EJENS
European Journal of Engineering and Natural Sciences
e-ISSN : 2458-8156

Smart Scale IoT Application

Radosveta Sokullu[1]

[1] *Corresponding author: Ege University, Department of Electrical and Electronics Engineering, 35100, Bornova/Izmır, Turkey. radosveta.sokullu@ege.edu.tr*

Abstract

The use of electronic and communication devices in homes, factories, cars and many other sectors are increasing day by day, helping us to create a more environment friendly world. There is a growing demand to manage and analyze the collected data in a simple way. Many devices around us are turned into “smart devices”. This concept implies that the devices can collect data in a controlled manner, save it and transmit it to other devices either automatically or with minimal human control and minimal effect on the environment. The "Internet of Things" (IoT) has emerged as a new technology which allows devices to connect to a network through various protocols and communicate with each other over wide area networks. Thus the need to collect and analyze the data in a more manageable way has become an important research issue. This paper describes the work for creating a smart scale that can be incorporated in various healthcare, home and industrial IoT systems. The developed prototype system allows transferring weight data, provided from smart sensors, to a data center using the MQTT protocol via ESP32. The user will be able to access the information in the data center from his mobile device or personal computer and follow up and/or analyze the data. A user interface was developed for Android based devices using the Android Studio. The project ensures full integration between the MQTT applications and the Android Studio using the Eclipse-Paho library. The prototype system can find usage in various areas, from personal health, to environmental and industrial sectors. Results from the smart scale can be transmitted electronically, integrated in public healthcare systems, environment monitoring systems and industrial applications.

Key words

Internet of Things, smart devices, smart scale, MQTT

1. INTRODUCTION

The use and communication of electronic devices in homes, factories, cars and many sectors are increasing day by day. For this reason, there is lot of theoretical and application research going on currently both in the academia and the industrial world. The Internet of Things (IoT) has emerged as the new technology that allows to connect and operate various devices over long distances with minimal or no human intervention for the purpose of collecting and evaluating data and when required carry out actuation of specific operations. The IoT technology has penetrated in different areas of our lives, starting from smart homes, smart appliances, smart traffic, and smart agriculture to smart vital signs collecting systems and industrial applications. Data collected over the IoT can be used in many diverse ways – from personal look-up applications, to vast data analytics systems. The work presented here addresses the issue of collecting weight measurement data and transmitting it over the IoT to be used for personal record tracking. The paper describes the design and prototype of a smart food weighing system created using the internet based technologies. This so called “Smart Scale” will transfer weight sensor data via ESP32 to a data center using the MQTT protocol. The user will be able to access the information in the data center from his mobile

device or personal computer and analyze the data. In this way, he will be able to determine the daily nutritional requirements and also remotely monitor the amount of food remaining in stock. From here on the paper is organized as follows: in the following section a short overview of some related work is presented. In section 3 the proposed system architecture and main components are described followed by user interface design, evaluation and conclusion.

