**Monitoring Activities of Elderly People Using Wearable Sensors and Secure Communication to the Remote Location**

Swayam Prakash Pati (19BCT0161) Rachit Elhance (19BCT0187)

([swayamprakash.pati2019@vitstudent.ac.in](mailto:swayamprakash.pati2019@vitstudent.ac.in)) ([rachit.elhance2019@vitstudent.ac.in](mailto:rachit.elhance2019@vitstudent.ac.in))

Rahul Chetan Thakar (19BCT0192) Disha Jain (19BCT0219)

([rahul.thakar2019@vitstudent.ac.in](mailto:rahul.thakar2019@vitstudent.ac.in)) ([disha.jain2019@vitstudent.ac.in](mailto:disha.jain2019@vitstudent.ac.in))

VIT - Vellore Institute of Technology

**I. Proposed Method**

We aim to build a framework which utilises Wearable Sensors and a smart-phone app for Smart Healthcare Monitoring System for elderly people.

The main objective of our model will be to

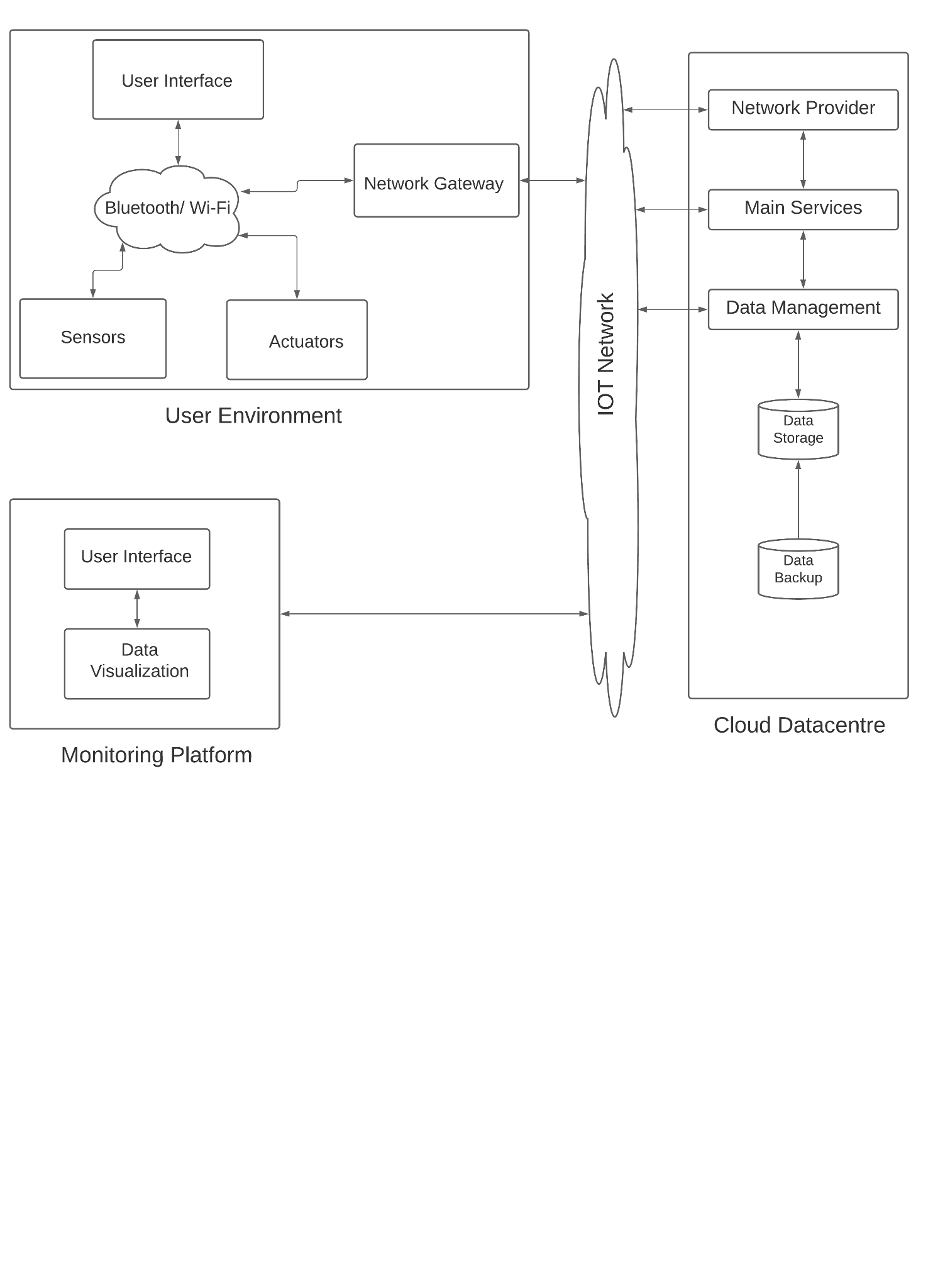
* + monitor the patient’s physiological data received from the sensor network,
  + upon the retrieval of data, the record is created and data is accordingly stored
  + provide the data and detected diseases to the healthcare monitoring facilities in real time
  + and provide alerts in cases of symptoms of abnormality

Our model will include the following 3 layers :

**Wearable devices (Patients Layer):** A network of wearable sensors are going to be connected to the patient's body to gather physiological data. Such sensors are measuring vital-sign, like blood oxygen saturation, skin temperature and heart-rate; a spread of healthcare sensors are available today. The data gathered is transmitted to the patient's PDA via Bluetooth device and ultimately to a cloud database through a network gateway. Moreover, sensors will operate to live and send the info regularly without patient attention to form everything automated, thus, this enhances users quality of experience and makes it easier .

**Cloud (Data Layer):** The Cloud receives patient’s data from the gateway over the web to be sorted then it becomes available for doctor’s inspections. Additionally , all data analysis and processing are going to be held within the cloud for any disorder detection in patient’s data, thus, the abnormal changes in patient’s data are going to be categories supporting patient status and diseases. All resulting data/info are going to be reported either to patient’s and/or doctor’s platform or emergency unit or both counting on patient status. Thus, Cloud enables collaboration and knowledge sharing through its infrastructure which allows medical professionals to host information, analytics and diagnostics about patients in order that other professionals around the similar interests can immediately access the info . This reveals faster prescriptions and real-time updates to patient’s data.

**Monitoring platform (Hospital Layer):** This layer may be a doctor’s platform to watch patient’s records and sensory data. The doctors are going to be ready to inspect reports provided by the system from the cloud and that they are ready to take actions. Data synchronisation during this platform in real-time by pulling all data from the cloud as soon as it becomes able to use to stay doctors up so far with patient’s status, also to assist paramedics to require early action just in case of emergency before things gets worse and stop hospitalisation.

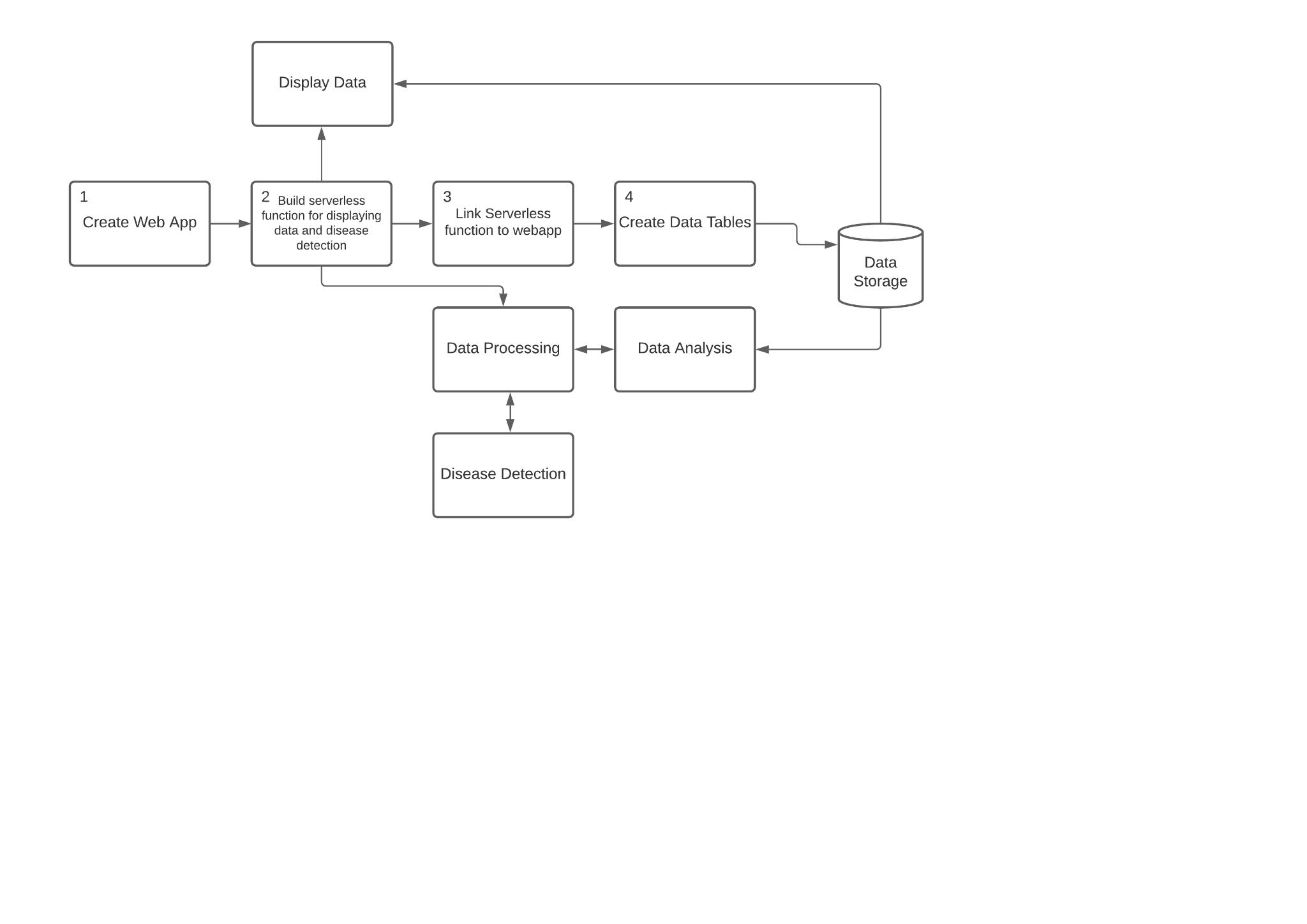


**The Wireless Sensor Network will have a star topology.**

**The main components of the model are:**

* **User Environment (wearable sensors):**
  + where all medical sensors and wearable devices are connected to patients and operated over the IoT network.
  + Each sensor/wearable (e.g., pulse sensor) has a unique identifier and communication capability over the network to interact with the gateway.
  + Data can be **bio-medical or environmental.**
  + The generated/gathered data in this layer are transmitted to the gateway via wireless or wired communication protocols, such as Bluetooth, ZigBee, and Wi-Fi.
* **Gateway:**
  + This comes under the patient's layer.
  + Perform prime processing on the sensed data.
  + Outcome of this segment is a summary of patients' conditions sent to the caregivers.
  + Gateway is also able to **react upon symptoms of abnormality** when it’s been detected, for instance, sending a help-request (e.g., request for caregiver for assistant) or **emergency-request** (e.g., call for an ambulance) as soon as an emergency situation is detected.
  + **In this research, a programmed Arduino Uno and a patient's smart-phone will be used as a gateway to prove the concept of the proposed solution.**
* **Cloud Data-center:** 
  + This segment was deployed to act on data provided from the Gateway.
  + Thus, the main data storage, data display and disease detection activity are executed.
  + This includes the machine learning (ML) activities for data training and analysis to detect diseases and abnormality.
* **Monitoring Platforms:**
  + This includes both the patient's platform and caregivers' platform.
  + Each dashboard provided for data visualisation, monitoring, and functionality controls over the collected data from sensors.
  + This platform requires the level of security access alongside with different access layers, for example, patients can have the master level of access, while caregivers have restricted access according to the granted level of access.

