|  |
| --- |
| See the source image  **IOT Internet Of Thing Management Systems** |
| By |
| **Ruba Almashhuri**  **D15125511** |
|  |
| *Submitted in partial fulfilment of the requirements for the degree of* |
|  |
| **BSc in Business Computing** |
|  |
| Technological University Dublin (City Campus)  *Supervisor: Jenny Munnelly* |

|  |
| --- |
| **Declaration** This is an original work. All references and assistance are acknowledged.  **Signed :** |
| **Date:** 1 May 2020 |

# Abstract

The Internet of Things (IoT) is a technology that is constantly developing today. This sophisticated system has various uses, including home control, data tracking, etc. More and more, home automation is resulting in greater comfort. Home automation describes homes where almost everything—devices, electrical outlets, heating and cooling systems—are fitted into a reasonably tractable device framework. Home automation requires a great number of IoT devices to communicate with one another. With the appreciable growth in the amount of devices on cloud services, the need for updating firmware emerges. It includes removing already-introduced gadgets, making necessary code changes and flashing the altered code anew. Possibly, information processing should be done elsewhere to overcome these issues. Node-RED is a visual cable tool that helps to easily connect gadgets through quick and easy connection setups. Gadgets are linked together to Arduino and a Mosquitto-based MQTT broker using Node-RED and a connection is set up for remote monitoring and control through Android mobile APP.

# Acknowledgment

Now I can comprehend the reasons students express their thanks to their friends and family for patience and understanding in their writings for you to read; it’s not for the mere reason that they are kinship or friends, but because of the amount of support and understanding from those people. This limitless support and understanding is a significant part of the realization of this project; I genuinely appreciate the people behind the scenes. Without them, it would have been impossible to deliver this thesis to your hands.

In the beginning and before anything, I thank God, the most gracious, the most merciful for empowering me to finish this thesis. With the most incredible honor and gratitude, I would like to express my thanks to my supervisor Dr. Jenny Munnelly for her guidance and support from start to finish, and I offer my regards and blessings to my colleagues, doctors and my second reader Dr. Catherine Higgins for the encouragement and help they provided.

Also, I’m deeply thankful to my parents Suad and Abdalrahman, my brothers, sisters, especially Ahmad and Ruwyda, and my dearest friends, Amnah, Rawan, Mohila, Ashwaq, Arwa, Nada, Mishael, Nora, Hanan, and Arwa Alrawas. For support and time, they have given me to finish this project. I hope someday I’ll be able to compensate each individual who helped me throughout this project.

In the end, I hope that your acceptance and satisfaction will appreciate my work and effort.

***Ruba Almashhuri***

# Table of Contents

[Abstract 3](#_Toc39192460)

[Acknowledgment 4](#_Toc39192461)

[Table of Contents 5](#_Toc39192462)

[Nomenclature 7](#_Toc39192463)

[Chapter 1: Introduction 7](#_Toc39192464)

[1.1 Introduction 7](#_Toc39192465)

[1.2 Objectives of the Project 8](#_Toc39192466)

[1.3 Business Case for Project 8](#_Toc39192467)

[1.4 Overview of Technologies Used in the Project 8](#_Toc39192468)

[1.4.1 Internet of Things (IoT) 9](#_Toc39192469)

[1.4.2 Mobile APP 10](#_Toc39192470)

[1.4.3 Node-Red and MQTT 11](#_Toc39192471)

[1.4.4 Database (MySQL) 13](#_Toc39192472)

[1.5 MySQL Database 14](#_Toc39192473)

[Chapter 2: Requirements Capture and Analysis 15](#_Toc39192474)

[2.1 System Requirements 15](#_Toc39192475)

[2.2 Requirements Analysis 16](#_Toc39192476)

[2.2.1 MySQL Database 17](#_Toc39192477)

[2.2.2 Node-RED 18](#_Toc39192478)

[2.2.3 Mobile API 19](#_Toc39192479)

[Chapter 3: Design 20](#_Toc39192480)

[3.1 System Architecture 20](#_Toc39192481)

[3.2 Choice of Technologies 20](#_Toc39192482)

[3.3 Data Model Design 21](#_Toc39192483)

[3.4 Software Design 23](#_Toc39192484)

[3.5 User Interface and Design Code: 23](#_Toc39192485)

[3.6 Issues and Resolutions 25](#_Toc39192486)

[Chapter 4: Implementation 26](#_Toc39192487)

[4.1 Overview of Implementation 26](#_Toc39192488)

[Chapter 5: Test Plan 31](#_Toc39192489)

[5.1 Test Plan 31](#_Toc39192490)

[Chapter 6: Conclusions 34](#_Toc39192491)

[6.1 Final Conclusions 34](#_Toc39192492)

[References 35](#_Toc39192493)

# Nomenclature

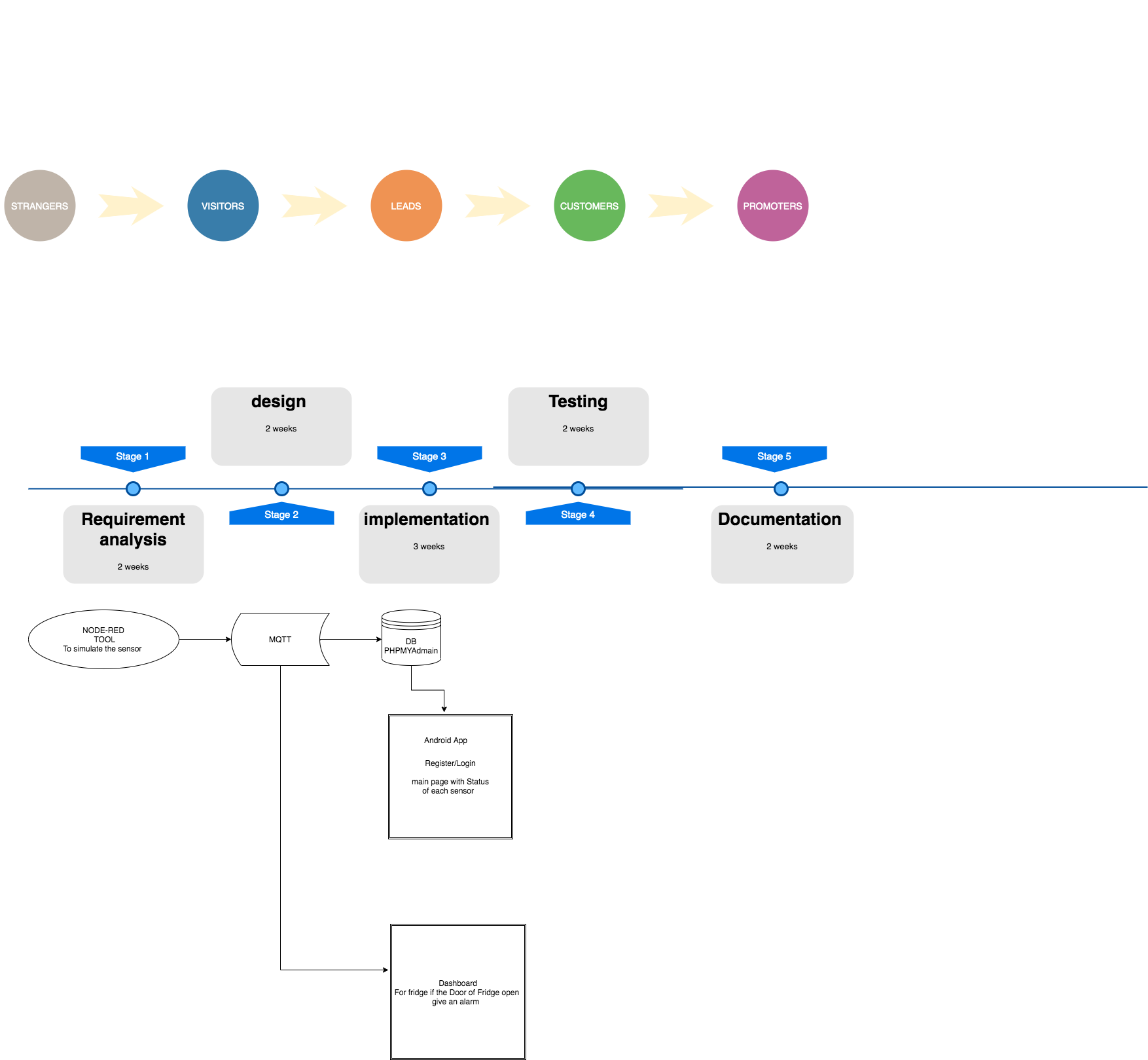
|  |  |
| --- | --- |
| **Abbreviation:** |  |
| HTTP  IC | Hyper Text Transfer Protocol  Integrated Circuit | |
| IoT  IDE | Internet of Things  Integrated Development Environment | |