2. RELATED WORK

An interesting study utilizing a smart weighing system is presented in [1] where the authors use a wireless weighing platform, comprising the latest advances in low power wireless sensor network technology, to better understand the condition of the colony in beehives. One of the main measures of the strength of a beehive is the weight of the colony. Changes in weight can accurately reflect the productivity of the colony. The installed system is able to scale the measurement linearly and has been found successful in terms of energy consumption. A farming related smart scale system is presented in [2]. The authors consider the case of collecting rubber from various small farmers in Malaysia. In the traditional practices, each farmer brings his load, which is weighted and recorded in a logbook manually. However, this practice is very prone to human errors and in many cases physical records are lost. So the authors propose the so called “Smart Rubber Scale” (SRS) System which utilizes cloud storage to ensure long term preservation and accessibility of the data. The SRS prototype consist of load cell sensor, scale sensor, Arduino UNO R3, sensor module HX711, Bluetooth module and an LCD display. A dedicated SRS website is developed by using PHP that allows the staff to automatically store rubber weight information automatically after measurements are taken. Both the farmer and the staff have access to the uploaded data and also detailed monthly records can be downloaded. Another system which presents an example of a smart IoT based personal system is presented in [3]. It describes the design of an IoT application aimed at storing personal vital signs information in a cloud. It provides a possibility to access and evaluate the data via a mobile application. In this process, the data collected from the patient with an embedded sensor device installed with the STM32 card and is then transmitted to a predefined database over the internet. An MQTT based cloud system is used. The saved data can be accessed and processed individually using a custom designed Android based application. As a result, a system with very low power consumption and low latency is designed. More details on how security issues in this type of applications can be handled are to be found in [4]. The work discusses the secure communication model for embedded devices over the Internet of Things (IoT). The goal is to examine innovations in the application layer together with the widely used MQTT protocol to provide secure communication. The basic idea is that the MQTT protocol enrolls using a username and password. In order to prevent this information from being captured by third parties during communication, a modified application logic and unnecessary encryption method is used. With this method, the risk could be partially compensated. In [5] the authors present a remote controlled dog feeding mechanism with changeable and adaptable characteristics. The proposed prototype system allows dog owners to take care of their dogs without interrupting their daily routines. At the same time it provides a good record of the feeding practices over a given period of time by determining the stock information, feeding frequency and consumption. The prototype uses a load cell for remote weight measurement, an Android application and a RFID for realizing control operations. When the feeding process is completed a message is sent remotely providing information about the time and details of the feeding. An example of a system using the ESP32 module for wirelessly transmitting locally collected data is presented in [6]. The authors’ aim is to create an application related to smart grid technology. The proposed solution allows transmitting and controlling electrical energy consumption by using a smart meter. Thus, the service provider and the consumer are able to obtain information about energy consumption at certain time intervals. The data measured locally is broadcast to an MQTT server with the help of a ESP32 module. Users have the possibility to access their personal recorded information whenever they want via the web over this MQTT server. This is an attractive application, which deploys an IoT network and makes it quite easy to keep track of our electricity consumption intelligently.

3. PROPOSED SYSTEM ARCHITECTURE AND MAJOT COMPONENTS

3.1. System Architecture

The proposed prototype comprises an IoT smart scale which can collect weight data from a smart load cell and transfer it through an wireless unit to a MQTT server. The functional diagram of the developed prototype is presented in Fig.1 below.

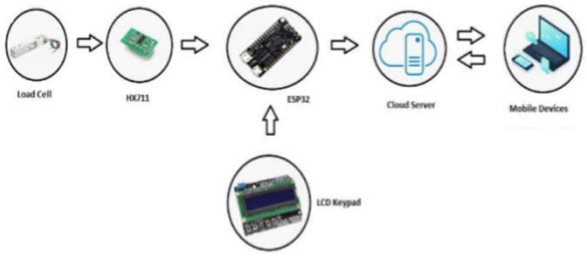

Figure 1. System Architecture

As can be seen from the figure the architecture of the proposed system contains a load cell, an HX711, a wireless transmission module ESP32 for collecting and transmitting the weight data; a cloud server to store the information and an Android based user application which allows access and evaluation of the recorded data. An LCD screen to provide user input when required.

3.2. Major Components

In this section the main components of the proposed system are discussed.

3.2.1. Smart Weighing Module

The smart weighing module is based on a load cell with HX711 shown in Figure 2. Measurement of the weight is done using load cell sensors, which convert the force acting on the load cell into a signal. This signal can be a change in voltage, current or frequency depending on the type of load cell and circuit. Thanks to the strain-gauge structure inside the two 5 kg load cells used in this work, an analog signal is generated by the difference in resistance caused by the load on it. This signal is transferred to the HX711, which is on a 24-bit ADC module. The E + and E- inputs of the load cells are connected to the E + and E- input of the HX711 by mapping. The red ends of the mutual load cells are connected to the A + and A- inputs of the HX711, providing the combined use of load cells.