We will be deploying our application using AWS cloud and thus the following procedure will be implemented.



# **Sensors Table:**

| **CardioVascular System** | **Function** | **Sensors** |
| --- | --- | --- |
| **Wrist Band** | To measure SpO2, Heart rate and Body Temperature. | * Pulse Oximeter Sensor * Temperature Sensor |
| **ECG Sensor** | Constantly monitoring cardiac activity. | * Dry Sensor |
| **Patch Based Wearable Sensor** | For sweat analysis. | * Sweat Sensor * Electrochemical Sensors |

| **Motion Monitoring System** | **Function** | **Sensors** |
| --- | --- | --- |
| Fall Detection System [24] | To monitor and alert sudden falls. | * 1 3-axis accelerometer * 1 3-axis gyroscope * 1 3-axis magnetometer |

| **Sleep Monitoring System** | **Function** | **Sensors** |
| --- | --- | --- |
| Sleep Quality Monitoring [15] | To monitor sleep patterns. | * Magnetometer |

| **Environment Monitoring System** | **Function** | **Sensors** |
| --- | --- | --- |
| Wearable Thermo-Hygrometer | To measure temperature and humidity around, to predict heat strokes. | * Thermo-Hygrometer |
| Pollutant Detection System | Sensors for toxic gases, monitoring pollutants such as heavy metals, and allergens such as pollen. | * Toxic Gas Sensor * Heavy Metal Sensor * Pollen Sensor |

| **Respiration Monitoring System** | **Function** | **Sensors** |
| --- | --- | --- |
| Respiration Monitor [14] | To monitor breathing patterns. | * Ionic electroactive polymer-based soft sensor |

**II. Implementation**

1. **Network of Sensors**

We have used the following sensors in our model -

Temperature Sensor - For Temperature Monitoring

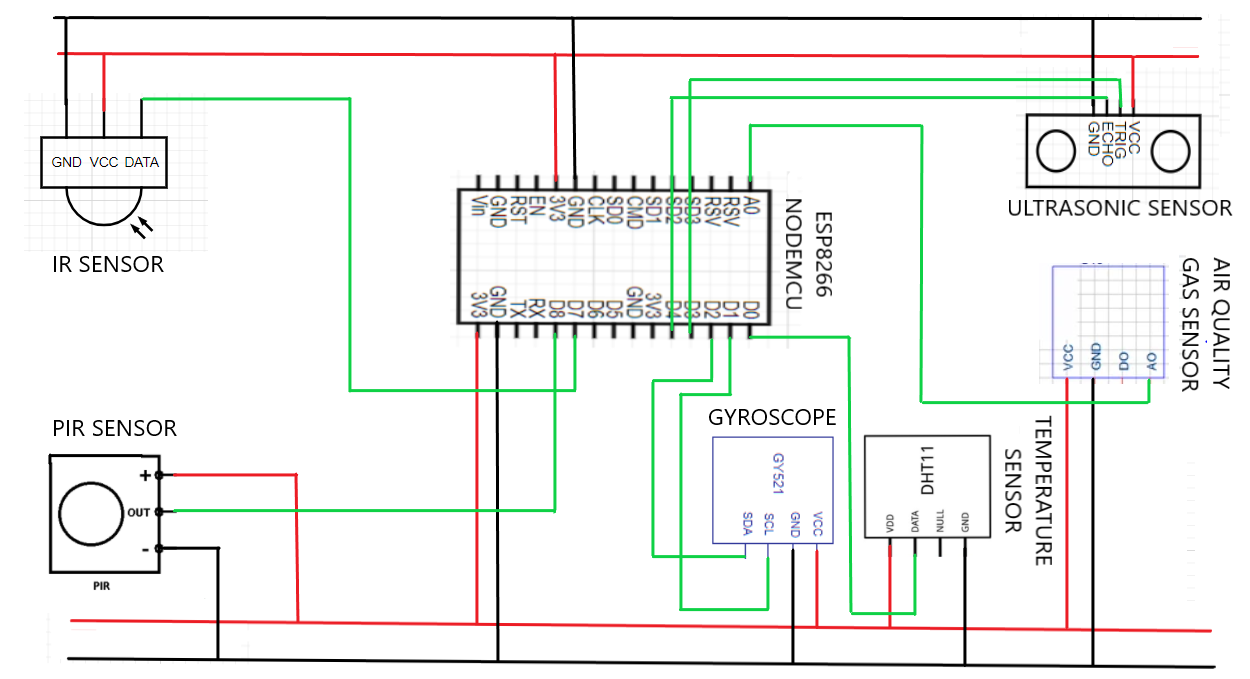
Accelerometer Sensor, Gyroscope, UltraSonic Sensor - For Fall Detection

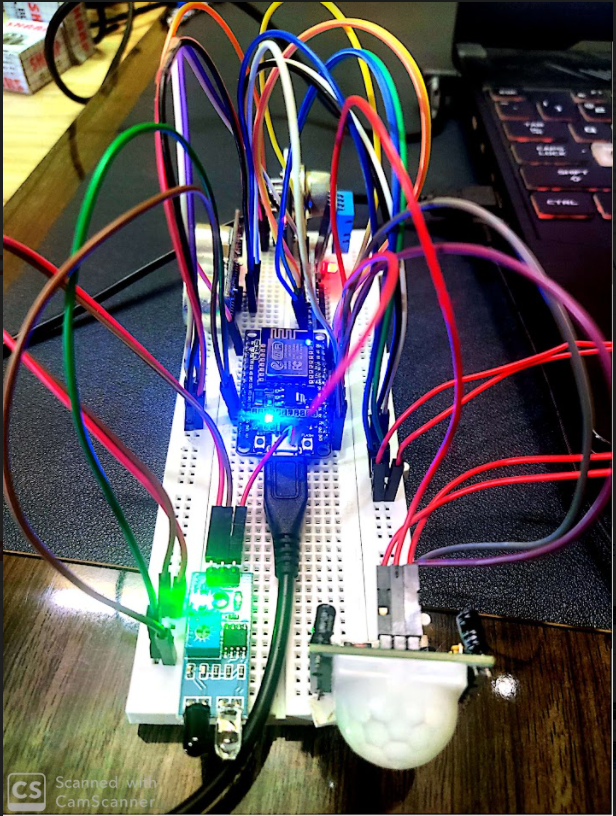
Air Quality Gas Sensor - For Measuring the Toxicity in the Air

