Design and Implementation of Car Parking Monitoring Come Management System Powered by IoT

Avishek Gupta[1], Sohini Banerjee[2] and Soumyendu Bhattacharjee[3]

[1,2] Assistant Professor, Department of Computer Science and Engineering, AIEM, Mogra

[3] Assistant Professor, Department of Electronics and Communication Engineering, AIEM, Mogra

[1]*avig2005@gmail.com*, [2]*banerjeesohini88@gmail.com*, [3]s_vlsi@yahoo.in

Abstract

Internet of Things (IoT) based technology have greatly increased and improved public necessities. IoT in a parking lot will make it possible for drivers to check their cellphones to see if a parking spot is available. This research work is designed to address the common daily parking issues that many individuals encounter. Some difficulties have gotten worse as our population has grown steadily. One of them is parking, which is a difficulty because there are so many cars on the road but not enough parking spaces. The IR sensors, servers, and cloud were used to build this system. To identify the presence of an automobile, controllers and sensors will be installed on the ceiling of each parking space. Installing a server and three sensor circuits in a parking lot allows for system testing. The tests involve timing the data transfer process and gauging how successfully data is transmitted from the parking lot to the cloud. The IR sensor circuit and FRID can communicate the parking lot data without error, according to the results of the tests mentioned above. Our Proposed design can update parking lot data in little more than one minute.

Keywords: *Internet of things, IoT cloud, IR Sensor, Car Parking Management System (CPMS), Radio Frequency Identification (RFID)*

1. Introduction

The majority of currently used parking lots lack an organized framework. The majority of them are somewhat inefficiently managed by hand. The issue that constantly arises at the parking lot is time lost when looking for open parking spots. Users will continue to circle the parking lot until they find a free spot. This issue typically arises in urban settings when the number of automobiles exceeds the number of parking spaces. These inefficient conditions developed as a result of the market's current technologies' lack of application [1]. Many neighborhood parking lots are now constructed inside retail centers or other multifunctional structures to accommodate automobile owners. This kind of parking lot typically has a parking guidance system that is focused mostly on the usage of message signs to notify drivers about parking availability inside the parking lot. The sensors that count the number of automobiles entering and exiting the parking garage often provide this information, but it can also be determined by comparing the fines issued by the machines. The availability of parking spaces inside a parking garage is typically shown on a display board at the garage's entrance by the words "full" or "empty." Rarely is the real number of parking spaces available inside the parking lot disclosed. Hundreds of vehicles enter the parking lot each day in search of available spaces to park. Therefore, it is crucial to have a reliable empty parking slot tracking system that can show drivers where the empty spaces are at each row of parking slots and direct them there. The creation of this project's prototype can serve as a way-finder to direct automobile drivers to parking spaces that are available inside the car park and direct them to go there. IoT is a concept that enables communication between various items (smart homes, cars, phones, and household appliances) by connecting them. Electronic parts, sensors, actuators, and network connectivity are all included in the IoT device [2]. With these parts, IoT devices can use the architecture of the internet network to send data or even be remotely controlled. A new idea known as "Smart City" is introduced when the IoT concept is applied to urban life. Although there is not a definite definition of a "smart city" just yet, its precise objective is to develop services that make it easier to access public facilities and raise their standards [3]. Urban IoT, or the application of IoT technology in urban infrastructure for public utilities, can achieve this goal [4]. Anyway, a smart parking system that can link to the cloud is required. Many users of vehicles can benefit from this parking system by finding parking spaces more quickly. This parking system's data can be supplied to mobile devices or other programmes. Local governments may keep an eye on abandoned parking lots remotely and in real time.