# Chapter 1: Introduction

## 1.1 Introduction

Home Automation has been around for long time as far as the control of lighting and basic appliances is concerned, and only recently has technology caught up with the idea of an inter-connected world, allowing full control of the home from anywhere, to become a reality, and thus to build a smart home. Smart Home generally refers to a home where the gadgets are linked to the cloud. It can be viewed as a framework that uses smartphones and PCs to control or connect to home gadgets. With advancements in home automation, you can direct and coordinate how the gadget should respond, when it needs to respond, and why it needs to respond. Your duty is to set the schedule and the rest is automated and based on your requirements, giving control and a smart home overall. Due to universal Wi-Fi accessibility, all appliances within a home can be linked together through a common gateway, and controlled using the MQTT protocol implemented on Arduino using Node-RED. Fig-1.1 shows the general block diagram for the project.

Figure 1.1 General Block Diagram



## 1.2 Objectives of the Project

The aim of this project was to develop an effective, simple, stable, applicable and inexpensive method for monitoring the home in real time, without the need for special and sophisticated hardware, but with rather ordinary appliances that anyone might have in their possession, e.g. smartphones. By simulating (NO interacting directly with the sensor) the sensor inside the home will remotely monitor the home door status (open / close), the window status (open / close), the refrigerator door status (open / close) and the ambient temperature, and the APP will provide remote control Light (on and off) and immediately send alerts through the dashboard when extensive sitting is detected.

## 1.3 Business Case for Project

In the last few decades, the advancements in life science and technology, drugs and public health, as well as a growing consciousness over nutrition, the environment and private hygiene have paved the way for a dramatic global increase in lifespan. However, there is growing concern for the increasing senior population. Seniors usually have special needs, such as nursing homes and healthcare facilities for the elderly, where there is often a lack of available space. In case such a facility is not available, older people have to opt for in-patient care, which sometimes leads to a serious financial burden. Thus, in order to cope with the growing need for elder-healthcare services, it is essential to develop affordable, advanced and easy-to-use home healthcare solutions. With the right technology platform, elders can stay in their own comfortable home environments instead of expensive healthcare facilities, and can have the ability to control their home through the platform, thereby reducing the need for help with their most basic needs, as for example turning lights on and off, closing/opening doors and windows or checking the indoor and outdoor temperatures.

## 1.4 Overview of Technologies Used in the Project

The proposed system discussed in this study is a combination of different technologies used to achieve the project goals, listed below:

* Internet of things (IoT)
* Android Mobile APP
* Node-Red tool and MQTT
* Database (MySQL)

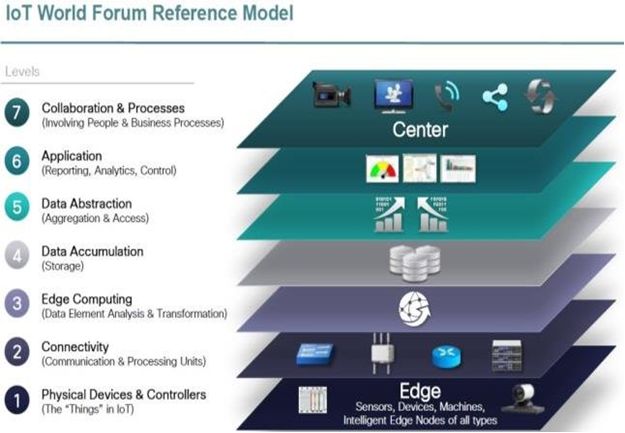
## 1.4.1 Internet of Things (IoT)

The expression "Internet of Things (IoT)" was initially used by Kevin Ashton in 1999, the pioneer of British innovation. As indicated by Ashton, Internet of Things characterises the arrangement of physical articles on the planet that interface with the web by means of a sensor. The IoT segments were condensed in a condition structure with the mix of sensors, physical articles, controller and actuators. The expression "Web of Things (IoT)" is an umbrella watchword that covers the different highlights, as for example the expansion of the web, the web as a physical domain, organisation of broad, installed circulated gadgets, sending and activation capacities. The term IoT is likewise called the future web.

The depiction of the IoT is identified with various definitions used by various sources for advancing the specific idea globally. As per the Internet Engineering Board's (IAB) definition, the IoT is the system administrations of shrewd items, which mean a colossal number of gadgets insightfully imparting within the sight of web convention that can't be straightforwardly worked by people, yet exist as parts in structures, vehicles or the condition. As per the Internet Engineering Task Force (IETF) association's definition, the IOT is the system administration of brilliant items, in which keen articles have a few imperatives for example, constrained data transfer capacity, power and preparing openness for accomplishing the inter-operability among keen articles. As indicated by the IEEE Communications class magazine's definition, the IoT is a system of everything that has a portrayal in the nearness of the web, so that new applications and administrations empower the cooperation in the physical and virtual worlds as Machine-to-Machine (M2M) correspondence in the cloud. As indicated by Oxford word reference's definition, the IoT is the collaboration of ordinary article's processing gadgets through the Internet that empowers the sending and accepting of valuable information.

The IoT World Forum Reference Model is shown in Figure 1.2. The Internet is essentially a customer server framework. In the recovery of data that can be obtained using the web, there are two significant parts: the customer, who demands the data and the server, which stores it. Each side requires a bit of programming to arrange the information. The link issue identification over the IoT identifies the precise shortcoming position to fix the proper test work for the less expertise technician. The expertise technician precisely knows which part has a shortcoming and just that zone to distinguish the deficiency source. The IoT innovation enables the experts to screen and examine issues on the web. At whatever point a shortcoming is present at a point shorting two lines together, the voltage is produced according to the resistors arrange blend. This voltage is detected by the micro-controller and is refreshed to the client. The data, passed on to the client, is the separation to which that voltage compares. The micro-controller recovers the separation point information and displays it over the LCD. It moves this information over the web to display on the web.