IR sensor - For Object Detection

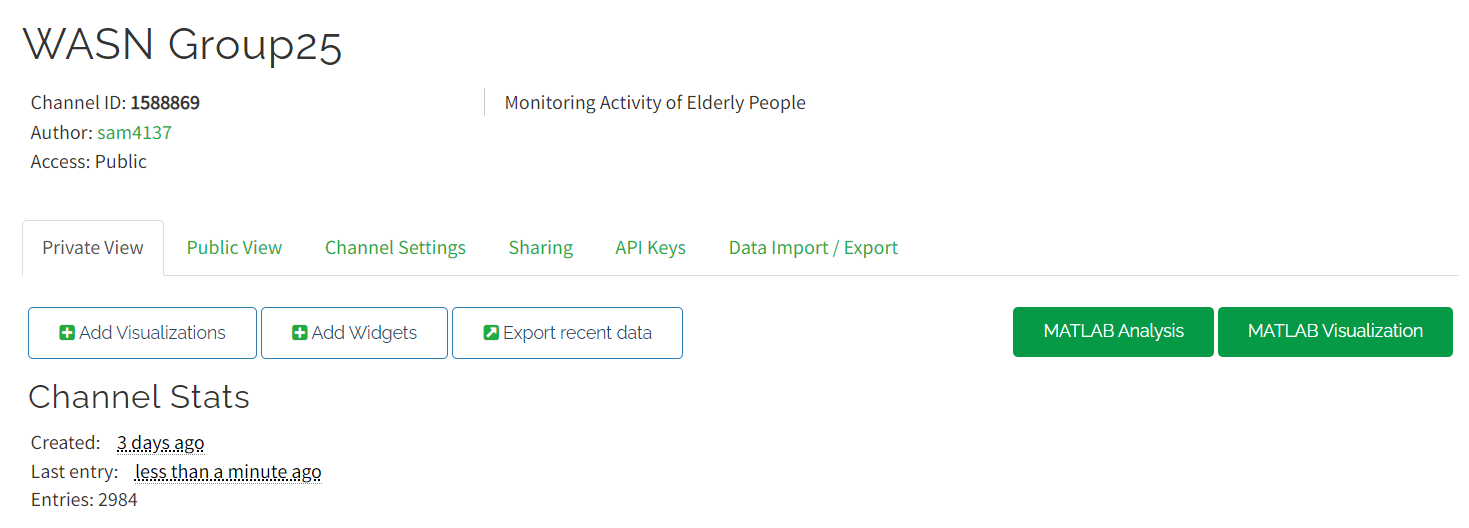
PIR sensor - For Motion Detection

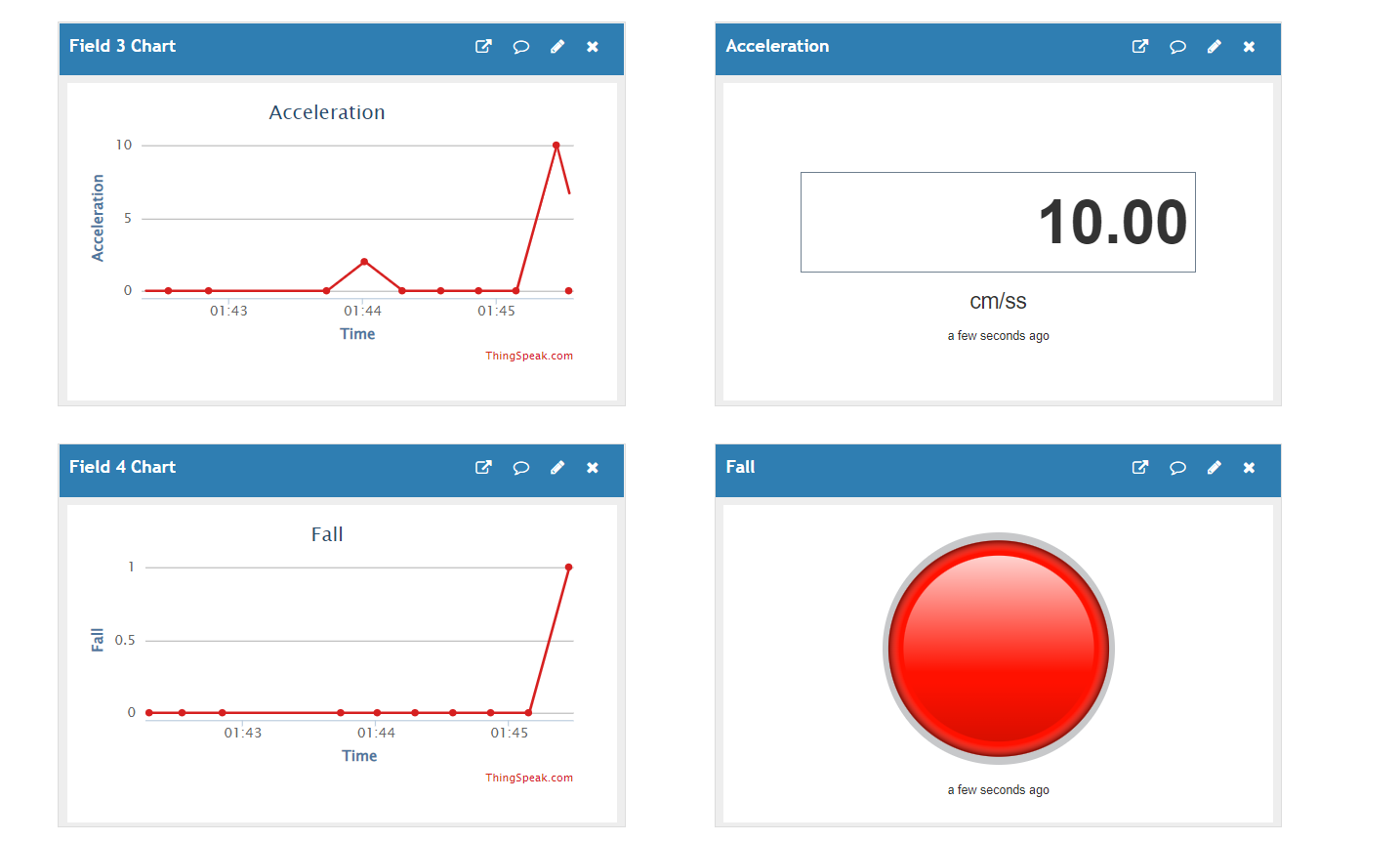
Here is the circuit diagram of the sensor network -





1. **Visualization using ThingSpeak**
   1. Creating a new Channel; giving appropriate name and adding the required fields
   2. After creating a channel, it is hosted as a private channel displaying the graphs for all the fields w.r.t. Date and time.Add appropriate widgets.
   3. Edit the titles and axes w.r.t. fields displayed in the graphs.
   4. In API keys we can see the read and write API keys of the channel. Add the write API key to the .ino file.
   5. Make the channel public in the channel sharing settings
   6. Next, we need the channel to create a new rule to trigger email whenever the data goes above or below the threshold. Under matlab analysis, select read data to trigger email and then create analysis.
   7. In the next page, edit the relevant fields in the code then save and run the code to check for errors. The email will be triggered accordingly whenever the threshold value(s) are reached.





1. **Initialization of DynamoDB table with a connection with MQTT client**

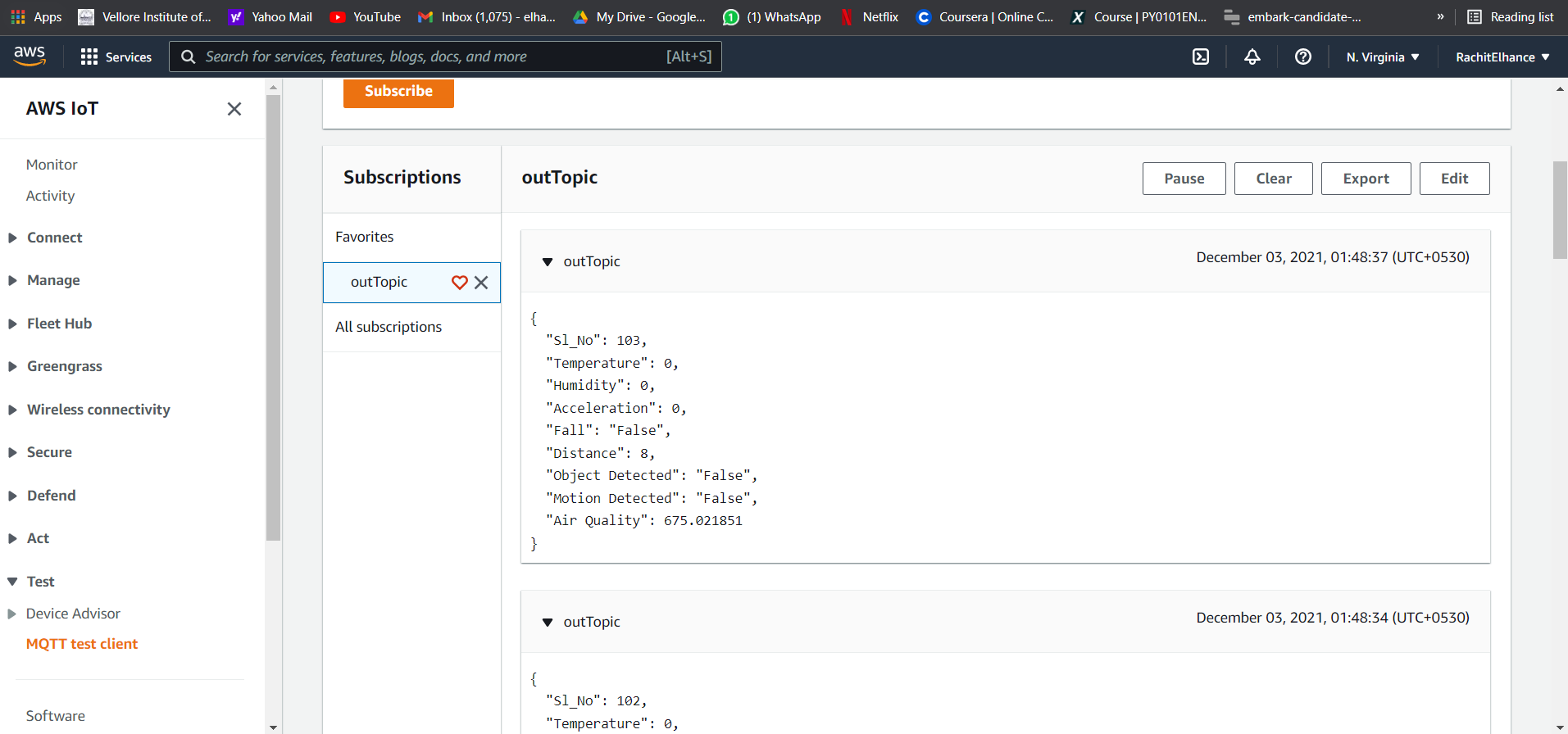
The AWS IOT CORE platform is used to create a thing named ESP8266 with the required details. The certificates and keys were downloaded accordingly.

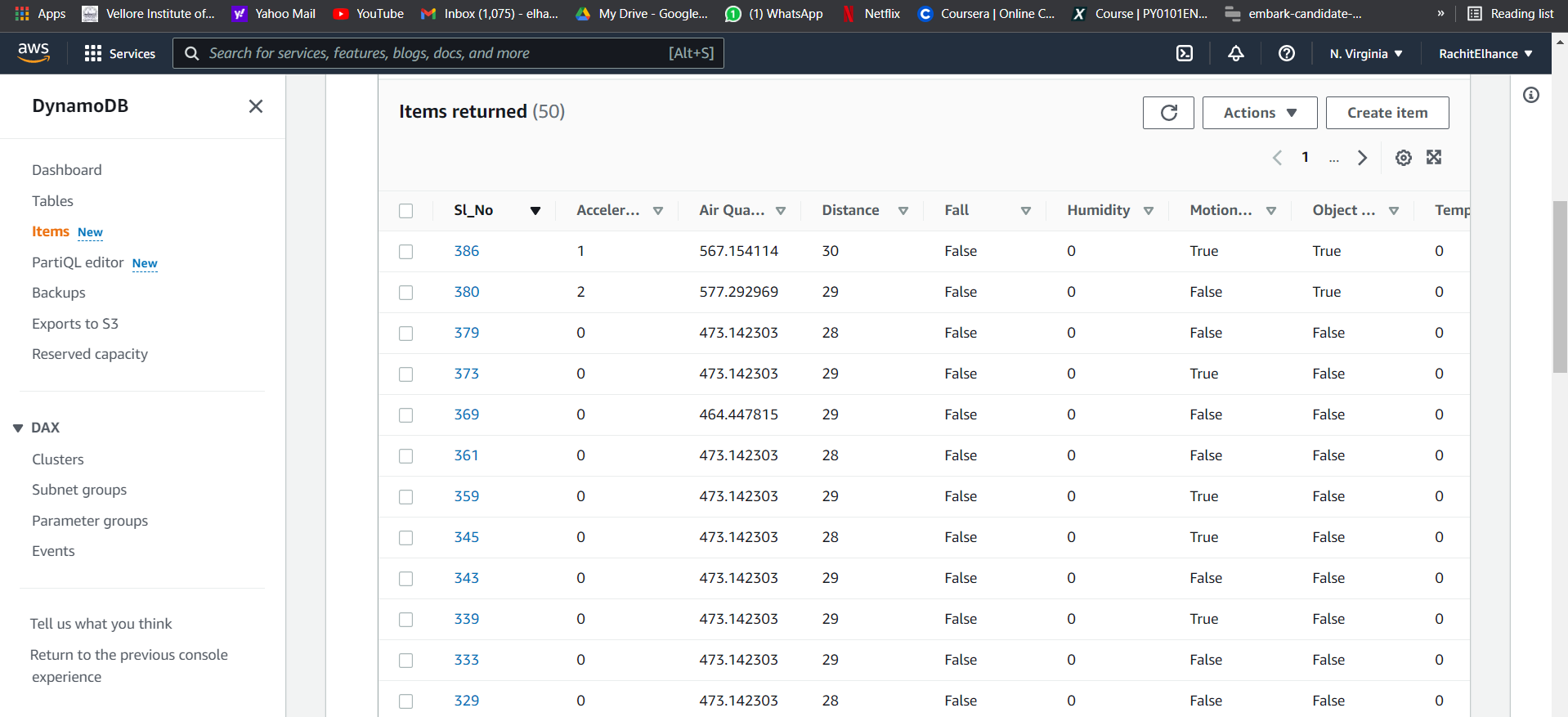
We then created a policy named ESP8266 and connected it with the rightful certificate.

