Smart Lab

Manish Mahesh Dalvi

University of Stuttgart

Abstract. The objective of the study project is to build an IoT system which has the capability to obtain the telemetry data of a room such as temperature, humidity, carbon-di-oxide, Indoor Localisation location and Actuation. It displays all the data from various sensors over one common platform. This platform also has the capability to actuate a required message unto an actuator. The entire platform is built using three layers namely sensors, middleware and Web application. The web application provides all the necessary information regarding the status of the room including the number of people entering and exiting the room. The Middle ware acts as a MQTT broker and a processing unit. The Sensors and Actuators provide environmental data and act upon the surroudings.

Keywords: IoT · MQTT · Smart · RaspberryPi · Angular · Node · Actuation · Location.

1 Introduction

The overall application of the smart lab telemetry is to provide status and information of the Lab on one common platform and also control required devices using the web application. Since there are 3 layers there has to be a mode of communication required in order to exchange data. The mode of exchanging data is over MQTT and using the standart JSON format for ease of expansion. The Web page gives the current status of the room in which the System is deployed. The System is able to detect number of people entering and exiting a room. It can also detect if there is motion of actual person rather than movement of non living objects.

2 System Architecture Design

The architecture of the Smart lab study project [Fig.1] consists of three layers communicating among themselves via MQTT. The Sensors are connected to the Raspberry pi and the telemetry data are sent to the MQTT broker, to the Web application back end which is built on Node.js. The webpage is built using the Angular as front end which sends GET request to the Backend every minute to obtain fresh data. There are three main parts in the web page namely telemetry data of sensors, Location of the Indoor localisation and Actuation button.

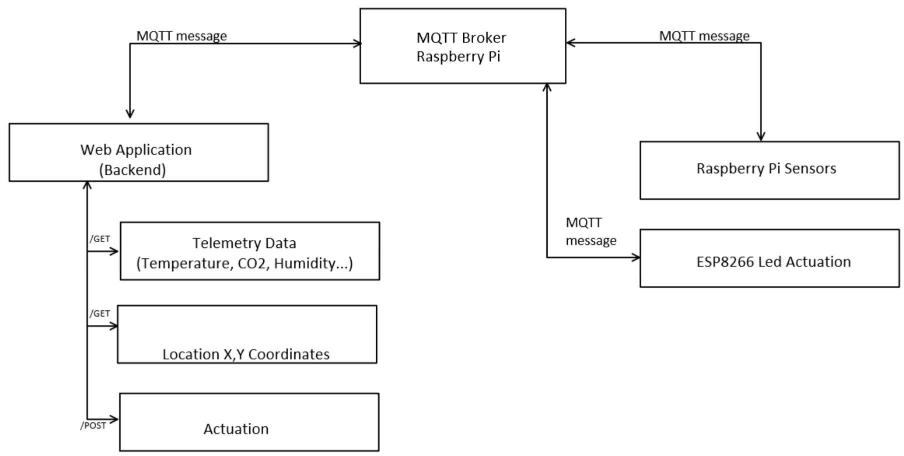

Fig. 1. System Architecture of Smart Lab Application

3 System Implementation

3.1 Telemetry data of Sensors

The first tab of the web page has 4 graphs for Temperature, Humidity, CO2 and Number of people entering a room data. The temperature, Humidity [Fig.3] and CO2 data is obtained basically from their respective sensors connected to the Raspberry Pi. The fourth data of the number of people entering and exiting a room is obtained by having two ultrasonic sensors call them U1 and U2 which can be placed at either side of the door. When a person is entering the room he will pass through the Sensor U1 then U2 and when he exits the room he will first pass by U2 then U1. Using this logic, the counter increments and decrements hence we can know about the number of people entering the room one at a time [Fig.2]. This part of the page also provides movement detection of human with the help of PIR Sensor.