Figure 1.2 *IoT World Forum Reference Model*



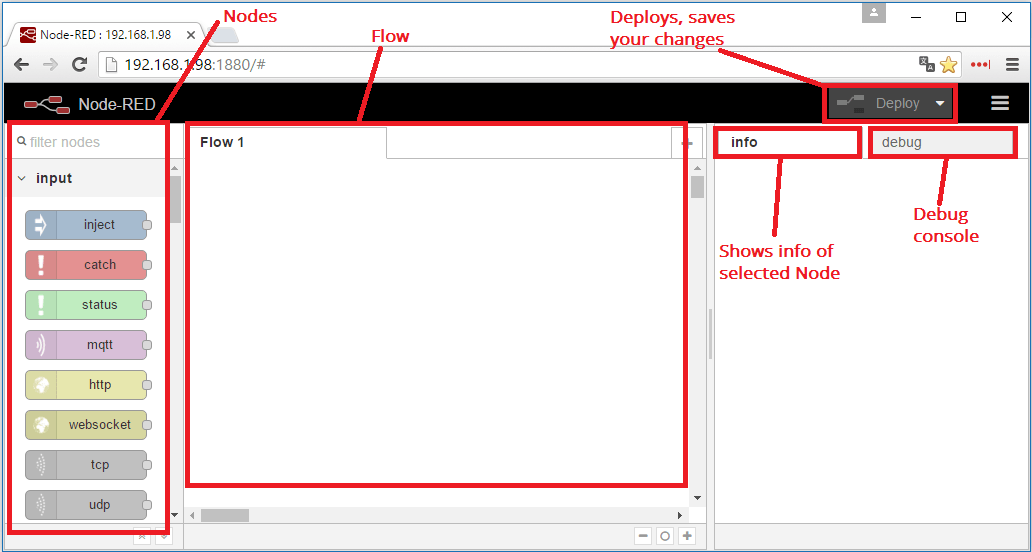
## 1.4.2 Mobile APP

A mobile application is also known as a mobile app, or the simplified word (app) is a software application programmed to run on a mobile device, such as a phone, tablet, or watch. Apps were initially intended for productivity support, such as emails and communication databases. But due to the growing appetite for apps, it led to rapid expansion to other sectors, such as gaming apps, business automation, location-based services apps, and ticket sales apps. Generally, Apps are downloaded from an application operated by the manufacturer of the mobile operating system. The most popular examples are the App Store (iOS) or the Google Play Store. Some applications are free, and others are priced, with the benefit being split between the developer of the application and the delivery company. Mobile applications are the contrast to desktop applications that are designed to run on desktop and web applications that run on mobile web browsers, rather directly on mobile devices.

## 1.4.3 Node-Red and MQTT

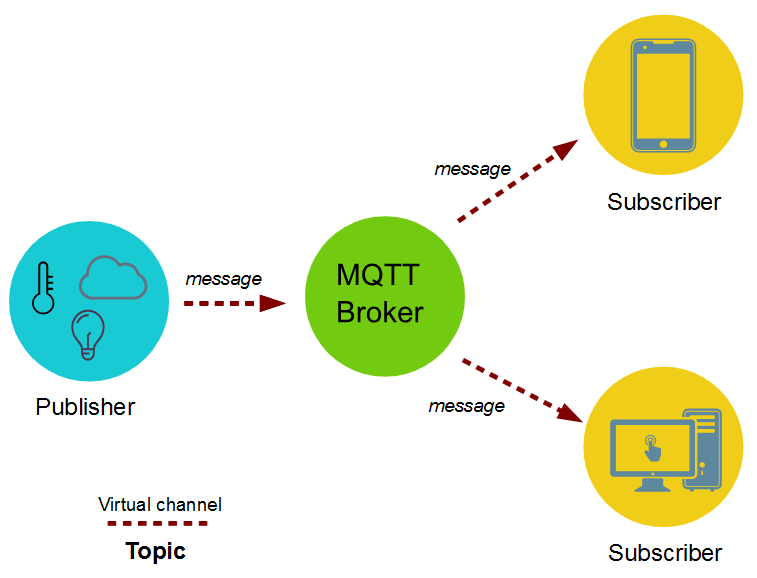
The Internet of Things (IoT) now does not just mean 'diverse things,' but has evolved into 'smart things' that have onboard calculations and system associations. In particular, they have the capacity to feel the environment around us, and therefore to act shrewdly. Node-RED is a tool that programmed to connecting APIs, hardware devices, and online services. Node-RED is an open-source service created by IBM Emerging Technology. It is essentially a visual programming tool intended for the Internet of Things, but can also be used for various applications to quickly assemble the flows of different services. It is based on Node.js (the server-side java scripting platform). Node-RED enables users to link web services and gadgets together by replacing common coding tasks, and this should be possible with a visual drag-drop interface. Various components in Node-RED are linked together to create a flow in the Node-RED editor.

**Figure 1.3 Node *Red Editor Overview***



Node-RED streams are stored using JSON, which can be easily imported and exported for sharing with others. Nodes are gathered together in a flow-holding system to achieve a pre-determined objective. Flows can also be assembled in a coherent manner to achieve higher-order objectives. Node-RED can run on a local browser at http:/localhost:1880.It also has a user interface module that provides node setup in Node-RED to quickly create a live data dashboard.

MQTT provides a lightweight messaging protocol which uses a publish/subscribe model, because of which it is worthy to be used on all devices from low power boards to servers. A client can publish or subscribe to a topic or do both. The broker receives all the messages, filters them and sends them to the subscribed client. It is based on a TCP / IP stack top. So both the client and the broker should have a TCP / IP stack. The publish/ subscribe pattern of the MQTT broker allows messages to be pushed to client devices that do not require gadgets that need to constantly search the server. MQTT can be secured with SSL over web sockets.

Figure 1.4 *MQTT Overall Ideas*

The MQTT connection itself is always between one client and the broker, and a client is not connected to any other client directly. A client initiates the connection by sending a CONNECT packet to the broker and in response, the broker sends a CONNACK packet and a status code. Now the established connection may be terminated after some time-out. To avoid this, the client passes a PINGREQ packet, to which the broker reacts by sending a PINGRESP packet and keeps the connection alive. Once the connection is set up, the broker will keep it open till the client doesn’t send a disconnect command or it loses the connection. And to quit the connection, the client just needs to send a DISCONNECT packet to the broker. Even if the client does not send a packet that it is alive the connection will be terminated.

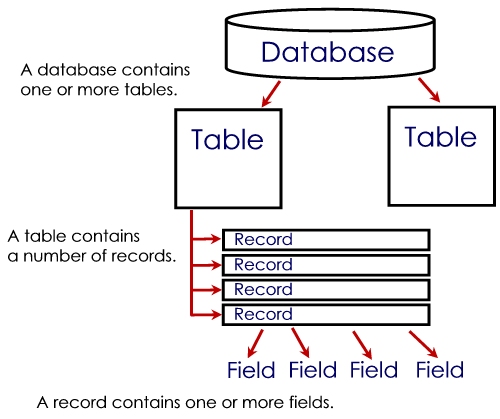
## 1.4.4 Database (MySQL)

A database is an application dedicated to storing collections of data. Each database has one or more distinct APIs for creating, accessing, managing, searching, and replicating the data it holds. Different kinds of data storage can also be used, such as files on the file system or hash tables, but fetching and writing won’t be so fast and easy with such systems.

Currently, the concentration is on the usage of relational database management systems (RDBMS) to store and control huge volumes of data. They are called relational databases because all the data is stored into different tables and relations are established using primary keys or other keys known as Foreign Keys.