Next, the MQTT connection is set up named outTopic.

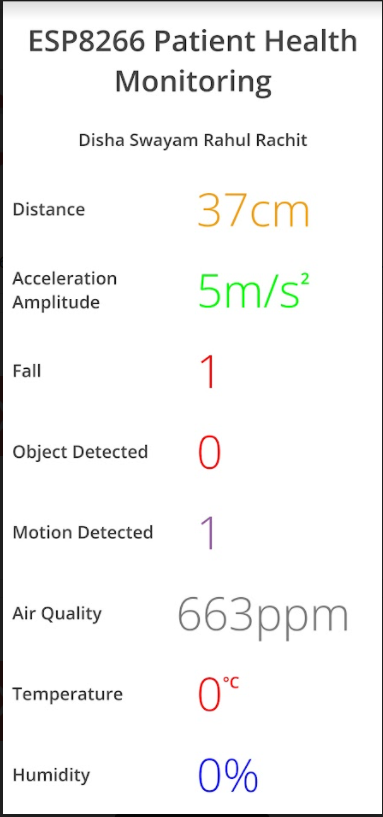
A rule is created on the IOT platform named ESP8266, and the required query statements are accordingly.   
Next a new table is created in DynamoDB named WASN Group 25 for which the partition key for Sl. No. with auto scaling options.

We then connected the DynamoDB table to the rule and defined the functions in the code accordingly.



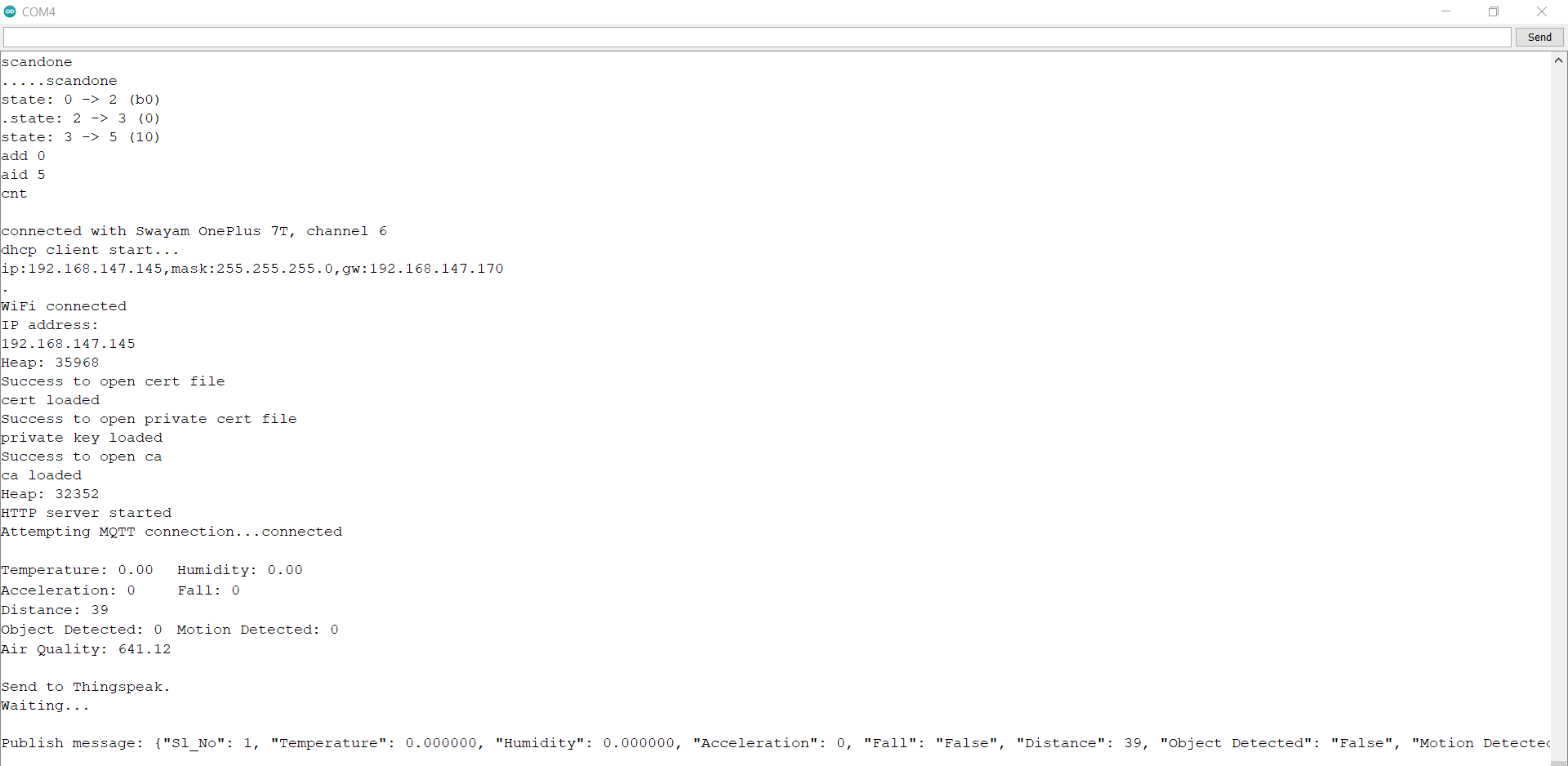


1. **Algorithm used for the code** 
   1. Including all the relevant libraries for the sensors used
      1. Adding special libraries that are included for reading and writing data we get from the temperature sensor.
      2. Adding special libraries that are included for reading and writing data we get from the Gyroscope.
      3. Adding special libraries that are included for reading and writing data we get from the Air Quality sensor.
      4. Adding the rest of the libraries that are required for server upload functions, MQTT transmissions, HTML local server transmissions, NodeMCU operations, wifi connections and for performing simple read, write activities on serial monitor and AWS, Thingspeak clients.
   2. Initializing variables
      1. To send data to the Thingspeak server we create an ‘api-key’ variable and a server address variable to connect with thingspeak using its write api key.
      2. To upload data to Thingspeak and the AWS server we create variables declaring the wifi and pass. The nodemcu will connect to the wifi using the details.
      3. Initializing all the variables required for data processing and assigning variables for the GPIO pins used.
      4. Initializing broker id and wificlient to set and send MQTT to AWS server.
   3. Creating a setup\_wifi() to connect to the wifi. Create the following reconnect function to loop until we are connected to the MQTT server client.
   4. Inside the void setup() we set up the serial monitor to print our data locally.
      1. Setting pins for INPUT and OUTPUT modes according to the sensors and their corresponding GPIO pin variables.
      2. Creating a Heap for private,public keys and certificates for the AWS Thing.
      3. Begin the HTTP local server.
   5. Inside the loop() we post the data onto the Thingspeak, AWS and HTTP servers for the outputs.
      1. Get all the data from the connected MPU and perform relevant operations to get the desirable output values from the Temperature, IR(obstruction), PIR(motion), gyroscope(acceleration and axis tilt), and Ultrasonic(proximity) sensors.
      2. Segregating all the data as different field variables and sending the data to the Thingspeak server via MQTT.
      3. Uploading all the data onto the AWS server in th JSON format, to be displayed in DynamoDB, using MQTT.
      4. Locally displaying all the information on the HTTP server.
   6. Closing any open connections when the transmission process is manually terminated.
2. **Implementation of User Interface**
   1. We are rendering the HTML code for the user interface from the source code itself by using the library ESP8266WebServer.h.
   2. We use the function SendHTML to append the HTML code with the parameters and send it to the server for the user to see.
   3. An empty string is defined in the function, where the entire HTML code is appended with the CSS properties and SVGs and the received physiological data.
   4. This entire string is then returned and thus the web page for the end user is shown on the server.