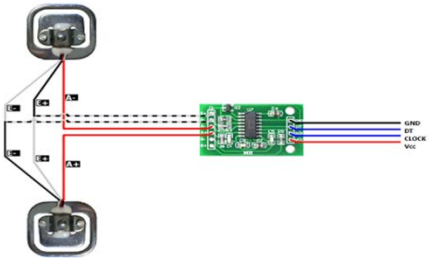

Figure 2. Load Cell with HX711 connection

3.2.2. The ESP32 Module

ESP32 is a microcontroller with a 240 MHZ clock frequency, a programmable WiFi receiver/transmitter with embedded power amplifier, filter and antenna, and is frequently used in IoT applications. It contains peripherals such as I2C, SPI, UART, and can be programmed with Arduino or MicroPython IDEs. ESP32 can be operated over protocols such as TCP, HTTP, MQTT, SNTP. In this prototype, the data from the HX711 is transferred digitally from the data end to the ESP32 with a specific clock frequency. The data is processed by using the "HX711.h" library in ESP32. Thus, the weight data measured from the load cell module is ready to be transferred to the server for use. After the user selects the product, the product information from the LCD keypad is transferred to the MQTT server along with the weight data. The ESP32 is defined to the MQTT server as a publisher. The ESP32 is connected to a WiFi network using the predetermined SSID and password information. Open source Cloud MQTT was chosen as the MQTT server.

3.2.3. The MQTT Module

MQTT is a machine-to-machine (M2M) protocol, one of the most widely used protocols for IoT. It provides a messaging network based on the publish/subscribe principle. The protocol works over TCP/IP, but any lossless, double-sided network protocol will work. It operates with limited data rate and is especially suited for IoT applications, which require low data rate and small data volumes. There are 2 main entities in the MQTT protocol: a MQTT server and a certain number of MQTT clients connected to it. The MQTT server receives messages from all clients and directs them to the addresses to be reached (clients). An MQTT device connects the server to the network and compiles the MQTT library. The MQTT packet consists of a header and message part. The messages are listed under headings. If a title appears on the server and there is no subscriber under this title, the server deletes it. "Publisher" is an MQTT client that sends a message to the server. "Subscriber" is an MQTT client that can view or use the messages under the header by registering for the title sent by the publisher. Three types of MQTT messages are used: Connect, Disconnect and Publish. The connect message is used for establishing the connection between the server and the units; the disconnect message causes the disconnection; publish message is the MQTT client request message. As shown in Figure 4, subscriber devices can reach the server and at the same time publish messages on the server as a "publisher", register in the headers and send messages under the header. MQTT applications use a microcontroller as a "publisher" to save the data it receives.

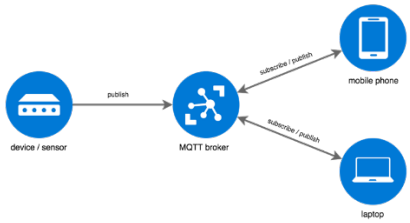

Figure 3. MQTT Protocol Operation

ESP32 is one of the most used and most suitable microcontrollers for this purpose. Devices connected to the server as "subscriber" are devices that provide the data to the user with applications installed on operating systems such as Android, IOS, Windows, usually from sensors. In the project created, a solution was produced for Android devices using Android Studio. The use of Eclipse-Paho library, where MQTT applications are executed for Android Studio, and its integration into the project has been provided.

3.3. System software

Eclipse-Paho library is used for the MQTT protocol in Android programming. A publisher/server application is developed with Java in Eclipse-Paho library. Each interface page for Android

consists of activities. Abstract operations are done with the help a similar interface. Interfaces define functions in a way that we can "override". Helper Java classes can also be created for use in activities that execute the interface. For example, MainActivity, AfterLogInActivity, ProgressActivity used in this prototype are examples of activities; AppDatabase, MyAdapter, MqttHelper are examples of Java helper classes. The Android programming was carried out in Java. The interface that welcomes the user when the application is opened is the interface where the user will register, i.e. MainActivity. The classes User, User Alme and AppDatabase, which are connected to the MainActivity, are the classes that perform the operations on the local database to store the information that the user has registered. The User class determines the user's variables, such as email, password, name and surname that the user must save in the system. AppDatabase is the class in which the connection configurations of the database are made. Android Room Database is used as the database.