* + A Relational Database Management System (RDBMS) is software that entails the following:
* Enables you to implement a database with tables, columns, and indexes
* the Referential Integrity Guaranteed between rows of various tables

automatically updates the indexes Interprets an SQL query and combines the information of various tables

Figure 1.5 *Database General Structure*

## 1.5 MySQL Database

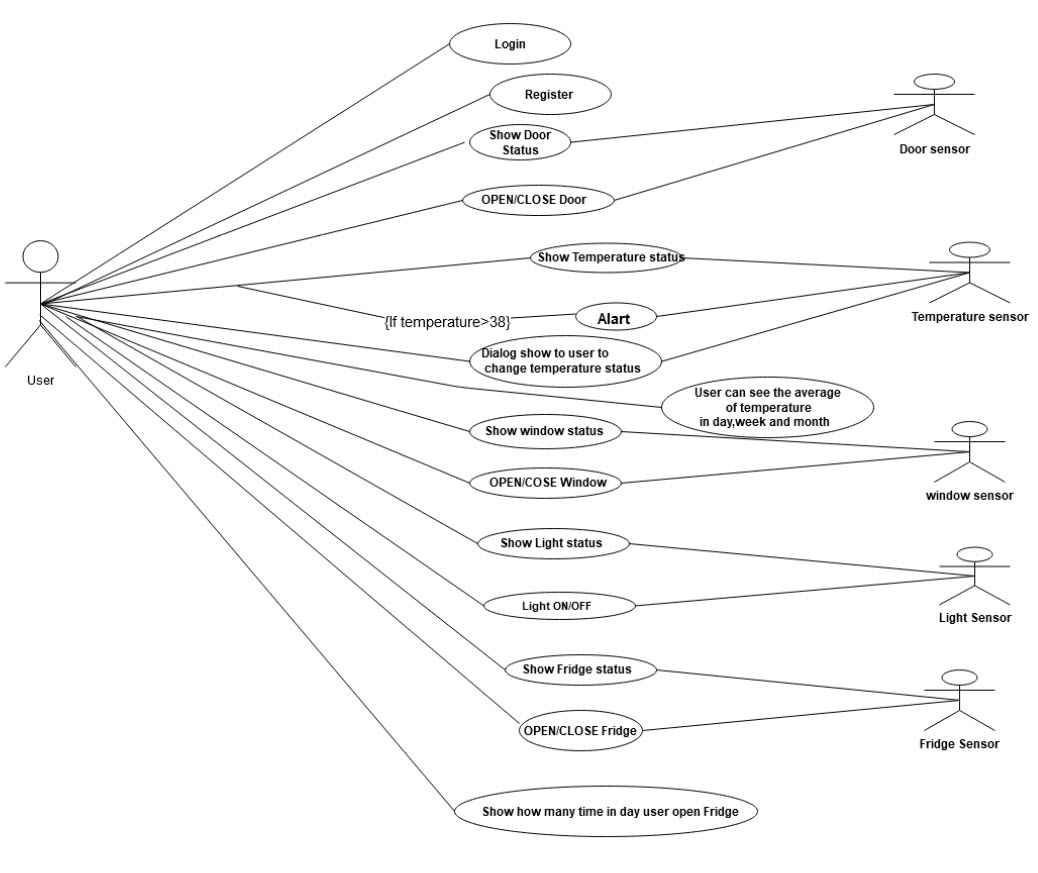
MySQL is an easy-to-use RDBMS used by different businesses, whether small and big, and it’s developed and marketed supported by the Swedish company MySQL AB. MySQL is becoming popular for many good reasons; it’s released with an open-source license. So it’s free to use. MySQL is powerful software, with lots of the functions of the most pricy database packages. MySQL works with a standard form of data language SQL. MySQL works on many operating systems and with many languages, including PHP, PERL, C, C++, JAVA, etc. MySQL works very quickly and well, even with large data sets. MySQL is very friendly to PHP, the most appreciated language for web development. MySQL supports large databases, up to 50 million rows or more in a table. The default file size limit for a table is 4GB, but you can increase this (if your operating system can handle it) to a theoretical limit of 8 million terabytes (TB).MySQL is customizable. The open-source GPL license allows programmers to modify the software to fit their own specific environments.

# Chapter 2: Requirements Capture and Analysis

## 2.1 System Requirements

* The system shall allow for on-line monitoring of different parameters in the home, including door status, fridge door status, window status and heater temperature.
* The system shall provide ability to control the home light bulb (On / off).
* The system shall display information through Android mobile APP and web dashboard.
* The system should be able to send alerts through the App when indoor temperature is above or equal 38 C.
* The system should be able to send alerts through the dashboard when Fridge door is open.
* The system should be able to simulate all sensors using Node-Red and MQTT technology.
* The system should be logging all data to the database.
* The system should be calculating average of temps per day, week, month and display information through Android mobile APP.
* The system shall display information of how many per day door of fridge was open through Android mobile APP.

Figure 2.1 *Business Process Overview*



## 2.2 Requirements Analysis

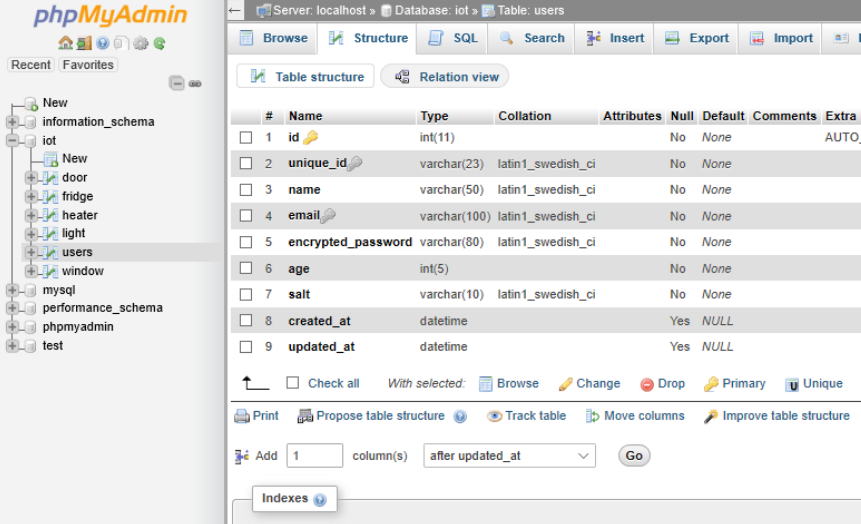
In this section, all system requirements analysis is illustrated by describing each segment of the proposed system as follows:

### 2.2.1 MySQL Database

First of all, we have to design the database of the proposed system, which is the back-end of the entire project. By opening a web browser and typing the public IP address of the VPS, a welcome page of the Apache Server will load.

Create a new database schema by clicking on PhpMyAdmin at the top right corner of the page and type the name of the system database which will be used later by the mobile application. Create the infrastructure for such schema by adding tables and assign N columns for each table as shown in Figure 2.2.