2. A Brief Review

Public parking for four-wheeled cars is an illustration of an essential public facility in metropolitan settings. In the past, several technologies have been developed for parking systems, and some of them have been applied in shopping malls. As an illustration, consider the idea of a smart parking lot that incorporates sensors, actuators, and microcontrollers. Users simply need to park the car in a specific spot; after that, the actuator will be told by the microcontroller to move the car to an available parking space [1]. This idea is highly intriguing, but it will cost a lot to implement because it requires a complete physical makeover of any current public parking lot. Another smart parking lot combining sensors, microcontrollers, and Light-Emitting Diode (LED) displays has also been proposed [2]. The sensor will gather information about how many parking spaces are available and transmit it to the microcontroller. The microcontroller will then show the number of open parking spaces on the LED display [3]. The IoT concept can be applied in this area to increase the community's access to parking lots [4]. To learn how much parking is available and where the lot is empty, people do not need to travel there. A smartphone application linked to the cloud can be used to determine which parking spots are available. To make it simpler to collect data that can be used for analysis, Radio Frequency Identification (RFID) can be placed to the parking lot [5]. One of its more effective strategies is The Web of Data, which has made a number of data sources compliant with

Semantic Web technologies, including RDF triple stores and SPARQL endpoints, available by adhering to the principles of Linked Data. On the other hand, these architectural principles have been conceptualized as the Representational State Transfer (REST) architectural style, which underpins the human-readable web[6-9] . One of its more effective strategies is The Web of Data, which has made a number of data sources compliant with Semantic Web technologies, including RDF triple stores and SPARQL endpoints, available by adhering to the principles of linked data. On the other hand, these architectural principles have been conceptualized as the Representational State Transfer (REST) architectural style, which underpins the human-readable Web [10-12]. In a particular research work, we concentrate on creating a customizable access control system for APIs. This is a crucial security feature that ensures the application of permission restrictions on resources when calling their API services [13]. For the formulation and execution of Attribute-Based Access Control (ABAC) policies, we have created an extension of the popular (open source) Spring framework's security framework, known as the Spring Security framework. We use situations that could occur in a smart energy eco-system to illustrate our approach [14]. An analysis of the existing TLS threat surface is given in a research paper along with a proven method for reducing the risk of TLS-compromise. In light of recent TLS vulnerabilities, this paper's key contributions are - discovery of configuration options that collectively maximize security assurances and specification of expected flows and automated comparison with observed flows to highlight inconsistencies [15-17]. A leap from mere quantity to a new quality, frequently referred to as the Internet of Things, is now possible thanks to ubiquitous IP technology, despite the fact that many restricted, special-purpose networks have been (and will be) constructed for this [18]. This might result in trillions of embedded devices during the following ten years, dramatically expanding the scale and reach of the Internet. Three classes of end-devices that are designed to satisfy various needs for certain applications can be implemented. Classes A, B, and C exist. Although the implementations of these classes vary, they always provide two-way communication between end devices and gateways [19-20]. For instance, class A allows for two listening/receiving instances after each transmission, whose scheduling is totally up to the demands of the end device. With the Net-Server's consent, a separate sub-band is designated for the second listening window. For devices that must connect to the server in order to get information, this class offers power-saving choices. This implies that the server can only get in touch with the end device when it has done so. In addition to the functionalities that are required for Class A end-devices, Class B end-devices can also arrange additional receive windows. The endpoint's scheduling of receive windows necessitates synchronizing the device's clock with the servers [21-22]. One of the earliest technologies to stand the test of time is radio frequency identification (RFID). The supply chain has traditionally used RFID technology for item tracking and identification, but it is being used in other areas, such as indoor positioning to support location-based services. Active RFID tags have their own power supply, which enables them to emit a powerful signal that enables readings from a distance, in contrast to passive RFID tags, which are powered by the electromagnetic signal that is broadcast by the transmitter [22-24]. Active RFID tags are able to handle functions that require more computation and memory than passive RFID tags. Due to their capabilities and versatility, active RFID tags can be used in monitoring systems for data gathering and/or actuation. The de facto standard for outdoor positioning is the Global Positioning System (GPS), however it has several drawbacks in areas with tall buildings, within structures, or in situations where signal penetration is weak. Furthermore, basic sensor nodes may not be able to meet GPS systems' higher energy requirements due to the potential reduction in battery life [25-26]. A complete solution for both interior and outdoor positioning would be provided by combining the two technologies, although using active RFID for positioning in locations where GPS is impractical to utilize could close the gap. An RFID tag is attached to the object whose location is to be determined in a mobile tag, whereas RFID readers are often stationary in a mobile reader.