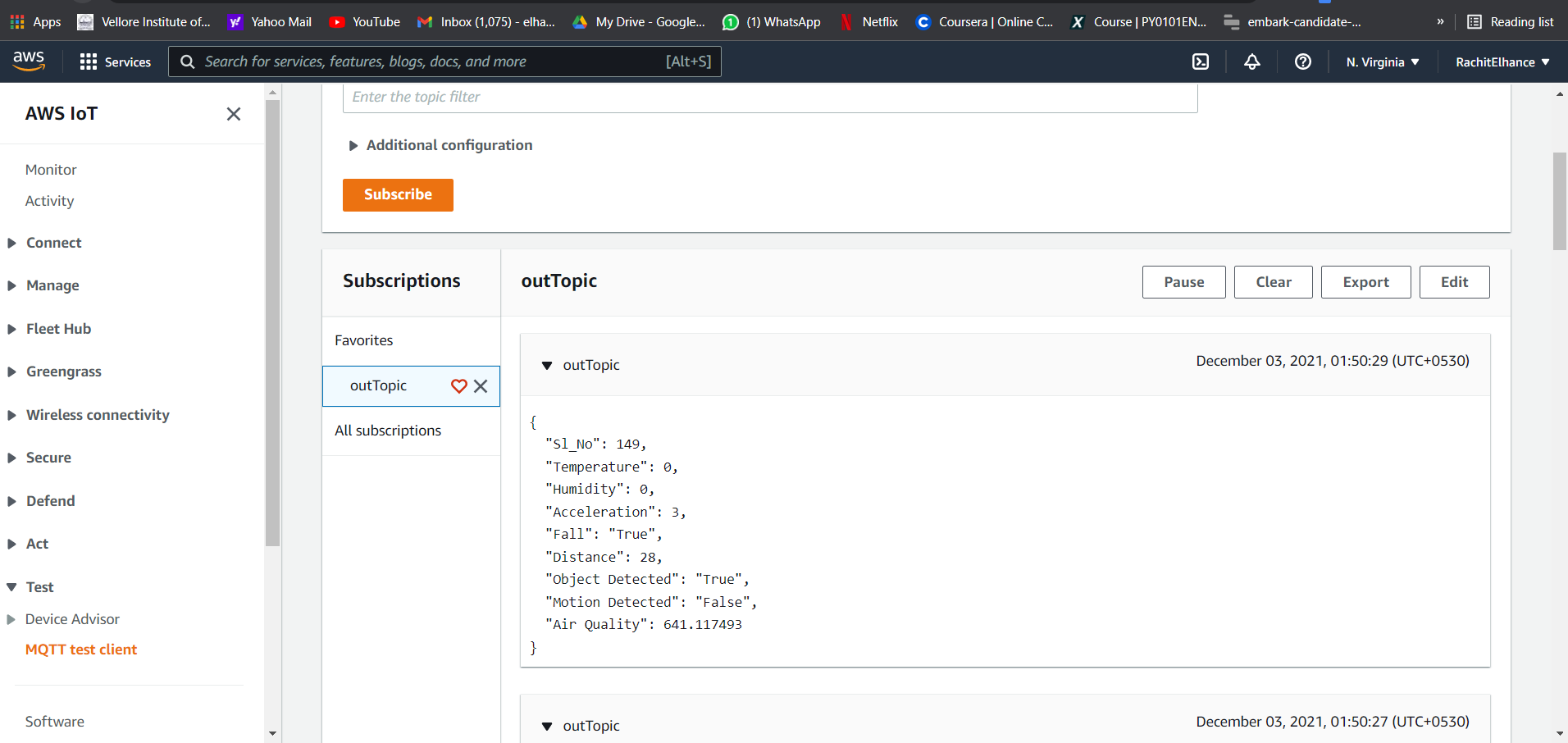


**III. Results**

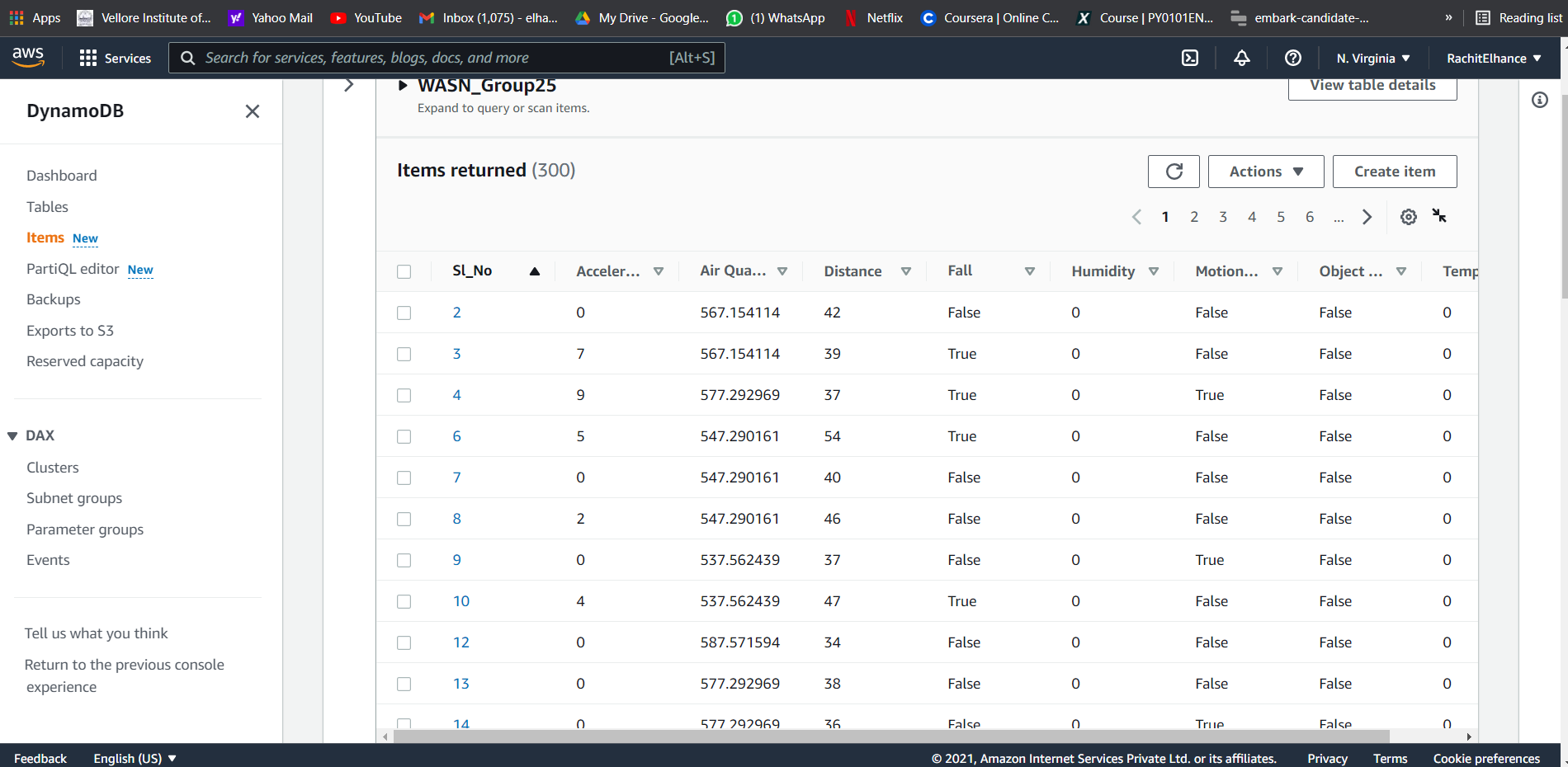
The Serial Monitor prints the data according to the specified formats.

****

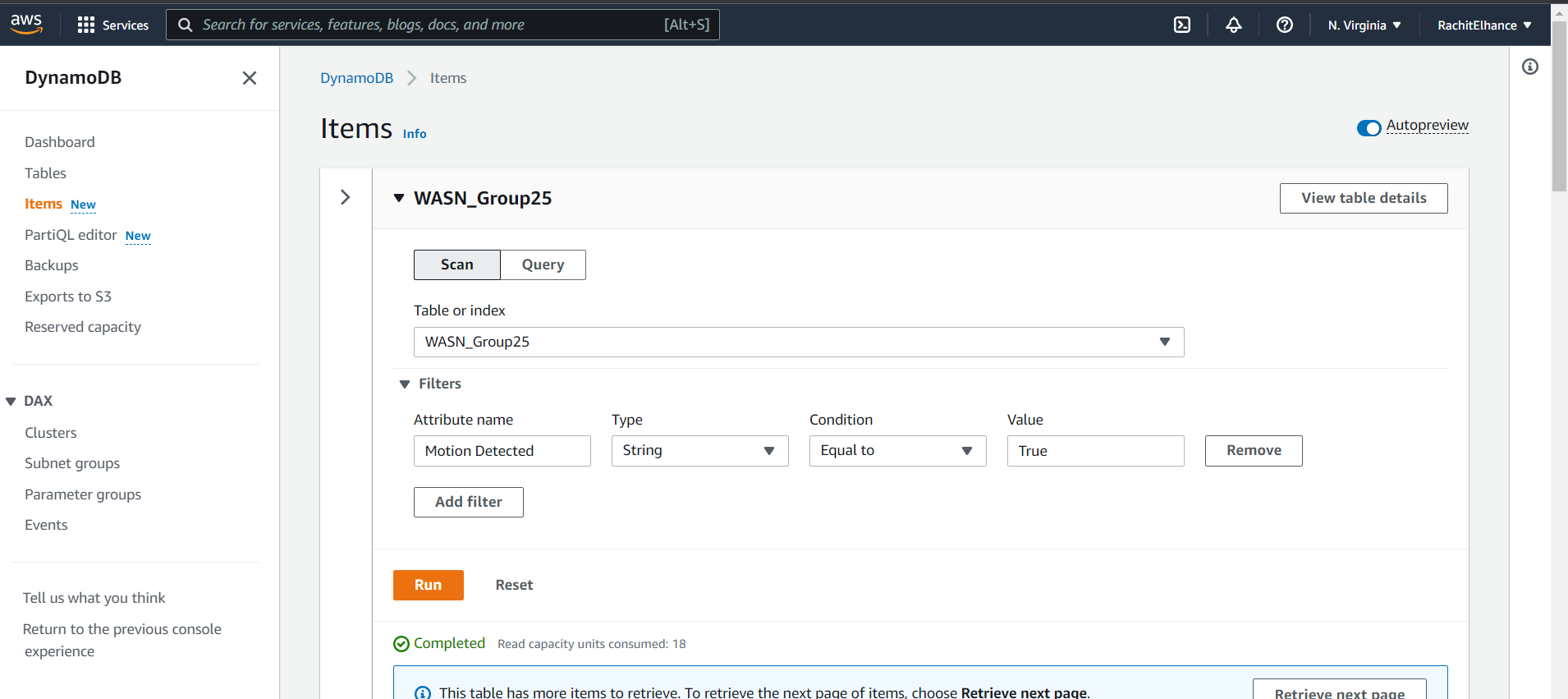
The object sent by the microcontroller is received at AWS IoT through the MQTT client.