Figure 2.2 *Designing the Infrastructure of Database*

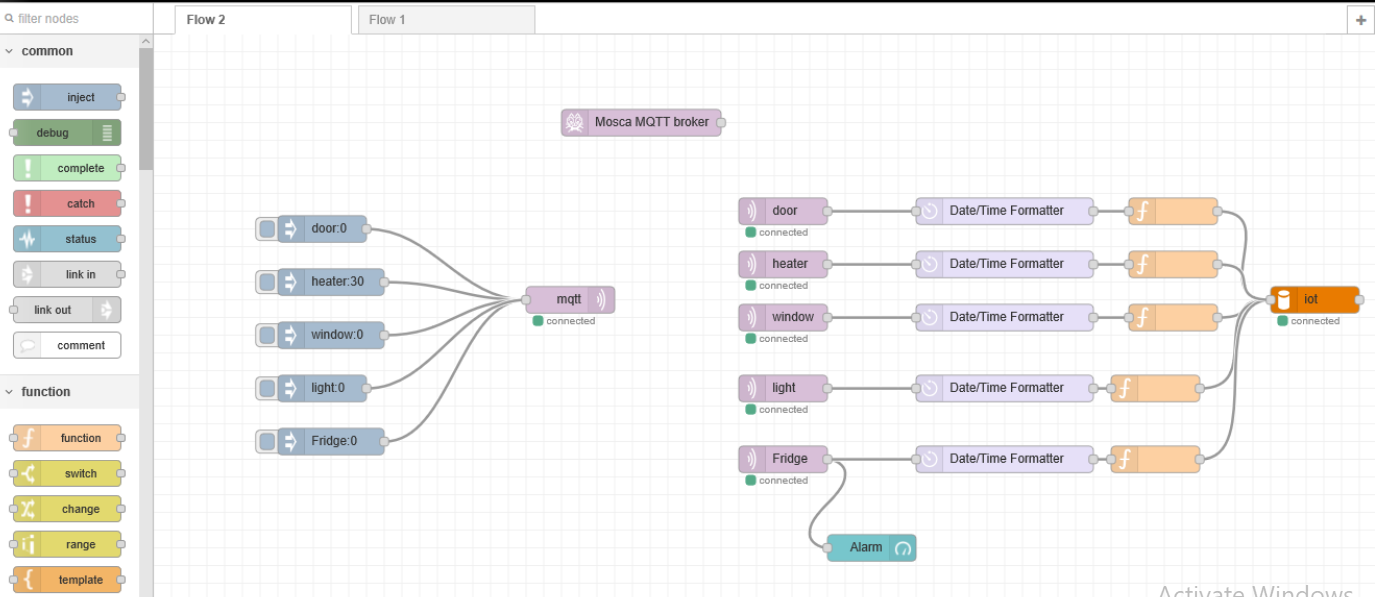
**

As you can see, the infrastructure contain vital columns like username, age, device ID, etc., which are the fundamentals for the system in which the mobile app is installed. It can modify and retrieve data to and from it. In this design, we create a schema titled “users” containing 14 different columns.

### 2.2.2 Node-RED

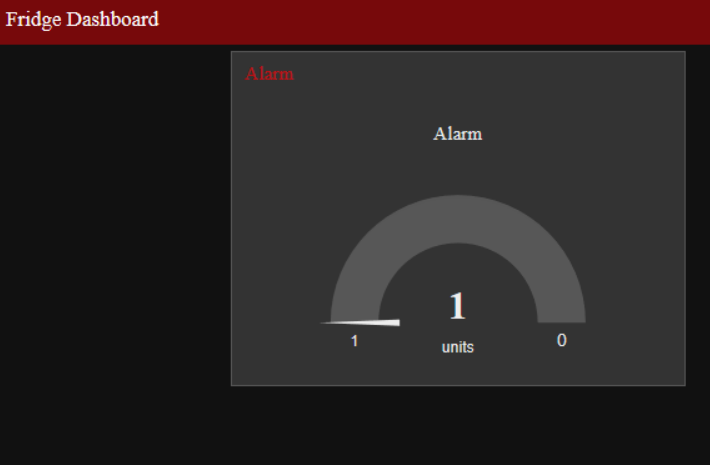
Inside the solution, a trigger is fired to simulate operations of physical sensors and a data output inserted into the previously created schema. This can be done by installing dashboard plugin from manage pallet for GUI design as in Figure 2.3.

Figure 2.3 *GUI Design of Virtual Sensors by Node-RED*

**

There are some elements in the GUI involved with executing some SQL statements (like: insert, update). These statements are used to insert a new record or update old ones respectively to the database in which these changes are reflected to the mobile app (monitoring). All virtual sensors are bonded to a “Connector”, which transmits every sensor value to a recipient slot which will be sent later to MySQL database as update SQL statement and this is applicable for every single sensor. To view a certain sensor as dashboard, open a new web tab and type the IP address for node-red followed by /ui. The response will be as depicted in Figure 2.4.

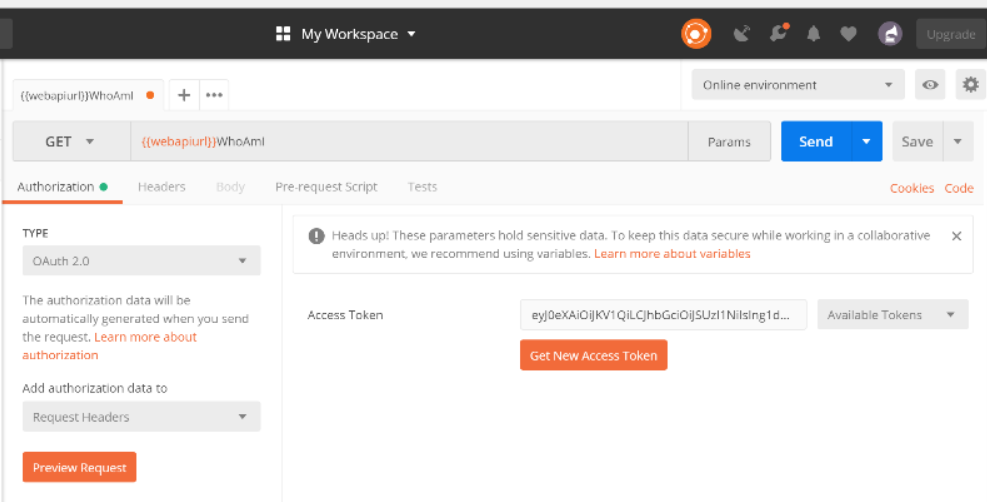
Figure 2.4 *Fridge Sensor Dashboard View*



The previous steps were performed for a certain user (e.g. id = 1). In order to monitor another user fridge door, one has to change the id value under SQL select statement.

### 2.2.3 Mobile API

In this section, the creation of the API is explained, so that it can work as a bridge between the mobile app and Node-RED to get the current values for deployed sensors and also used for the database to check the credentials of users. API can be tested by using “postman” to simulate requests coming from a mobile device, as in Figure 2.5 below.

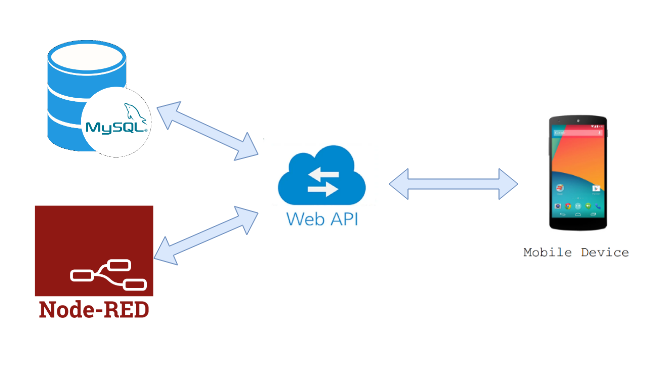
Figure 2.5 *POSTMAN Tool for Testing API*

# Chapter 3: Design

## 3.1 System Architecture

Figure 3.1 depicts the system flowchart. The MySQL server is online and Node-RED as well. The mobile device connects to the MySQL database through a web API to check the validity of the entered credentials for certain user. If the posted data is correct, then the mobile app is allowed to connect to the virtual sensors through Node-RED and can receive all responses for the requests which have been issued by such device.

Figure 3.1 *System Flowchart*



## 3.2 Choice of Technologies

The technologies utilised and their corresponding version is specified in Table 3.1.

Table 3.1 Technology Versions

|  |  |  |
| --- | --- | --- |
| NO. | Technology | Version |
| 1 | MySQL Community Server | v8.0.19 |
| 2 | Node-RED | v1.0.6 |
| 3 | Apache http Server | v2.4.43 |
| 4 | Android Studio | v3.6.3 |

## 3.3 Data Model Design

The design view of the pre-designed database is found in Figure 3.2.

