IoT-Based Smart Car Parking Sensor Using an ESP8266MOD and HC-SR04 Sensor

Noam Vaknin, Noam Perets
Ben Gurion University

Abstract: This article presents the development of an IoT-based smart car parking sensor system utilizing an ESP8266MOD microcontroller and Ultrasonic Ranging Module HC-SR04. The sensor communicates real-time parking space data to the AWS cloud and Arduino Cloud through an MQTT broker. The system includes LED indicators on the ESP8266MOD and an IoT remote app reflecting parking conditions. Additionally, a safety feature prompts user responses in emergency situations. The article also highlights security vulnerabilities discovered during firmware analysis, emphasizing the need for enhanced IoT data protection.

Keywords: Internet of Things (IoT), HC-SR04, Message Queuing Telemetry Transport (MQTT) Broker, ESP8266MOD, Amazon Web Services (AWS), Arduino Cloud.

I. Introduction

The appearance of Internet of Things (IoT) technologies has revolutionized various domains, prompting innovative solutions for everyday challenges. In line with this progression, our research focuses on the development of an IoT-based smart car parking sensor system. Leveraging the capabilities of the ESP8266MOD microcontroller and the HC-SR04 sensor, this project aims to address critical aspects of parking management by seamlessly integrating with cloud services. This paper aims to present a concise and clear introduction, outlining the problem addressed, the proposed solution, and the rationale behind our study. The contemporary challenges in parking management systems underscore the need for a sophisticated approach that goes beyond traditional methodologies. Existing systems often lack real-time data gathering, leading to inefficiencies in monitoring parking space conditions. In response to this, our smart car parking sensor system aims to bridge this gap by providing a comprehensive solution that seamlessly integrates with cloud services. To contextualize our research, it is essential to review the current state of parking management solutions. While various studies have explored smart parking systems encompassing features such as guidance systems [3], transit-based information [1], and e-parking automation [2], many fall short of providing a holistic solution. Our work diverges from existing approaches by integrating essential features such as parking log and alert system [1].
Key to our proposed solution is the use of the ESP8266MOD microcontroller and the HC-SR04 sensor. The ESP8266MOD establishes a connection to local Wi-Fi, facilitating communication with cloud services, while the HC-SR04 sensor ensures accurate distance measurements in the parking environment. The data gathered by the sensor is transmitted to the AWS cloud using an MQTT broker, creating a seamless flow of information that enhances the overall efficiency of the parking management system. In addition to addressing parking space monitoring, our system (Fig.1) incorporates dynamic LED indicators and a safety mechanism based on distance thresholds (Fig.6). These features enhance user experience and safety, setting our solution apart from traditional parking systems. As we delve into the technical aspects of our smart car parking sensor system in subsequent sections, it is crucial to underscore the innovation and potential impact of our project. By combining cutting-edge technologies and addressing the shortcomings of existing solutions, our research contributes to the evolution of parking management systems in urban environments. The integration of security analysis emphasizes the importance of data protection in IoT implementations, further enhancing the relevance and applicability of our work.

II. Related Work

In the realm of smart parking systems, our project stands alongside several notable studies, each contributing unique perspectives. One such study introduces an IoT-based parking solution using RFID and HC-SR04, incorporating features like membership payment and real-time slot availability [1]. While sharing the use of HC-SR04 sensors, our project distinguishes itself by seamlessly integrating with cloud services, incorporating dynamic LED indicators, and emphasizing a safety mechanism. Another relevant study explores IoT-based home automation, skillfully merging Raspberry Pi, AWS, and MQTT App, underscoring the versatility of AWS in diverse IoT applications [2]. Although our primary focus is on smart parking, our work aligns with this study by showcasing the application of AWS for secure data storage and emphasizing the importance of user data protection. A comprehensive review of smart parking solutions in urban areas is presented in another study, addressing challenges associated with growing car ownership. Our project aligns with this study in addressing urban parking management challenges, but uniquely stands out by incorporating specific features like LED indicators and a safety mechanism [3]. Additionally, a study evaluates the state-of-the-art technologies in smart parking sensors, emphasizing low-power consumption. While our project leverages HC-SR04 for distance measurements, it extends the discussion to include security aspects, reinforcing the significance of data protection in IoT implementations [4].

