research Awards 61463-0049. We would like to thank CET graduated student Mohamed Alborati.

Conflicts of interest

Authors declare that there is no conflict of interest.

References

1. World Population Prospects 2019: Highlights (ST/ESA/SER.A/423).
2. Qin Ni, Ana Belén García Hernando, Iván Pau de la Cruz. The Elderly's Independent Living in Smart Homes: A Characterization of Activities and Sensing Infrastructure Survey to Facilitate Services Development. *Sensors (Basel)*. 2015;15(5):11312–11362.
3. Patel S, Park H, Bonato P, et al. A review of wearable sensors and systems with application in rehabilitation. *J NeuroEngineering Rehabil*. 2012;9:21.
4. Laplante PA, Kassab M, Laplante NL, et al. Building Caring Healthcare Systems in the Internet of Things. *IEEE Systems Journal*. 2018;12(3):3030–3037.
5. Uddin MZ, Khaksar W, Torresen J. Ambient Sensors for Elderly Care and Independent Living: A Survey. *Sensors (Basel)*. 2018;18(7):2027.
6. Yu Wang, Sunghoon Jang. A pulse sensor interface design for FPGA based multisensor health monitoring platform. *International Journal of Biosensors & Bioelectronics*. 2019;23-27.
7. Cippitelli E, Fioranelli F, Gambi E, et al. Radar and RGB-Depth Sensors for Fall Detection: A Review. *IEEE Sensors Journal*. 2017;17(12):3585–3604.
8. Niezen G, Eslambolchilar P, Thimbleby H. Open-source hardware for medical devices. *BMJ Innov*. 2016;2(2):78–83.
9. Kamal N, Ghosal P. Three Tier Architecture for IoT Driven Health Monitoring System Using Raspberry Pi. 2018 IEEE International Symposium on Smart Electronic Systems (iSES) (Formerly iNiS). Hyderabad, India. 2018. pp. 167–170.
10. Rahman RA, Aziz NSA, Kassim M, et al. IoT-based personal health care monitoring device for diabetic patients. IEEE Symposium on Computer Applications & Industrial Electronics (ISCAIE). Langkawi. 2017. pp. 168–173.

11. Yu Wang, Hunter W, Chen X, et al. Improved Hardware Design of IoT Prosthetic Device. The 2018 ASEE Mid-Atlantic Fall Conference. Brooklyn, NY. 2018.
12. Martinez G, Chen X, Lai T, et al. IoT in Myo-Prosthetics. 2018 ASEE Northeast Section Annual Conference. Hartford, CT. 2018
13. https://learn.adafruit.com/dht
14. https://www.broadcom.com
15. https://www.wpc.ncep.noaa.gov/html/heatindex.shtml