Types of Car Parking

This parking system concept is built around three primary processes:
(i) Payment method when the car is taken entry.
(ii) Finding suitable parking space.
(iii) Going out from a particular space.
The first step in the procedure is paying at the gate, where an RFID reader and controller with a liquid crystal display (LCD) will be installed. A RFID card with an e-money feature is required for car drivers. The gate will open after the car driver simply taps the card on the scanner, sending data from the card to the local server and cloud [27]. This chapter will not produce a parking gate bar; instead, it will merely create a series of data readings and transmissions. However, because relays are supplied in the circuit, the microcontroller and RFID reader can be connected to the parking gate bar if necessary. The procedure for securing a parking space is the second process [28]. Car detection sensors will be installed in each parking space as well as extra sensors on the parking marker to maintain order in the parking lot. For the purpose of mapping out the parking lot, each space has a distinct address. The system will update the most recent data in the cloud whenever an automobile is found at a particular slot address. The process of leaving the parking space is the third process. The second technique and this one is practically identical. The system will update the current data in the cloud in accordance with the slot address under changing circumstances if the sensor detects the car exiting a particular parking space.

Methodologies Used for General Design of CPMS

The established parking system is a tool for counting and identifying the presence of vehicles in a parking space. The parking lot data will be uploaded to the cloud database. The information will be used by an Android application to direct people to the appropriate parking place. The parking system's mechanism is depicted in Figure 1.

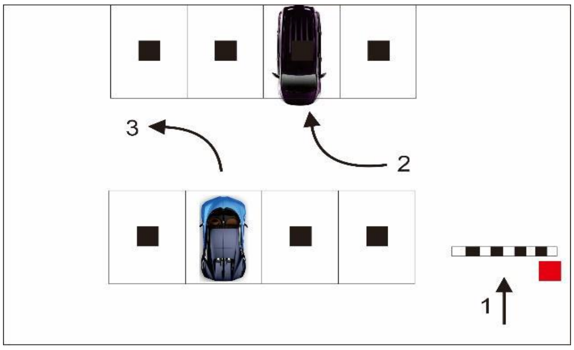

Figure 1 - Mechanism of parking system.

The presence of the car is detected using both sensors. The current parking limit on the side is measured using a proximity sensor. The controller will analyse the sensor's output value if a car is parked somewhere other than its designated spot. The buzzer will then switch on to alert the driver to the incorrect parking situation.

Novelty of This Research Work

The creation of a prototype for an internet-based smart parking system demonstrates how well it can detect, process, and transfer parking information to the cloud. The important findings from the testing data include the ability of IR sensor and RFID circuits to transfer data to Ubidot's IoT cloud without any data errors in less than twelve seconds. The second is that the RFID circuit can read and write data back to the card in less than two seconds without experiencing any data errors. The third is that data can be processed by sensor, RFID, and server circuits in under two seconds. The last is that delivering data to the cloud with optimised time takes longer than transferring data from a circuit to a server.

Our Proposed Design

The entire design is illustrated in detail into two parts.

a) Circuit Level Design

The system's controllers are Arduino Pro Mini boards, and its modules include an ESP8266, two HC-SR04, a 16 by 2 LCD, and a proximity sensor. Transistor, 3.3-volt Zener diode, LED, buzzer, and resistor are additional supporting parts. We have also used ultrasonic sensor. The presence of the car is detected using both sensors. The current parking limit on the side is measured using a proximity sensor. The controller will analyse the sensor's output value if a car is parked somewhere other than its designated spot. The buzzer will then switch on to alert the driver to the incorrect parking situation.