Figure 3.2 *Database Schema Design View*

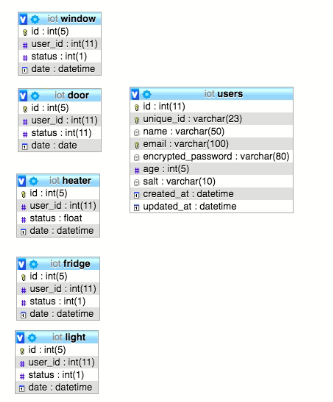
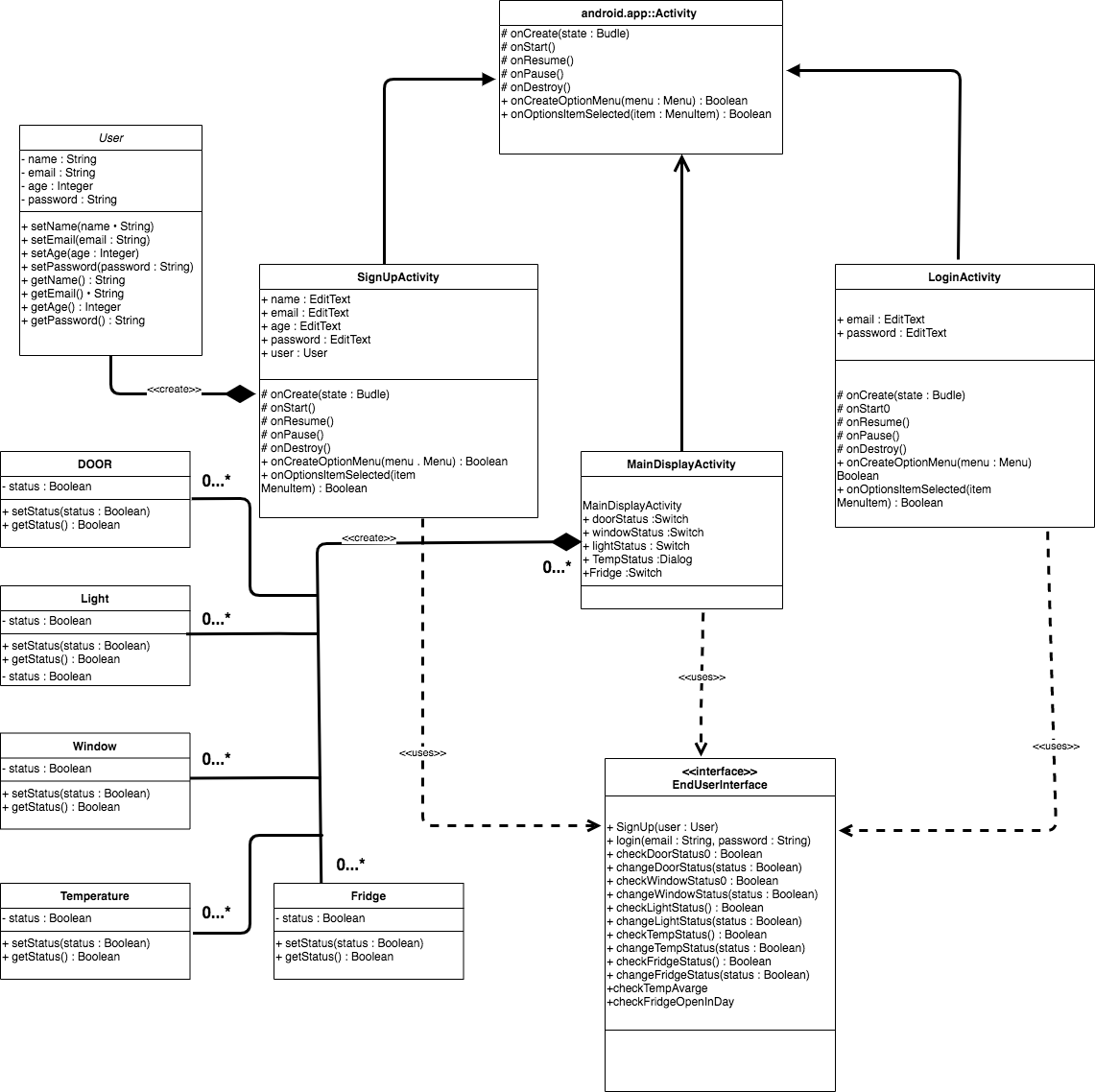


Figure 3.3 *Entity Relationship Diagram*

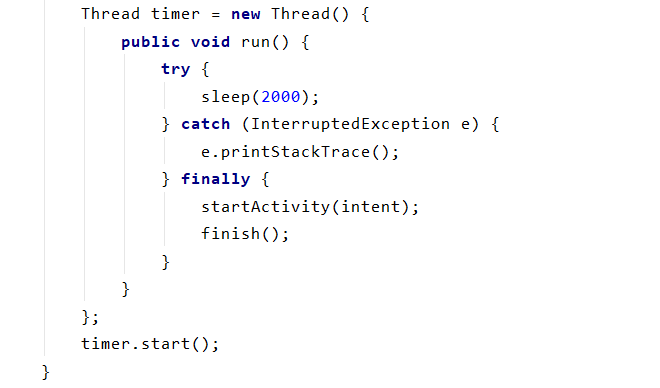


## 3.4 Software Design

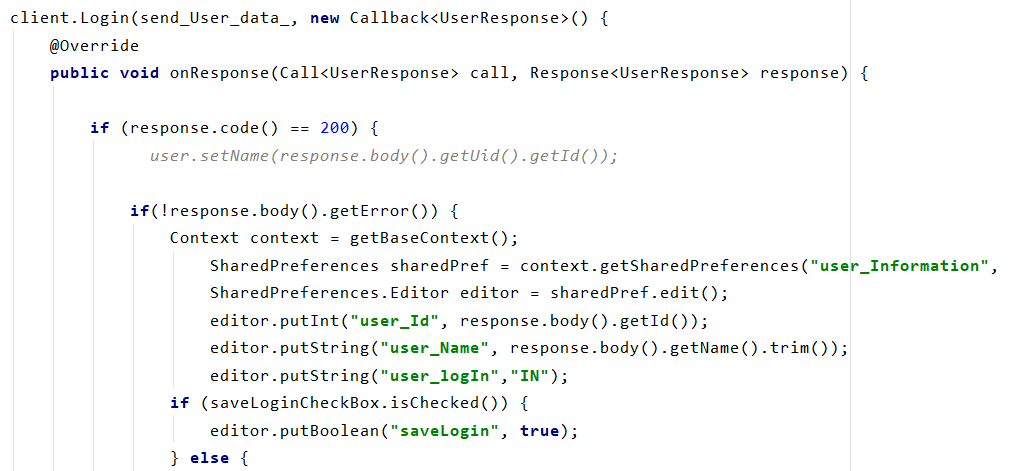
The mobile app has been developed by Android Studio v3.6.3.

## 3.5 User Interface and Design Code:

Welcome page or Splash Screen: This page appears once the app icon is hit, indicating the start of the application and the loading of all required libraries that the app will use later.