3.2 Location Beacon data

The location tab of the web page has a graph [Fig.4] which is used to plot the incoming location x and y coordinates from the Indoor Localisation result. Apart from this we can add the location of the stationary beacons as a reference which would tell us how the location algorithm is performing with respect to the placed beacons. The page also provides restrictions for access of zones, for example, if a person moves into a zone where he/she is not allowed, then the page displays a warning stating that the person should not be in this zone. This feature is helpful in places like Airport and Hotels where there is access to certain areas only for the staff members.

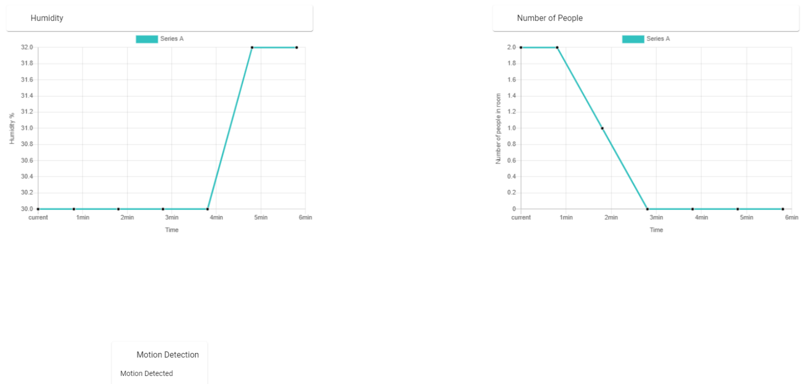

Fig. 2. Telemetry data of Sensors

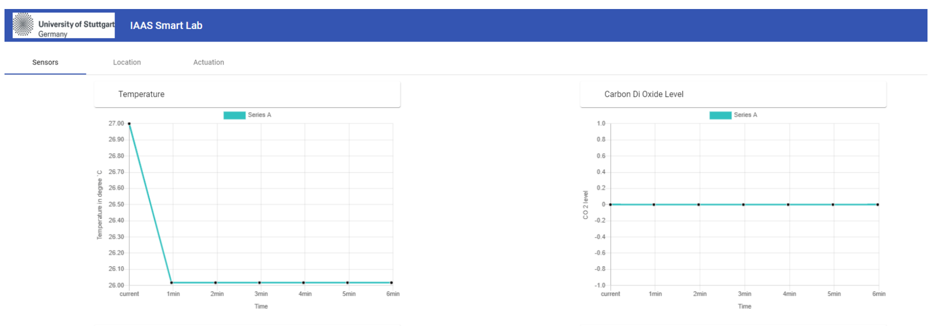

Fig. 3. Telemetry data of Sensors

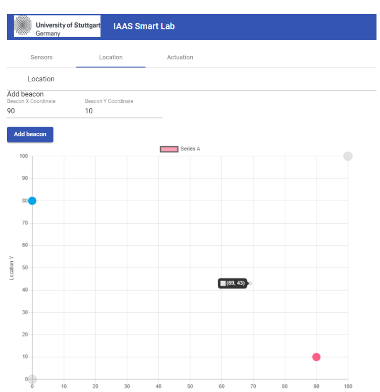

Fig. 4. Indoor localisation Location

3.3 Actuation

The Actuation tab [Fig.5] has one check box which when clicked turns on and off the led on the ESP8266 WiFi Module. When the checkbox is clicked upon, there is a post request to the backend of web application which forwards the request further to respective devices by publishing the data over MQTT. The ESP8266 which subscribes to the topic will check the message and turn on or off the led and perform the required action.

The MQTT Publish/Subscribe has been generalised as shown in Table.1 and Table.2 so that it can be used in future work with minimal changes. MQTT publish is used mainly for actuation and subscribe for obtaining telemetry data.

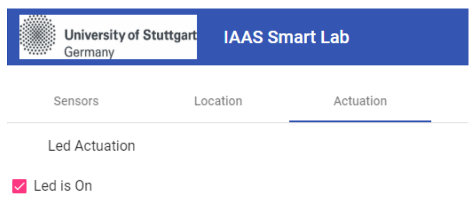

Fig. 5. Actuation Tab

Table 1. Mqtt Publish Address format

Sensor	MQTT Publish address
Sensor_type	sensor_data/sensor_type/device_id
Ex. Temperature	sensor_data/temperature/device_1

Table 2. Mqtt Subscribe Address format

Sensor	MQTT Subscribe address
S heightActuate	iaas_actuate

4 Devices and Softwares

This section gives the details of the hardware devices, framework for web development, software tools and programming languages used.