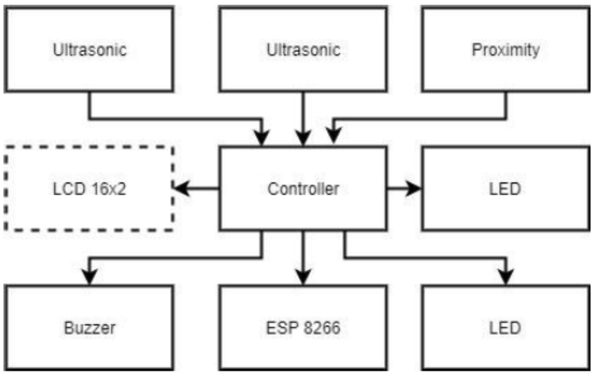

Figure 2 – Block diagram of Sensor based part of the design

Two LEDs are only utilised to visually communicate information about the car's presence. One LED will turn on if there is no car in the parking space, and another LED will turn on if the space is occupied. A visual information device that reads data from the three sensors is LCD as well.

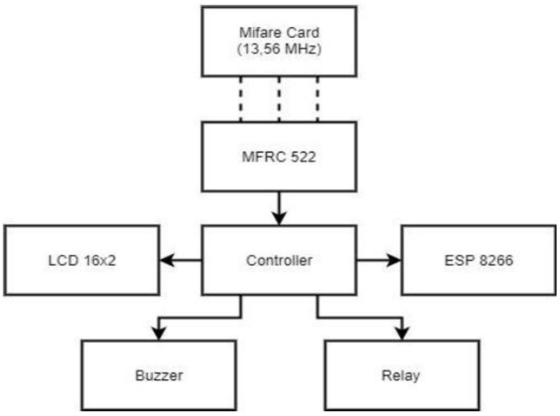

Figure 3 – Block diagram of RFID based part of the design

Due to the location of the circuit in the parking area's ceiling, LCD is not always required to be placed. A data transmitter medium from the microcontroller to the local server is provided by the Wi-Fi Module. Similarly, the designing part related to the RFID is shown above in the figure 3. The RFID reader utilised in this system is an MFRC521, and the only RFID card that may be used is one that is compatible with the 13.55 MHz protocol. Local servers and Arduino servers communicate via the ESP-826.

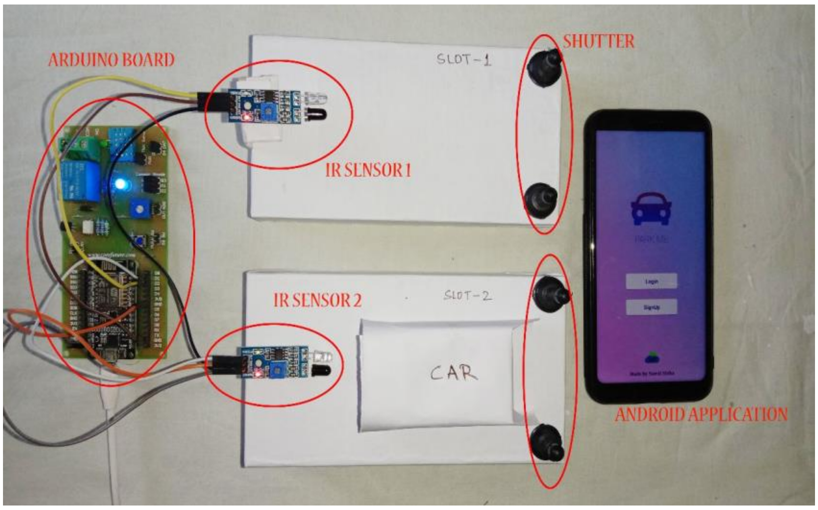

Figure 4 – Circuit diagram of the proposed design

An electrical device that monitors and detects infrared radiation in its environment is called an infrared (IR) sensor. Despite being on the same electromagnetic spectrum as visible light, IR has a longer wavelength than visible light, making it invisible to the human eye. The above design is based on IR Sensors because of its so many advantages. Due to the line-of-sight or point-to-point communication mode, it offers secure communication. Infrared devices always consume less power, their batteries last for a long time. Motion is successfully detected by infrared motion sensors both during the day and at night.

b) Software Level Design

The system starts by initialising variables, I/O ports, and used libraries, as shown in Figure 5 below. The sensor circuit will by default believe the parking space is empty. After everything has been initialised, the Arduino will calibrate the sensor by repeatedly sampling the sensor readout data.