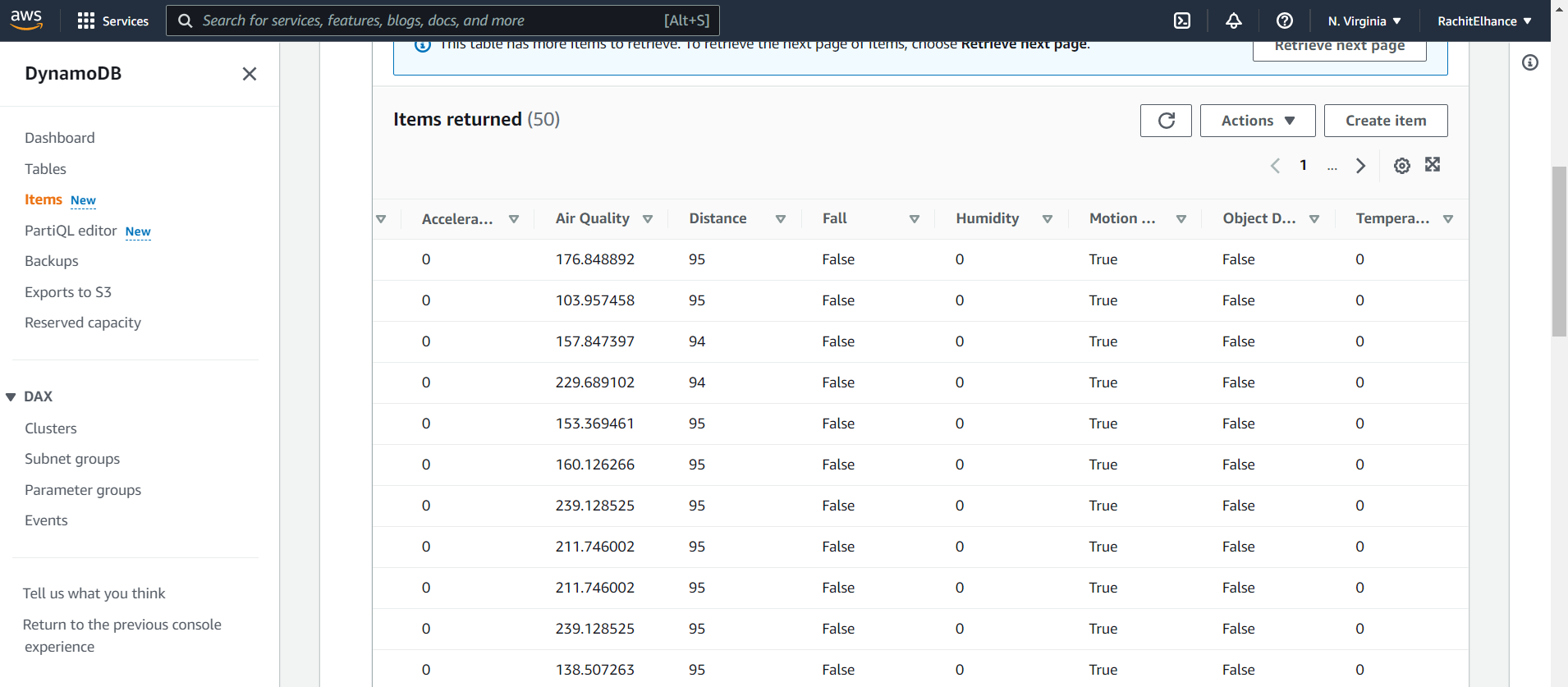


The object is sent to DynamoDB and stored in the cloud.

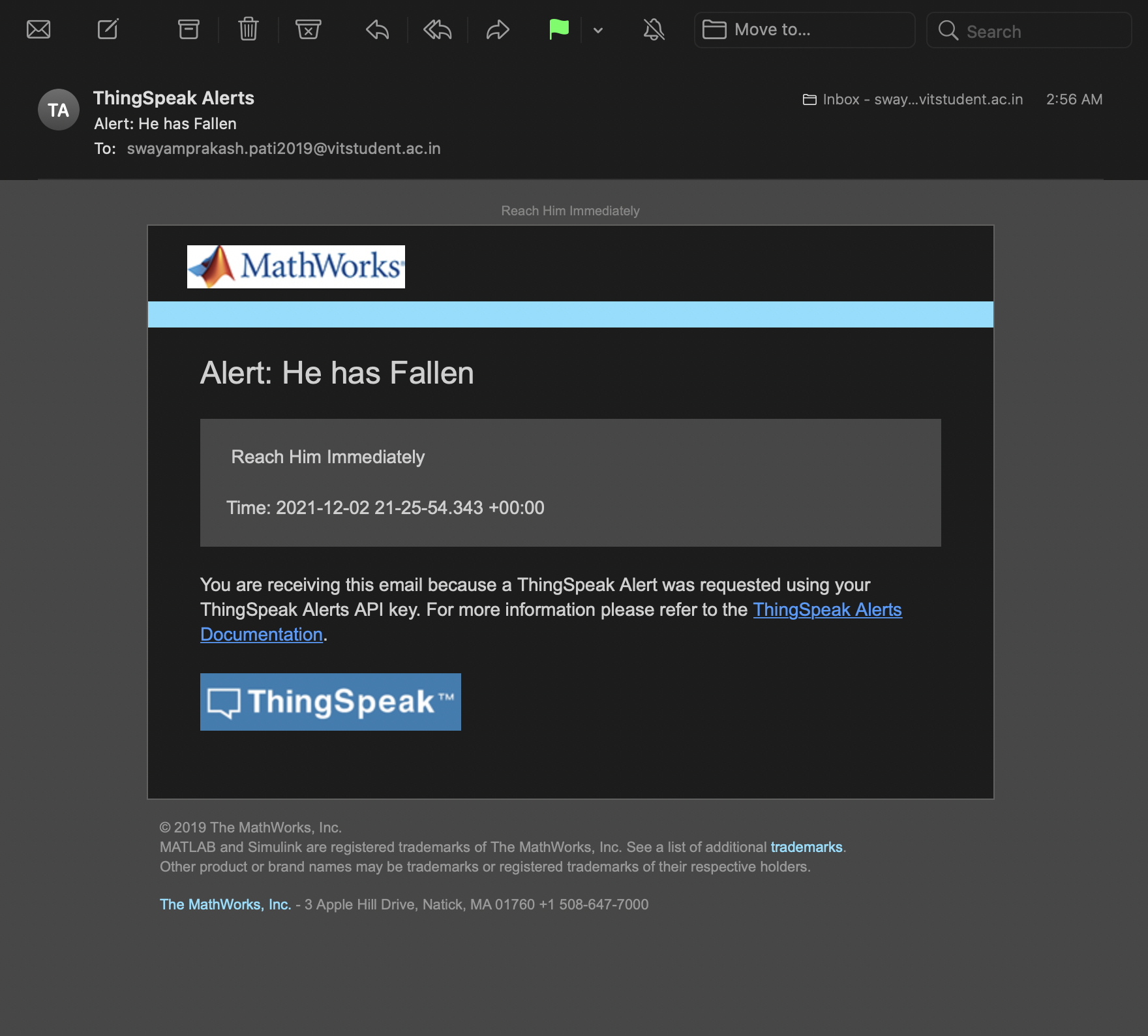


The Scan function can be used to scan and fetch data based on the wish of the user. For example, here the data entries with the parameter value of “Motion Detected” as “True” are scanned and displayed.

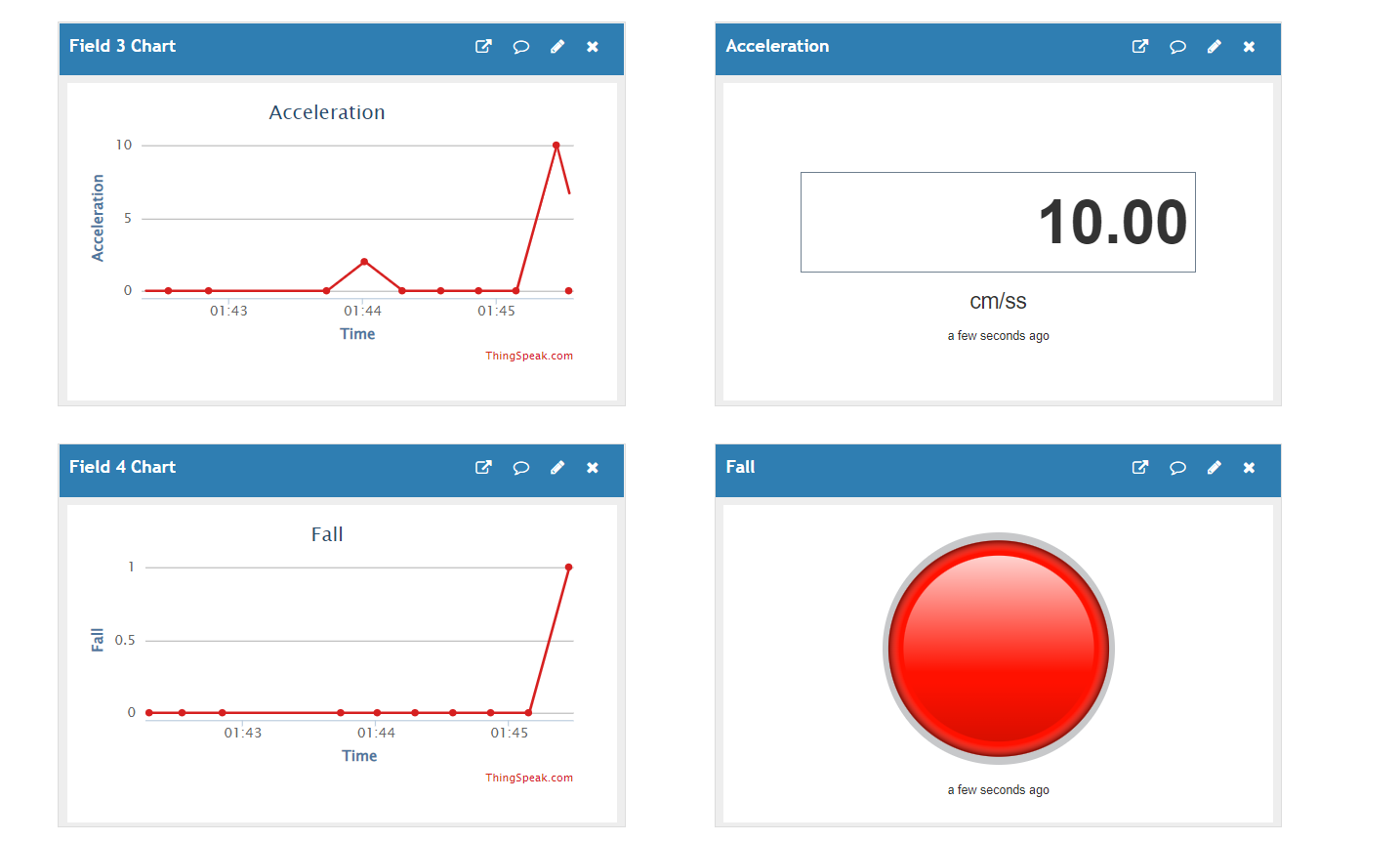
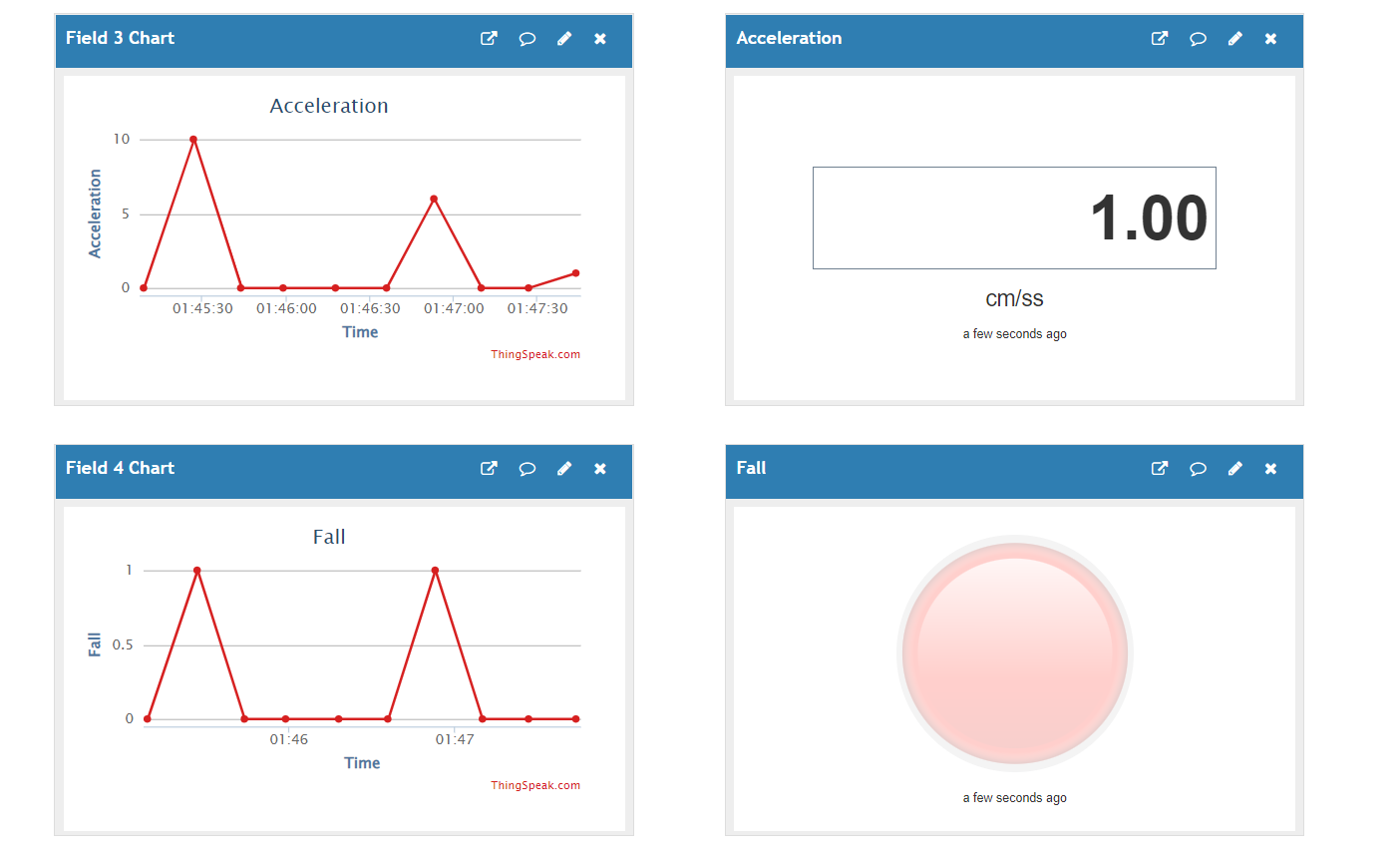