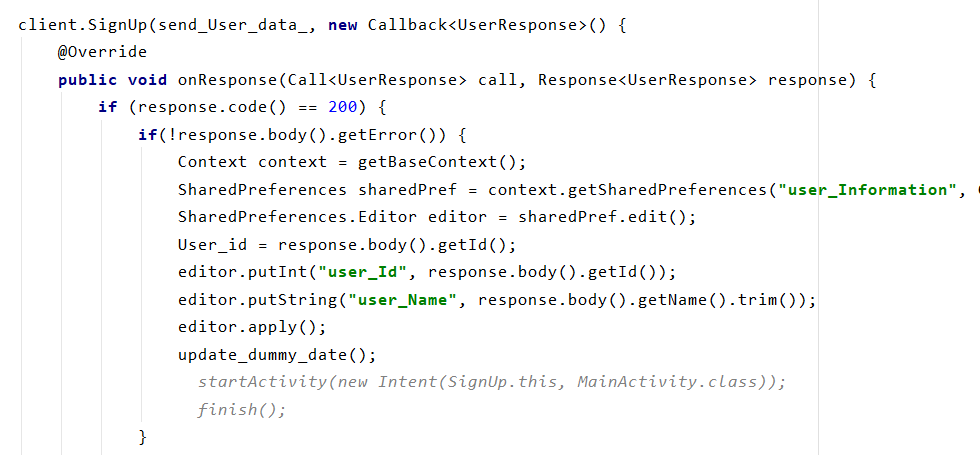


The timer is used in thread, sleep(2000) is the waiting time for the screen before starting new activity.

Login page: This page is essential to ensure that only the registered user is able to use and access both the system database and virtual sensors using web API.

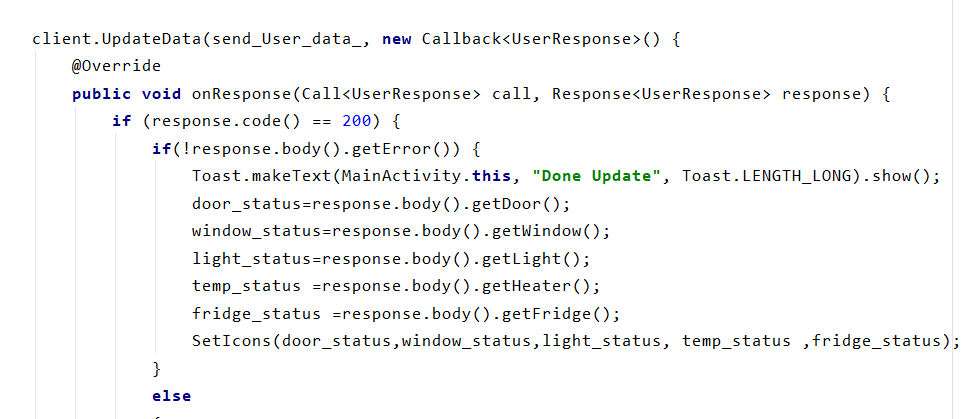
The OnResponse is retrofit library request with call back.

Registration page : This page to register a new user and able them to login



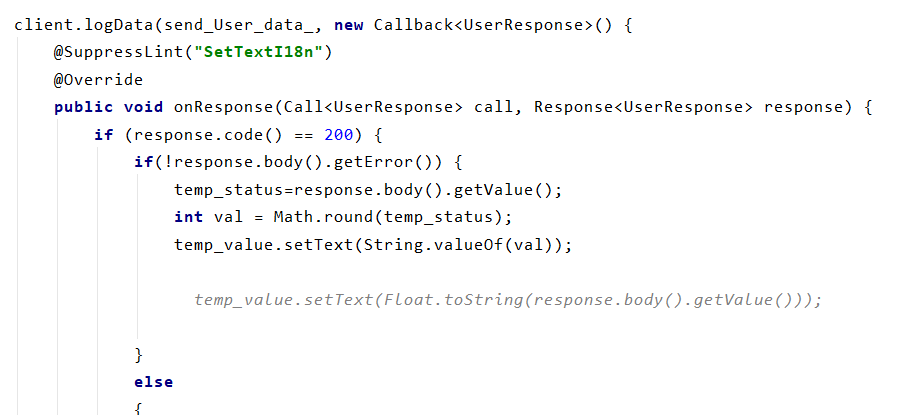
The signup method with retrofit library sends an HTTP request to my database with a callback Response from Database. The “send\_user\_data” is the field with send all the field using model pair.

Interaction page: This page indicates the current state for each object (door, window, light and fridge). For example, the main function for a door is opened or closed. As for light, the main function is on or off.



The “MainActivity” page contains the main work for our app. It contains of buttons with each button connected to the above code. I also refresh my data with a waiting thread with same concept which we used in our welcome or splash screen.

Review page: This page where the user can check the average Temps per day, Week, Month as well as can check how many per day fridge door was open.



This part of my code get the data form my Database and then calculate the data and takes the average for temperature plus number of fridge open per days etc.

## 3.6 Issues and Resolutions

There are a lot of issues we have faced and solved through some research on such errors, including the developing part and the designing part as well.

The following are some of the problems and the fix method applied for each:

Git version control errors in the IDE:

Operations requiring authentication in Git version control are broken in the IDE for Android Studio 3.6. To fix this issue, upgrade to Android Studio 3.6.3.

1. Could not start MySQL service because the default port is used by another software; one has to change the connecting port from the MySQL configuration manager manually and restart the service.
2. The resolution of the mobile app was not open in full screen mode when the utilised resolution is changed. The solution for this is to reconfigure the main xml file for the application and make it work as “portview” to enable the growth and shrink feature to fit any device with different resolutions.

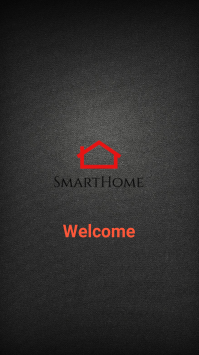
# Chapter 4: Implementation

## 4.1 Overview of Implementation

In this section, a simple demo will be carried on to test the functionality of the proposed system specially the mobile-app side. The process starts from the beginning in which the following steps should performed in the following order:

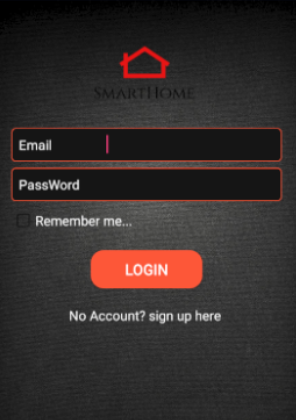
Starting Application will show the welcome page of our implemented app.

Figure 4.1 *App Welcome Page*



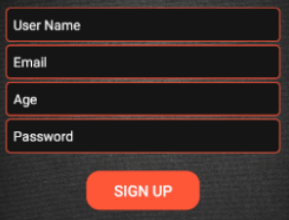
* After 2 second or 2000 milliseconds, the login page will appear as follow:

Figure 4.2 *App Login Page*

**

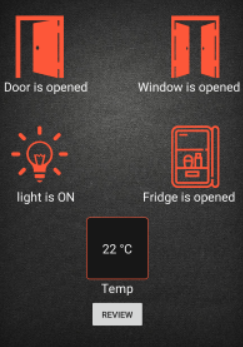
A new user has to register his/her information by clicking sign-up link. This will lead us to the registration page as shown in the following pic.