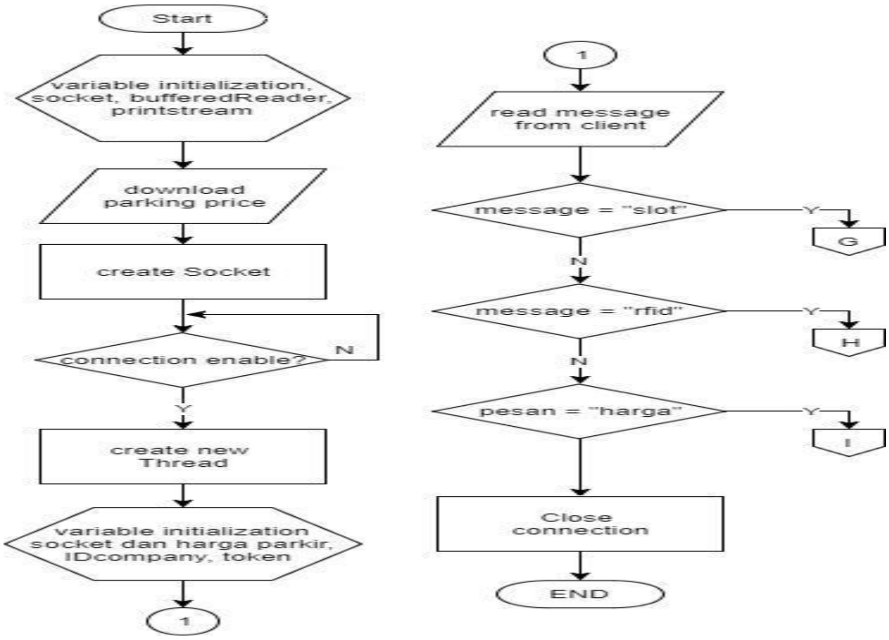

Figure 5 – Flow Chart of Designing part of Server

When an automobile is not there, the average reading will show a typical distance. The Arduino has been connected to the network via ESP8266 once calibration is finished. The Arduino is prepared to perform sensor readings once it is networked. Using both ultrasonic sensors, Arduino will first determine whether the car actually exists. The presence status of the car will be changed to one if both sensors detect a difference in distance of about 1.4 metres, indicating the presence of a car in that block. However, if there isn't an automobile, its presence status will be set to zero. The Arduino will compare the state of the preceding automobile with the status of the current car if parking status has been established. Arduino will connect to the local server and provide the information for the parking slot if there was previously no car and there is now a car, or otherwise. However, Arduino won't connect to the server if the current and previous conditions are the same. The next step is to inspect the parking lot for cleanliness. To identify messy parking, the circuit will measure the proximity sensor's distance. The buzzer will sound only if the car is found; otherwise, it will be shut off. The programme will then hesitate for one second before again verifying the existence of the car.

System Level Testing and Analysis

a) Testing of Sensor Circuit

By detecting the presence of the car ten times while assuming a 150 cm car height and a 350 cm floor to ceiling distance, the entire circuit's data transfer to and from the cloud is tested. The circuit's ability to convey data successfully, the amount of time needed, and the accuracy of the data are the test's benchmarks. The programme records the difference between the program's start and end times to calculate the reply time on the Arduino and the server.

Test	RFID 1	Response Time			Total Time (s)	Data Error
		Arduino (ms)	Server (ms)	Ubi dots (s)		
1	Pass	1 396	751	7	9.147	0
2	Pass	925	362	8	9.287	0
3	Pass	1 406	776	7	9.182	0
4	Pass	1 303	550	7	8.853	0
5	Pass	2 156	500	2	4.656	0
6	Pass	1 488	603	7	9.091	0
7	Pass	1 714	622	8	10.333 6	0
8	Pass	1 214	161 4	6	8.828	0
9	Pass	1 062	121 1	3	5.273	0
10	Pass	1 302	546	6	7.848	0
Avg	100 %	1 396.6	753.5	6.1	8.250 1	0 %

Table 1: Testing Result for IR Sensor Circuit

By figuring out the difference between the server sent time and the Ubi-dots received time, the response time on the cloud is calculated. The sensor circuit has a 100% success rate for delivering data to the cloud, with an average total time required of 6.4004 s for each data, according to the testing results presented in Table 1 below. The test did not uncover any data mistakes.

b) Testing of RFID Circuit

RFID cards are tapped ten times to test RFID circuit communications to the cloud. The programme already has techniques for determining the react time, including testing the sensor circuit. The circuit's ability to send data successfully, the amount of time it takes to send data to the cloud, and the consistency of the data sent and received are the key performance indicators.