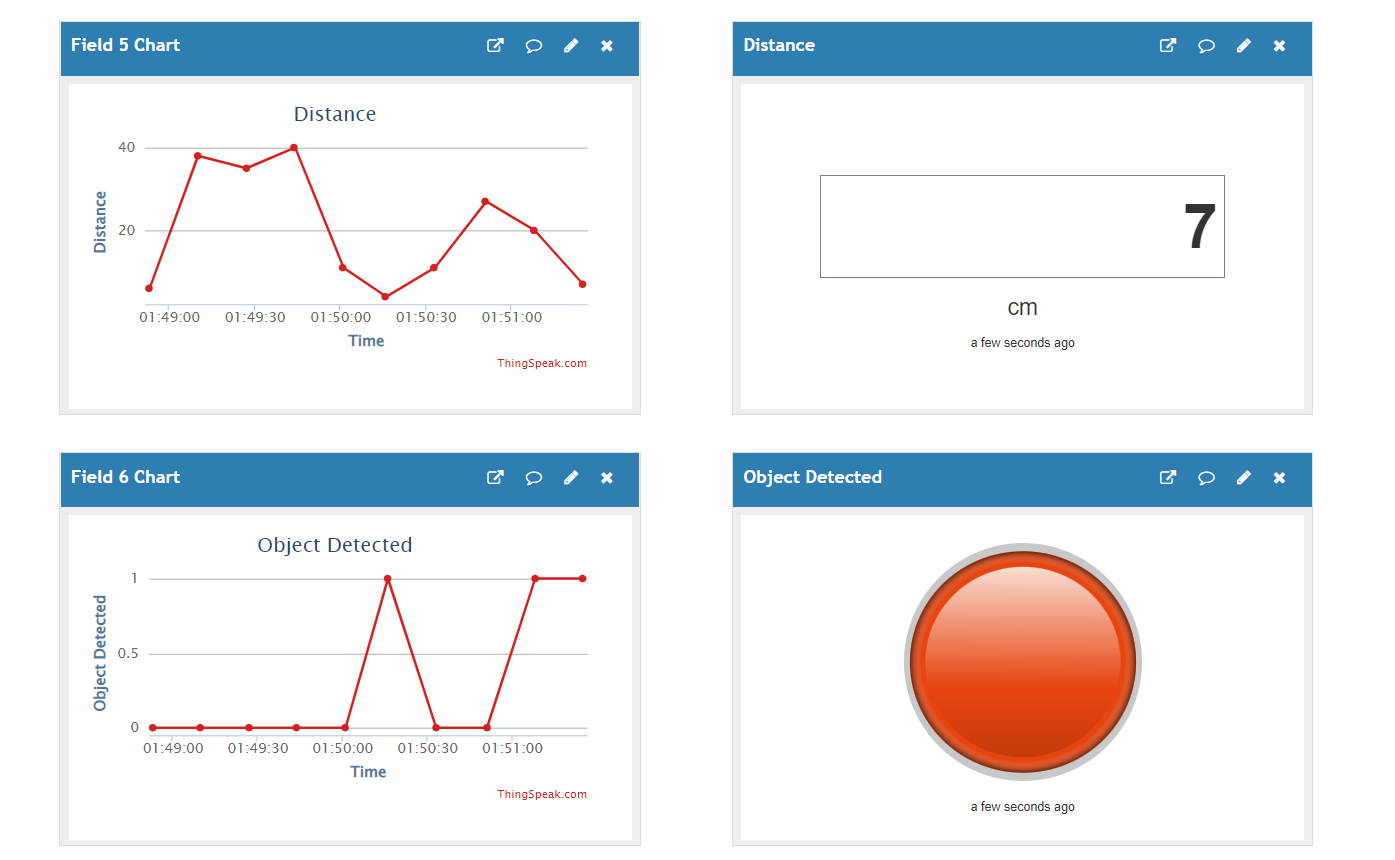


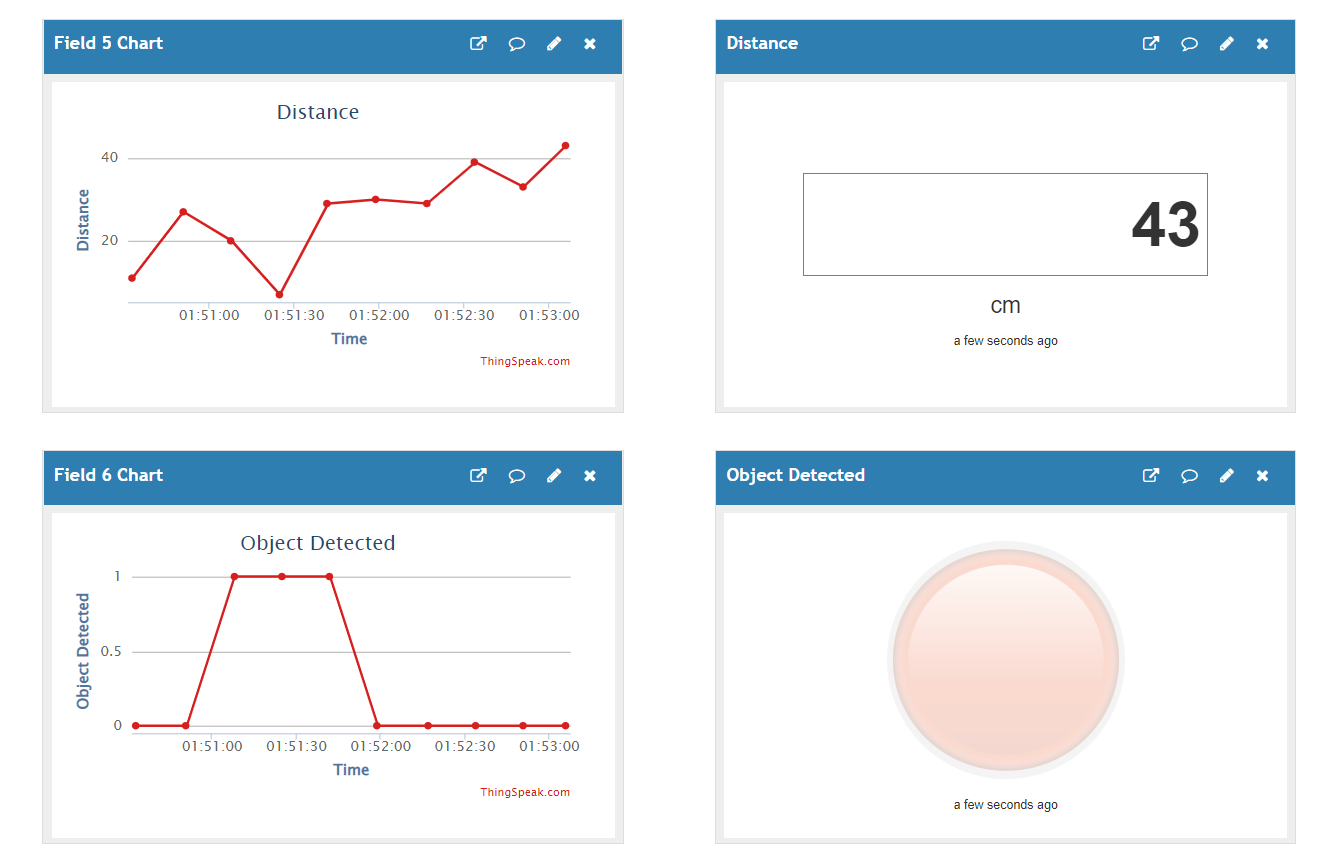
The alert is sent as soon as the incoming data crosses the input parameters. For example, here as soon as the value of the parameter “fall” becomes “True”, an alert is sent to the user.