III. Proposed Solution

Our IoT-based smart car parking sensor system addresses the limitations of traditional parking management systems by introducing an innovative and comprehensive solution. Leveraging the ESP8266MOD microcontroller and the HC-SR04 Ultrasonic Ranging Module, our proposed system seamlessly integrates with cloud services to provide real-time data on parking conditions.

A. System Architecture

The core architecture (Fig.1) of our solution involves the ESP8266MOD microcontroller and the HC-SR04 sensor working concurrently. The ESP8266MOD establishes a secure connection to local Wi-Fi, enabling communication with cloud services, while the HC-SR04 sensor accurately measures distances within the parking environment[1]. The choice of the MQTT protocol facilitates efficient and reliable data transmission to the AWS and Arduino cloud.

B. Real-Time Data Transmission

To ensure timely and accurate data updates, our system employs the MQTT protocol. This lightweight and efficient protocol facilitates the transmission of distance measurements from the HC-SR04 sensor to the AWS and Arduino cloud. The real-time data is subsequently stored in the Amazon TimeStream database and Arduino Dashboard, providing a comprehensive parking log (Fig.7) for analysis[1].

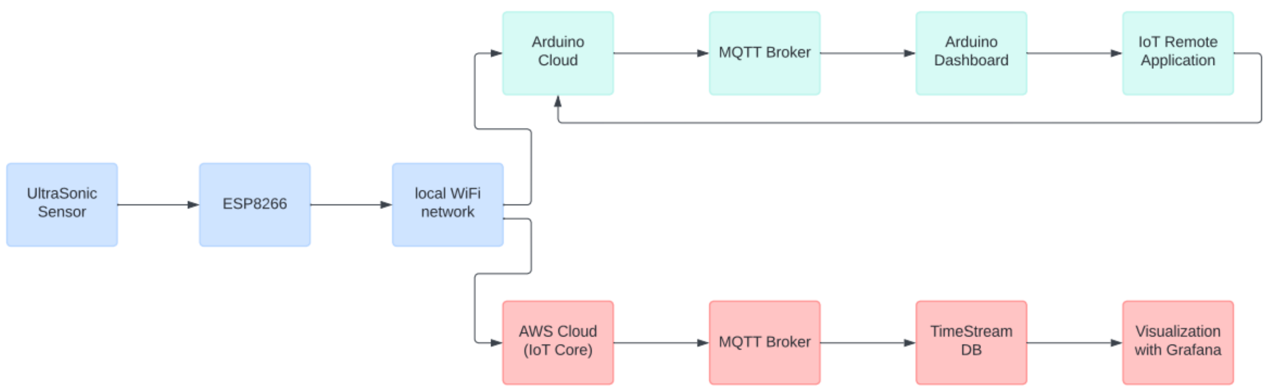

Figure 1: Block diagram of the workflow.

C. LED Indicators and Safety Mechanism

Our solution enhances user experience and safety through the incorporation of dynamic LED indicators on the ESP8266MOD and a safety mechanism based on predefined distance thresholds. When the distance exceeds 20cm, signaling a safe distance, the ESP8266MOD indicates this by turning off its LED indicator, while the IoT remote app displays a green light. Conversely, distances below 20cm prompt the ESP8266MOD to display a constant light, and the IoT remote app signals a red light, indicating a dangerous distance from an obstacle. A critical safety feature is activated when the distance drops below 5cm. The cloud sends a message to the IoT remote app, querying the user about their safety. Users can respond through the app, signaling whether everything is okay. A positive response triggers a message of reassurance, while a negative response prompts an emergency message, indicating a potential issue and signaling the need for immediate attention[2].

In summary, our proposed solution combines the strengths of the ESP8266MOD microcontroller and the HC-SR04 sensor to create a sophisticated IoT-based smart car parking sensor system. By addressing traditional limitations and incorporating features that prioritize user experience and safety, our solution contributes to the evolution of parking management systems, aligning with the demands of modern urban environments. The subsequent sections will delve into the technical details of our implementation and present the findings of our experimental validation.