Test	RFID 1	Response Time			Total Time (s)	Data Error
		Arduino (ms)	Server (ms)	Ubi dots (s)		
1	Pass	1 396	751	7	9.147	0
2	Pass	925	362	8	9.287	0
3	Pass	1 406	776	7	9.182	0
4	Pass	1 303	550	7	8.853	0
5	Pass	2 156	500	2	4.656	0
6	Pass	1 488	603	7	9.091	0
7	Pass	1 714	622	8	10.333 6	0
8	Pass	1 214	161 4	6	8.828	0
9	Pass	1 062	121 1	3	5.273	0
10	Pass	1 302	546	6	7.848	0
Avg	100 %	1 396.6	753.5	6.1	8.250 1	0 %

Table 2: Testing Result for RFID

RFID circuits have a 100% success rate and a 0% transmitted data error rate while transmitting data to the Ubidots cloud. The average amount of time from the moment the card is detected till Ubidots receives the data is 8.2501 s. The average processing time in Arduino is 1 396.6 ms, and the server processing time is 749.5 ms. The average amount of time taken to send data from the server to Ubidots is 6.2 seconds.

c) **Detection of Car Position**

By using a car detection test and the premise that the car has a height of 150 cm, the accuracy of sensor readings was evaluated. For each sensor circuit, testing is repeated up to ten times by moving obstacles. The barriers will be positioned between the sensor and the floor if a car is parked. The barrier will be removed if the parking space is unoccupied, allowing the sensor to detect the ground right away.

Conclusion

The creation of a prototype for an internet-based smart parking system demonstrates how well it can detect, process, and transfer parking information to the cloud. The important findings from testing data include the ability of sensor and RFID circuits to transfer data to Ubidot's IoT cloud without any data error in less than twelve seconds. The second is that the RFID circuit can read and write data back to the card in less than two seconds without experiencing any data errors. The third is that data can be processed by sensor, RFID, and server circuits in under two seconds. The last is data delivery to cloud, which can take up to 20 seconds longer than transmitting data from circuit to server.

References

1. Alarcon, R. and Wilde, E., 2010, April. Linking data from restful services. In Third Workshop on Linked Data on the Web, Raleigh, North Carolina (April 2010).

2. Al-Fagih, A.E., Al-Turjman, F.M., Alsalih, W.M. and Hassanein, H.S., 2013. A priced public sensing framework for heterogeneous IoT architectures. Emerging Topics in Computing, IEEE Transactions on, 1(1), pp.133-147.

3. Alpaydin, E., 2014. Introduction to machine learning. MIT press.

4. Anthes, G., 2012. HTML5 leads a web revolution. Communications of the ACM, 55(7), pp.16-17.

5. Armando, A., Carbone, R., Chekole, E.G. and Ranise, S., 2014, June. Attribute based access control for APIs in spring security. In Proceedings of the 19th ACM symposium on Access control models and technologies (pp. 85-88). ACM.

6. Atighetchi, M., Soule, N., Pal, P., Loyall, J., Sinclair, A. and Grant, R., 2013, October. Safe configuration of TLS connections. In Communications and Network Security (CNS), 2013 IEEE Conference on (pp. 415422). IEEE.

7. Bormann, C., Castellani, A.P. and Shelby, Z., 2012. Coap: An application protocol for billions of tiny internet nodes. IEEE Internet Computing, 16(2), p.62.

8. Centenaro, M., Vangelista, L., Zanella, A. and Zorzi, M., 2015. Long-Range Communications in Unlicensed Bands: The Rising Stars in the IoT and Smart City Scenarios. arXiv preprint arXiv:1510.00620.