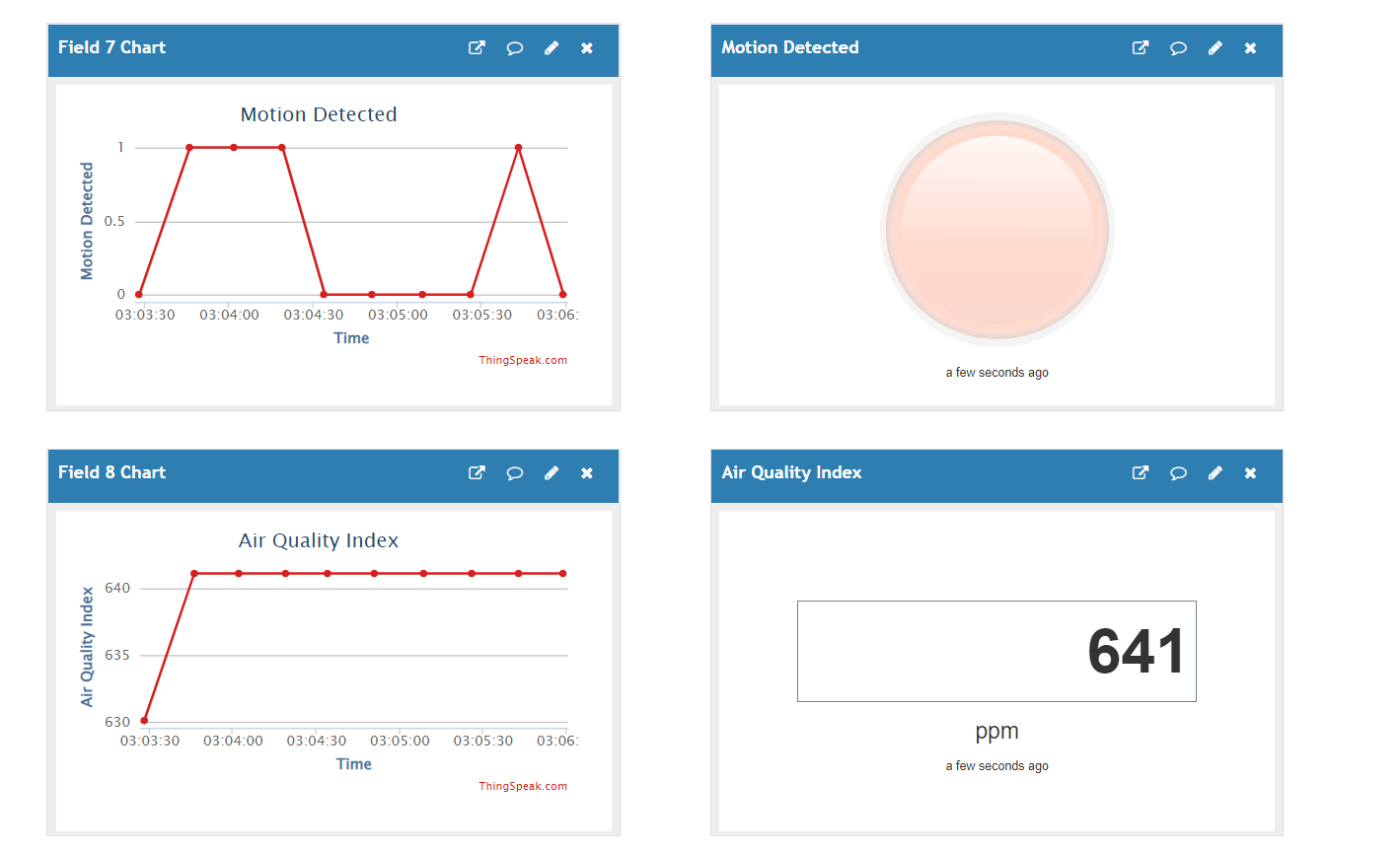
****

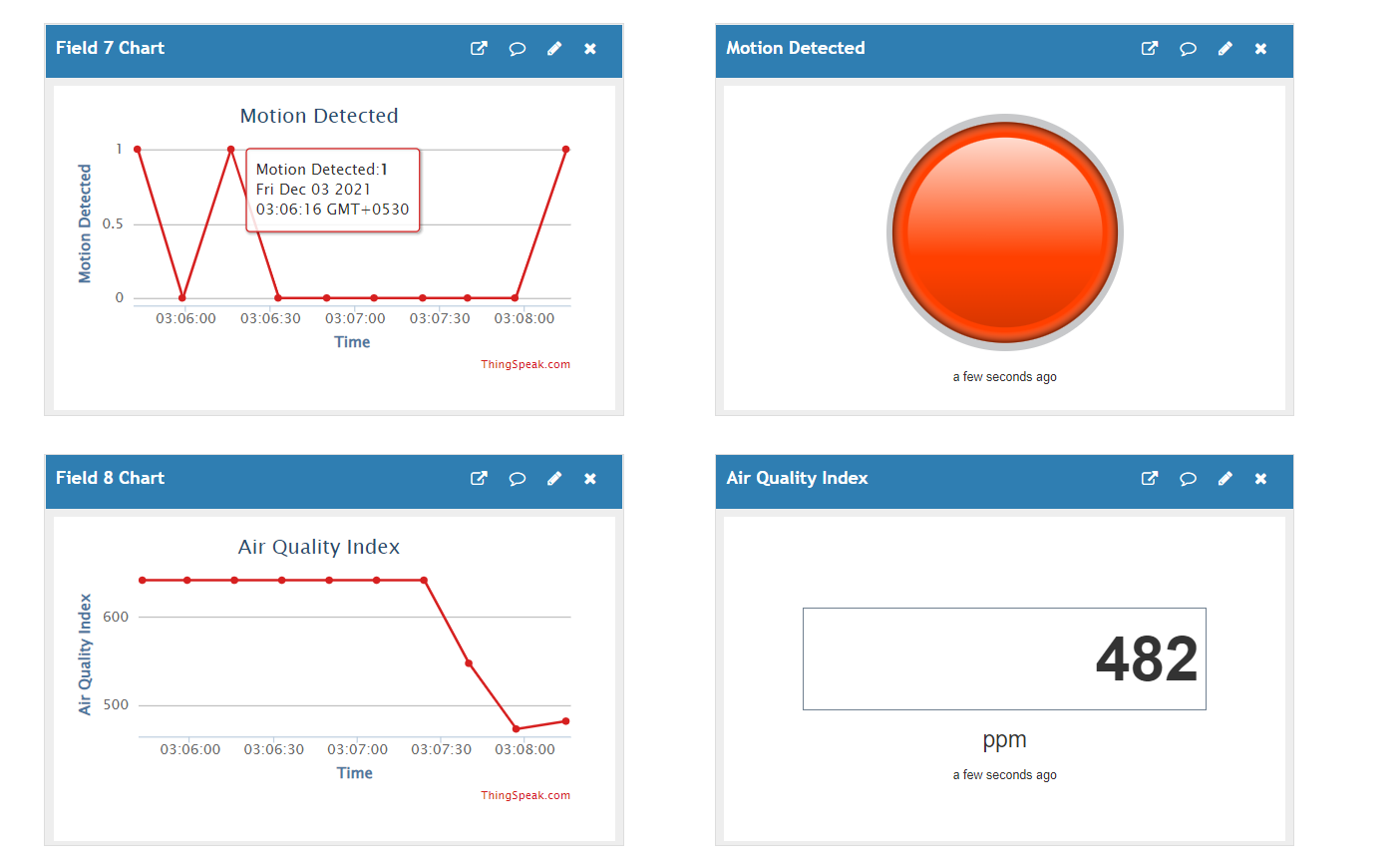
The statistical data is displayed in the ThingSpeak platform in the form of graphs and buttons with the help of widgets.



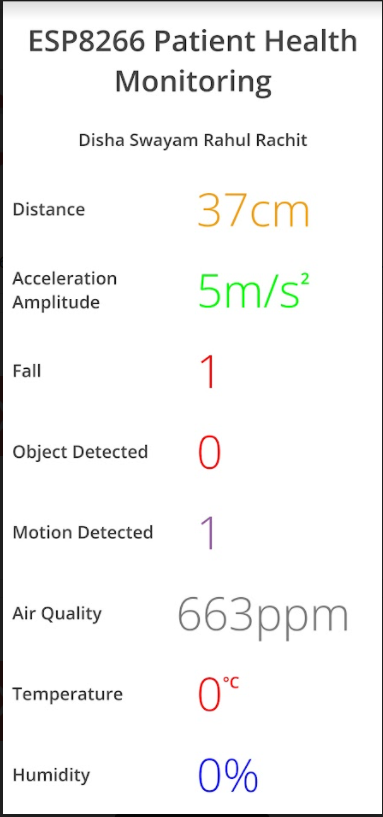








All the values are displayed in the user interface as well in real time.



**IV. Conclusion**

The project focused on retrieving physiological data from the wearable sensors worn by the patients and storing it in the database as well as displaying it at all times to the health monitoring department to avoid any casualties. The network of wearable sensors was created with the help of jumper wires and a breadboard. The microcontroller ESP8266, acting as the gateway successfully transmitted the data to the cloud with the help of the AWS IoT platform. The data was fetched from the microcontroller and was sent to the User Interface for the healthcare monitoring department to see and keep a check. The system works properly and efficiently. The data is transmitted in real time and is recorded in a timely. The statistical graphs are also displayed accordingly. The alerts are also sent as soon as any of the incoming parameters cross the threshold values. application. The propose of obtaining complex data values like angle and tilt and then the further system can be used for the gym and fitness industries for posture. Right now, the proposed system has a limited battery source which can be improved as well to accomodate more sensors.

**V. Future Work**

The system of wearable sensors can be improved by adding more efficient sensors for providing more accurate and detailed results. The multiple application sensors would be very beneficial as that would reduce the cost and number of sensors used in the system. Many creative low cost sensors exist but are not used that often. With a good amount of research in the sensor domain, low cost sensors can be introduced to the system making it even more affordable for the hospitals and the end users.The network of sensors can be made more compatible and comfortable for the patients.

The functions can be used to generate more physiological parameter details based on the incoming data using mathematical formulas.

Using Natural Language Processing and Machine learning, the sensor data can be used to predict route patterns and help can be provided more efficiently in case of emergency.

With the help of Machine Learning, datasets from Kaggle or UCL can be used for disease prediction as the function will be working for a finite number of diseases. This increases the flexibility and accuracy of the disease prediction function of our system. New datasets can be generated using our algorithm for further prediction of diseases which did not exist in the detection applications of the real world.

Human requirements for comfort, durability, and often aesthetics are always changing, so technological capabilities in terms of accuracy, performance, and modalities could be explored and worked on accordingly. As Big Data and IoT wearables become increasingly prevalent, they are likely to enable new medical tools and biomarkers for the elderly, enabling home care and pharmaceutical treatment by speeding up and optimizing clinical trials.

**VI. References:**

* <https://electronicsinnovation.com/storing-esp8266-data-into-amazon-dynamodb-using-aws-iot-coremqtt-arduino/>
* <https://electronicsinnovation.com/how-to-connect-nodemcu-esp8266-with-aws-iot-core-using-arduino-ide-mqtt/>
* <https://aws.amazon.com/console/>
* <https://stackoverflow.com/>
* <https://thingspeak.com/>