Figure 4.3 *App Sign-Up Page*

**

* A user has to provide username, email, age and finally, the password. After all data been entered, a sign-up button has to press by the user.
* A status page will appear which indicates all pre-deployed sensors which bounded to real-life objects like doors, lamps, windows, etc…

Figure 4.4 *App Status Page*

**

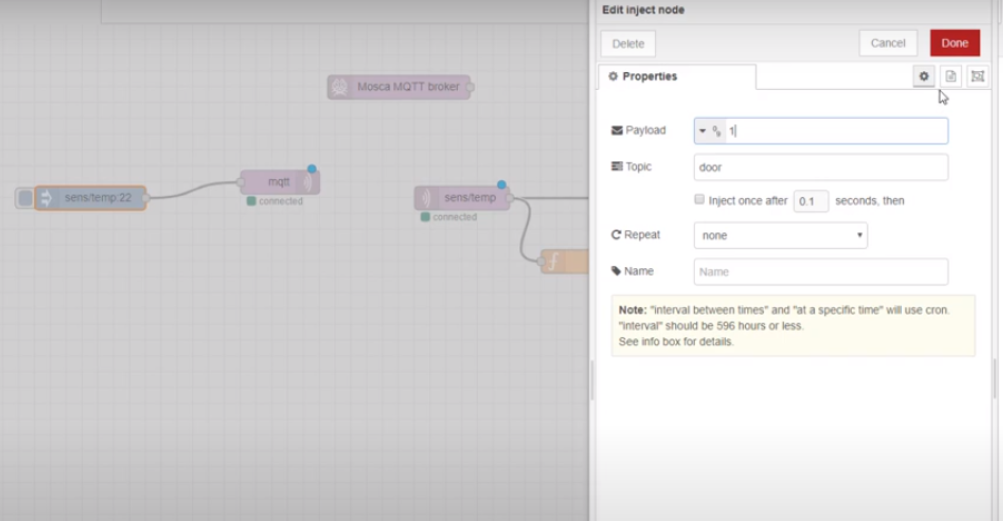
if user want to change the value of sensor through the app he/she can do that by clicking on the pictures which will be immediately change from the current status to the status user want and will be stored in database of this user.

*A screen shot of a computer

Description automatically generated*Figure 4.5 *review Page*

Review page where current user can check average of indoor temperature per day ,week, month or specific day current user want. As well as user can check number per day the fridge was open.

*A screenshot of a cell phone

Description automatically generated*Figure 4.5 *Node-Red Set Sensor Values*

The sensors values can be modified by using Node-Red.

Assume that we want to change the status for the sensor that is bounded for a door. One has to type the value which indicates the status of the door in the payload text box as shown in above figure after defining the target which will be assigned by such sensor. As well as we need to define the id of current user through the function node to inject this changes to the current login user database. Finally, if we check the status page on the mobile app for our current user , we will notice that the status is changed according to the modification we made to our sensor and the same holds for all other sensors and the temperature as well.

Figure 4.6A screenshot of a cell phone

Description automatically generated *Fridge Sensor Current Status*

This dashboard is part of node-red tool will monitor the value injected by sensor if the injecting value is 0 it is mean open so it will be reaching the max value and give alarm through the dashboard.

# Chapter 5: Test Plan

## 5.1 Test Plan

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Test Scenario** | **Expected Result** | **Actual Result** | **Pass/Fail** |
| 1 | Register User | User will fill form and then become a registered User | As Expected | Pass |
| 2 | Login User with valid details | User is logged in and transferred to the main page | As Expected | Pass |
| 3 | Check status of user Light | User can see the status of Light-whether it is on or off as well as can switch it to the desired status | As Expected | Pass |
| 4 | Check status of user Window | User can see the status of Window, whether open or closed, as well as switch it to the desired status | As Expected | pass |
| 5 | Check status of user Door | User can see the status of Door, whether open or closed as well as switch it to the desired status | As Expected | pass |
| 6 | Check status of user Temperature | User can see the degree of temperature; if equal to or more than 38, user can see an alarm as well as dialog appear if user wants to change the degree of temperature. | As Expected | pass |
| 7 | Check status of user Fridge | User can see the status of Fridge if it open or close as well as can switch it to the status the user wants. | As Expected | pass |
| 8 | Check average per day of user Temperature | User can see the average of temps per day | As Expected | pass |
| 9 | Check average per week of user Temperature | User can see the average of temps per week | As Expected | pass |
| 10 | Check average per month of user Temperature | User can see the average of temps per month | As Expected | pass |
| 11 | Check how many occasions per day fridge door was open | User can see how many occasions per day door of fridge was open | As Expected | pass |
| 12 | User can logout | User can logout | AS Expected | pass |

# Chapter 6: Conclusions

## 6.1 Final Conclusions

With the accessible monitoring platform of IOT management system there are great advantages and opportunities that would be beneficial for vulnerable and people with special needs. Thus, the IOT enables the seniors to live conveniently in their homes, by ensuring their safety and capability in controlling doors, windows, lights, temps, and Fridges in cost effective manner. Moreover, the health services centres can take advantage of such monitoring platform as this technology would provide agile control solutions to facilitate personal care and protection. IOT has already conquered our world; in almost every inch of our plant, there’s cameras or sensors that have been built on the technology of the Internet Of Things (IOT). These resources produce and process tons of data daily, and without a smart system to manage it effectively, most of that data will lack any potential. That is why we need IOT technology, as we have mentioned in the details of the project.

Physical devices get connected and controlled without human intervention, alternatively, through web-API infrastructure that manages this process digitally and centrally. In this way, the output will be produced faster since the machines communicate with high precision and accuracy. Node-RED is a tool for wiring APIs and hardware devices in a bright new method. It provides the user with an easy web-based editor that helps with the implementation. Finally, the database management system, MySQL, comes with the advantage of its powerful engine that can be applied on multiple different platforms without the need to change it to seek compatibility.

# References

* Brundha, S. M., Lakshmi, P., & Santhanalakshmi, S. (2017). Home automation in client-server approach with user notification along with efficient security alerting system. 2017 International Conference On Smart Technologies For Smart Nation (SmartTechCon). doi:10.1109/smarttechcon.2017.8358441
* Chayapathy, V., Anitha, G. S., & Sharath, B. (2017). IOT based home automation by using personal assistant. 2017 International Conference On Smart Technologies For Smart Nation (SmartTechCon). doi:10.1109/smarttechcon.2017.8358401
* Eclipse Mosquitto. (n.d.), from https://mosquitto.org/
* Team, T., 2020. Introducing The MQTT Protocol - MQTT Essentials: Part 1. [online]Hivemq.com.Availableat:<https://www.hivemq.com/blog/mqtt-essentials-part-1-introducing-mqtt/> .
* Kodali, R. K., & Soratkal, S. (2016). MQTT based home automation system using ESP8266. 2016 IEEE Region 10 Humanitarian Technology Conference (R10-HTC). doi:10.1109/r10-htc.2016.7906845
* Kumar, A., & Johari, S. (2015). Push notification as a business enhancement technique for e-commerce. 2015 Third International Conference on Image Information Processing (ICIIP), 450-454. doi:10.1109/iciip.2015.7414815
* Mandula, K., Parupalli, R., Murty, C., Magesh, E., & Lunagariya, R. (2015). Mobile based home automation using Internet of Things(IoT). 2015 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT). doi:10.1109/iccicct.2015.7475301
* NodeMcu -- An open-source firmware based on ESP8266 wifi-soc. (n.d.). Retrieved from http://www.nodemcu.com/index\_en.html

* Connect to an MQTT Broker : Node-RED. (n.d.). Retrieved from https://cookbook.nodered.org/mqtt/connect-to-broker
* Rajalakshmi, A., & Shahnasser, H. (2017). Internet of Things using Node-Red and alexa. 2017 17th International Symposium on Communications and Information Technologies (ISCIT). doi:10.1109/iscit.2017.8261194
* Mobile app. (2020, April 14). Retrieved from https://en.wikipedia.org/wiki/Mobile\_app
* New integration package for node-RED - Sentilo. (n.d.). Retrieved from http://www.sentilo.io/wordpress/new-integration-package-node-red/
* Global, A. (2020). Advantages of IoT: What to Expect in 2020. Retrieved 30 April 2020, from <https://aist.global/en/advantages-of-iot>
* The advantages and disadvantages of Internet Of Things (IoT). (2020). Retrieved 30 April 2020, from <https://www.linkedin.com/pulse/advantages-disadvantages-internet-things-iot-tommy-quek>
* node-red-contrib-nem2.(2020).Retrievedfrom <https://flows.nodered.org/node/node-red-contrib-nem2>