The main algorithm is given in Figure 4 below. After the user registers with their profile information, they are directed to ProressActivity. ProgressActivity is an activity that offers the user a welcome screen with their first and last name. In ProgressActivity, the name and surname information given by the user while registering to the system is taken from the Room Database (a local database in Android). Following the input of the user's name and surname, a special welcome screen is activated for the user followed by direction to the AfterLogInActivity. MQTT messages are displayed sequentially in AfterLogInActivity. There is a RecyclerView library in this class that allows the functions and messages that receive MQTT messages to be displayed sequentially. The class that allows RecyclerView to connect to the activity is the MyAdapter class. In addition, each message is set up with CardView to print a cloud icon and message. The class in which MQTT functions are defined is the MqttHelper class. The MqttHelper allows integration of MQTT operations and defines the actions to be taken when the message is received.

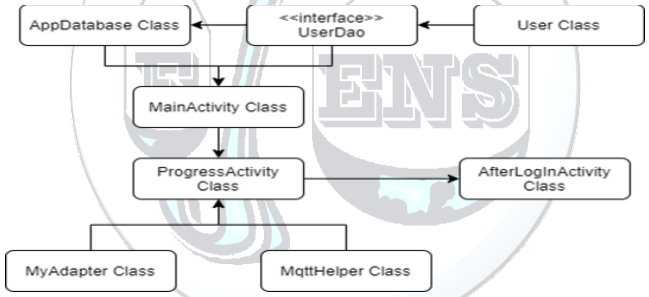

Figure 5. The main algorithm

4. USER INTERFACE DESIGN

An important part of the design is the user interface. The first screen on the LCD that meets the user is given in Figure 5 below and shows the message "Welcome, Press a Key".

Figure 5. The Welcome LCD screen

After connecting to the server, the system offers the user the opportunity to choose a product. While the user selects the desired product by pressing the keys, the scale measures the weight of the product. Each time the user presses a key, the weight data from the scale is updated. Thus, when there is a new user, or the user wants to change the weight, he will be able to weigh the product again and save it to the system with new data. An example regarding the product information and the sensitivity of the scale is shown in Figure.6.

Figure 6. An Example of Weighing a Milk Carton

After the product selection and weighing process is completed, the information on the screen is sent to the MQTT server as soon as the user presses the "Select" button. Thus, it is connected to the MQTT server as "Publisher" with ESP32. When the user connects to the server as a "Subscriber" from any Android device, they can view the messages under the "topic" on their device. The mobile device software was developed using Android studio. Examples of the design can be seen in Figure 7 below.

5. PRELIMINARY RESULTS AND DISCUSSION

The work described in this paper is part of an ongoing project for creating a user-friendly home environment. The described prototype has been tested as a standalone system and the results are presented below. Experiments regarding the input interface and weighing sensitivity of the scale have been developed. Two groups of tests were carried out: the first was testing the connectivity of the system and the proper operation of software. Examples of the user screens at the stage where the user is greeted on the Android device and the profile information is saved to the system are given in Figure 7 below. The second group of tests regarded the measuring systems. The load cell-based system was calibrated and tested with various samples of goods. For a 1.5 lt water bottle it was observed that the weight measurement was had a deviation of 6.25%; in the tests performed with 1 kg tea package, weight measurements were obtained without error; tests performed with a 300 gram snack package showed a deviation of 3.44%. The average error was estimated at below 2.6%. Compared to other similar systems like the ones presented in references [1], [2] and [5] these results are acceptable. The system has met the requirements and smoothly operates over the provided wireless environment. Privacy and protection of the data is secured by the login requiring registration, username and password. Since the goal was to create a simple, low cost prototype using off-the-shelf elements no further protection mechanisms were incorporated.