IV. Security Measures

While our proposed IoT-based smart car parking sensor system demonstrates innovation in parking management, a critical aspect of our research unveiled vulnerabilities in the realm of data security. Through the physical extraction of firmware from the ESP8266MOD, we successfully accessed the RAM memory using esptool.

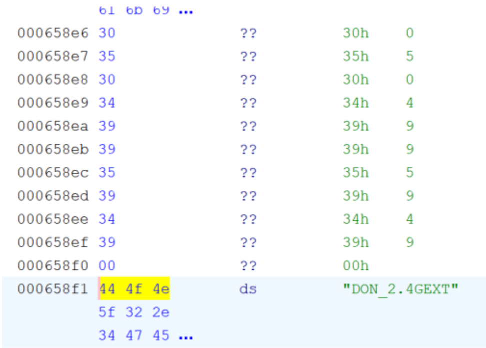

Figure 2: Extracting sensitive information using Ghidra.

Utilizing Ghidra[5], we identified and extracted private certificates, keys, and even the Wi-Fi password stored in the memory (Fig.2) . This discovery raises concerns about the vulnerability of sensitive information in IoT implementations. Acknowledging this, we emphasize the imperative for future work to enhance security practices in IoT projects. Strategies for improved protection of private data and the implementation of secure storage mechanisms warrant careful consideration to ensure the integrity and confidentiality of information in similar applications.

V. Experiments

A. Experimental Setup

The ultrasonic ranging module HC-SR04 [9], a fundamental component of our experimental setup, offers a non-contact measurement function with a range spanning from 2cm to 400cm, ensuring a versatile and accurate distance measurement capability. Noteworthy is its impressive ranging accuracy, capable of achieving precision up to 3mm.

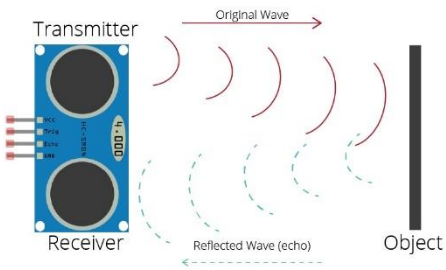

Figure 3: HC-SR04 Behavior

Comprising ultrasonic transmitters, a receiver, and a control circuit (Fig.3), the module operates on a basic principle: 1. initiating an IO trigger for a minimum of 10 microseconds to generate a high-level signal, 2. The module autonomously dispatches eight 40 kHz pulses and detects the existence of a pulse signal in return, and 3. If a signal is detected, the duration of the high-level output IO represents the time taken for the ultrasonic signal to travel to the target and back. The test distance is determined using the formula (1).

$$(1)\ Distance\ to\ Object = \frac{(Speed\ of\ sound)*time}{2}$$

$$Speed\ of\ sound = 0.343\,\frac{km}{s}$$

For our experiments, we restricted the sensor's measurement range to values within 2cm to 400cm, ensuring that only valid distances are registered. The sensor's wiring configuration includes connections for 5V supply, trigger pulse input, echo pulse output, and ground, facilitating a straightforward integration with the ESP8266MOD microcontroller for seamless data acquisition in our IoT framework. This strict range limitation ensures that the ultrasonic sensor captures meaningful and relevant environmental data for real-time monitoring applications.

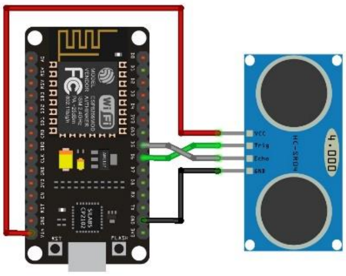

Figure 4: ESP8266MOD.

In our experimental setup, the ESP8266MOD microcontroller [6]-[8] serves as a key component for seamless data acquisition and integration into our IoT framework. The ESP8266MOD, based on the ESP-8266 32-bit microcontroller, boasts an 80 MHz clock speed and 4 MB of Flash memory, making it a powerful yet cost-effective choice for our application. This Wi-Fi-enabled module supports various communication interfaces, including Serial, SPI, I2C, and 1-Wire (Fig. 4) through software libraries. With 11 digital I/O pins, one analog input, and built-in 802.11 b/g/n Wi-Fi, the ESP8266MOD facilitates efficient control and communication with the HC-SR04 ultrasonic sensor. Its affordability and cost-effectiveness enhance the accessibility of our experimental setup for a wide range of applications.