4.1 Hardware

1. **Raspberry Pi 3B+** - The Pi acts as the middleware communication device between the hardware and the web interface. It is connected to WiFi and contains the MQTT Broker through which all other devices communicate over the WiFi network.
2. **Grove Pi+**[5] - It is a board that can be placed over Raspberry Pi to provide ease of connecting the sensors.
3. **Grove Ultrasonic Range V0.2**[6] - Ultrasonic sensor to estimate the distance of an object in cm. It is connected to Grove Pi+ board.
4. **Grove MH-Z16 CO2 Sensor**[6] - Provides the CO2 levels in ppm. Connected to Grove Pi+ board.
5. **Grove PIR Motion Sensor v1.2**[6] - Senses if there is motion of a human. It can be used along side Ultrasonic sensor in order to detect human movement rather than object movement. It is connected to Grove Pi+ board.
6. **Grove Temperature and Humidity Sensor DHT11**[6] - It provides the temperature and Humidity by using the DHT11 library from GrovePi library. It is connected to Grove Pi+ board.
7. **Arduino ESP8266 WiFi Module** - Provides WiFi capability hence can be used to connect to the MQTT Broker. In this application it is used for actuation of LED. It can also be extended by adding more actuation devices to the ESP8266.

4.2 Framework

1. **Angular 8**[1] - Used for developing the front end of the web application since it provides component wise two way binding, it is useful in application of Sensing and Actuating. Sends GET request from particular component for fetching sensor data. Sends POST request for actuating an actuator. The requests are sent to the back-end.
2. **NodeJS and ExpressJS**[2] - ExpressJS is used for easier building of the server. NodeJS is used to handle the GET and POST requests from the front-end. It also communicates to the MQTT Broker via MQTT publish/subscribe for fetching sensor data and also actuating.

4.3 Software Tools and Languages

1. **Pycharm**[4] - For coding the Sensor and Middleware part using Python 3.7[3].
2. **Visual Studio Code**[10] - Entire web application written in this software tool as it provides plugins which ease the work. Nodemon[11] to run the back-end server and AngularCLI[1] to run the frontend webpage. Front end used Typescript and backend used Javascript as the base programming language along with HTML and CSS.
3. **FileZilla**[8] - Helpful in transfer of files from System to Raspberry Pi over the network.

4. **Putty**[9] - To access the Raspberry Pi terminal in order to execute programs of Middleware.
5. **Arduino IDE**[7] - To program ESP8266 WiFi Module which is used for actuation of LED . It is programmed in Arduino language.

5 Discussion and Conclusions

The overall Smart Lab system can be integrated with any room which has a WiFi connection without an Internet as it is required for only local MQTT. Since there is telemetry, actuation and location data all within one application it would be easier to control lighting from within the application. The System can be extended for more sensors and actuators as the foundational system is provided and expansion is very simple since the basic functionality of communication and message passing are already built. As future work, the System can be extended by providing a Database and run analytics over the same for decision making. Currently the System works only for one room and over one Edge computation device without talking with other Edge computing devices. In future, Addition of more intercommunicating Edge computing devices can be added so as to extend to the usefulness of the application.

6 References

1. https://cli.angular.io/
2. https://nodejs.org/en/
3. https://www.python.org/
4. https://www.jetbrains.com/pycharm/
5. https://www.seeedstudio.com/GrovePi.html
6. https://www.dexterindustries.com/GrovePi/supported-sensors/
7. https://www.arduino.cc/
8. https://filezilla-project.org/
9. https://www.putty.org/
10. https://code.visualstudio.com/
11. https://www.npmjs.com/package/nodemon