9. Chinrungrueng, J., Sunantachaikul, U. and Triamlumlerd, S., 2007, January. Smart parking: An application of optical wireless sensor network. In Applications and the Internet Workshops, 2007. SAINT Workshops 2007. International Symposium on (pp. 66-66). IEEE.

10. Datta, S.K., Bonnet, C. and Nikaein, N., 2014, March. An IoT gateway centric architecture to provide novel M2M services. In Internet of Things (WF-IoT), 2014 IEEE World Forum on (pp. 514-519). IEEE.

11. Django, 2016. Django overview. Django Software Foundation. Accessed 2 April 2016. Available at: https://www.djangoproject.com/start/overview/

12. Egners, A., Gatzen, D., Panchenko, A. and Meyer, U., 2012, March. Introducing SOR: SSH-based onion routing. In Advanced Information Networking and Applications Workshops (WAINA), 2012 26th International Conference on (pp. 280-286). IEEE.

13. Framling, K., Kubler, S. and Buda, A., 2014. Universal messaging standards for the IoT from a lifecycle management perspective. Internet of Things Journal, IEEE, 1(4), pp.319-327.

14. Fu, Q. and Retscher, G., 2009. Active RFID trilateration and location fingerprinting based on RSSI for pedestrian navigation. Journal of Navigation, 62(02), pp.323-340.

15. Geng, Y. and Cassandras, C.G., 2012. A new "smart parking" system infrastructure and implementation. Procedia-Social and Behavioral Sciences, 54, pp.1278-1287.

16. Gupta, V., Goldman, R. and Udupi, P., 2011, June. A network architecture for the web of things. In Proceedings of the Second International Workshop on Web of Things (p. 3). ACM.

17. Herrero, J.L., Lozano, J. and Santos, J.P., 2015. A Restfull Approach for Classifying Pollutants in Water Using Neural Networks. In New Contributions in Information Systems and Technologies (pp. 371-380). Springer International Publishing.

18. Holt, G.K, Sep 16, 2014. Get Smart: SSH reverse tunnels for some access to smart meters. researchgate.net.

19. Hunkeler, U., Truong, H.L. and Stanford-Clark, A., 2008, January. MQTT-S—A publish/subscribe protocol for Wireless Sensor Networks. In Communication systems software and middleware and workshops, 2008. comsware 2008. 3rd international conference on (pp. 791-798). IEEE.

20. Ian Poole, 2014. LoRa Physical Layer & RF Interface. Radio-Electronics.com. Accessed 10 March 2016. Available:http://www.radio-electronics.com/info/wireless/lora/rf-interface-physical-layer.php

21. Jeliazkova, N. and Jeliazkov, V., 2011. AMBIT RESTful web services: an implementation of the OpenTox application programming interface. J. Cheminformatics, 3, p.18.

22. Kim, K.J., Kim, C.G., Whangbo, T.K. and Yoon, K., 2016. A continuous playing scheme on RESTful web service. Cluster Computing, pp.1-9.

23. Krco, S., Pokric, B. and Carrez, F., 2014, March. Designing IoT architecture (s): A European perspective. In Internet of Things (WF-IoT), 2014 IEEE World Forum on (pp. 79-84). IEEE.

24. LaGrone, B., 2013. HTML5 and CSS3 Responsive Web Design Cookbook. Packt Publishing Ltd.

25. Le Blond, S.P., Holt, A. and White, P., 2012, October. 3eHouses: A smart metering pilot in UK living labs. In Innovative Smart Grid Technologies (ISGT Europe), 2012 3rd IEEE PES International Conference and Exhibition on (pp. 1-6). IEEE.

26. Lee, E.A. and Seshia, S.A., 2015. Introduction to embedded systems: A cyber-physical systems approach. Lee & Seshia.

27. Lee, S., Kim, H., Hong, D.K. and Ju, H., 2013, January. Correlation analysis of MQTT loss and delay according to QoS level. In Information Networking (ICOIN), 2013 International Conference on (pp. 714-717). IEEE.