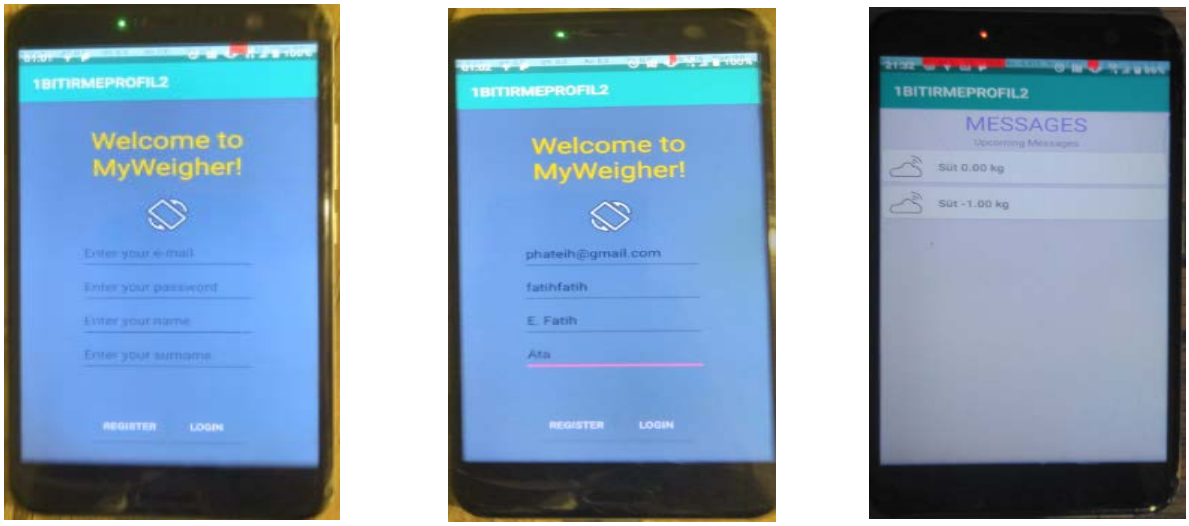

Figure 7. Examples of user interface screens

6. CONCLUSION

In this paper we have discussed the proposed prototype and main components for an IoT based smart scale application. The user has been provided with easy access to food weight information. ESP32 was chosen as the microcontroller, and the weight data received from the load-cells were stored on Cloud MQTT servers using the MQTT protocol along with the product information from the LCD keypad. The data on this server can be accessed through the application developed for Android devices. The designed prototype had in mind a personal food tracking information system, as part of a smart environment project, but with small modifications it can be used in other situations including also industrial weighing applications

ACKNOWLEDGMENTS

The authors would like to thank Eyüp Fatih Ersoy and Ata Durak for their contributions.

REFERENCES

[1]. D. W. Fitzgerald, F. E. Murphy, W. M.D. Wright, P. M. Whelan, E. M. Popovici, “Design and Development of a Smart Weighing Scale for Beehive Monitoring”, 26th Irish Signals and Systems Conference, ISSC 2015, 21 July 2015 DOI: 10.1109/ISSC.2015.7163763
[2]. N.S.N. Ismail, N; S.Z.B. Mustafa, F. Yunus, N.B.A.Warif, Internet of Things Smart Rubber Scale System Using Arduino Platform In Proc.: 2020 IEEE International Conference on Automatic Control and Intelligent Systems (I2CACIS) pp.45-50, Shah Alam, Malaysia, June 2020
[3]. D. Yi, F. Binwen, K. Xiaoming, M. Qianqian,“Design and Implementation of Mobile Health Monitoring System based on MQTT Protocol”, IEEE Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC), pp.1679-1682 Harbin, Oct, 2016
[4]. P. Peniak, M. Franeková, “Extended Model of Secure Communication for Embedded Systems with IoT and MQTT”, International Conference on Applied Electronics, 19 October 2018, DOI: 10.23919/AE.2018.8501434, 2018
[5]. V. K. Karyono, H. T. Nugroho, “Smart Dog Feeder Design Using Wireless Communication, MQTT and Android Client”, In Proc. 2016 International Conference on Computer, Control, Informatics and its Applications: Recent Progress in Computer, Control, and Informatics for Data Science, IC3INA 2016, , pp191-196, February 2017
[6]. R. K. Kodali, S. Sahu, “MQTT Based Smart Metering”, Proceedings of the 2nd International Conference on Green Computing and Internet of Things, ICGCIoT 2018, August 2018, pp.399-402. DOI: 10.1109/ICGCIoT.2018.8753052
[7]. MQTT website. [Omline]. Available: http://mqtt.org/