The physical configuration [10] involves linking the TRIGGER_PIN (GPIO 5, D1) on the ESP8266MOD to the ultrasonic sensor's trigger input, allowing the microcontroller to initiate distance measurements. Simultaneously, the ECHO_PIN (GPIO 4, D2) on the ESP8266MOD is connected to the ultrasonic sensor's echo output, enabling the microcontroller to capture and measure the duration of the reflected ultrasonic pulse.

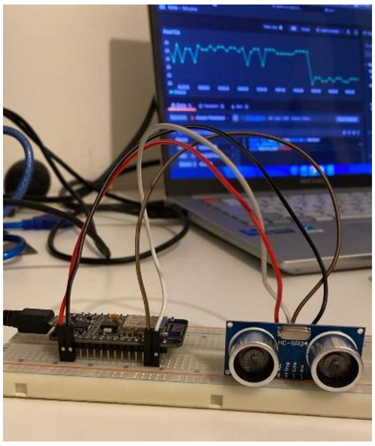

Figure 5: Connection of the hardware components.

B. Arduino Cloud Experiment

Our experiment with the Arduino Cloud platform aimed to seamlessly integrate our IoT-based smart car parking sensor system with a user-friendly cloud environment. The process began by establishing a connection to the Arduino Cloud platform and configuring cloud variables for real-time data display. This involved ensuring connectivity to the MQTT broker, local network, and Arduino Cloud. LED indicators were implemented on the device to provide immediate alerts when objects were detected too closely, enhancing user awareness. Furthermore, the integration with the user app [11] played a pivotal role in providing a comprehensive user experience and enabling seamless interaction with the device.

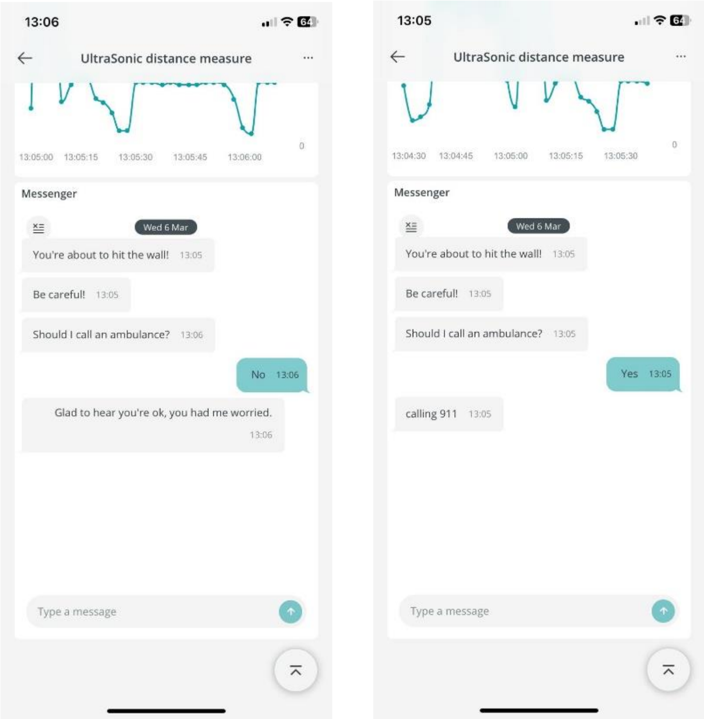

Figure 6: Interactive alert system.

Through live updating of sensor data, users could receive real-time feedback and monitor parking conditions. Rapid LED alerts were triggered when the measured distance fell below the predefined threshold of 20cm, facilitating timely and intuitive responses. To emphasize the interactive capabilities of our system, we utilized messaging functionality to alert users via the app when proximity to an object became critical. This process was visually represented with images showcasing the dynamic conversation between the cloud and the end-point user when the distance measured was below 5cm (Fig.6). Interactive chat prompts initiated in such instances allowed for immediate feedback and user response, ensuring an enhanced level of safety and engagement with the smart car parking sensor system. The results of the Arduino Cloud experiment validate the effectiveness of our proposed solution in real-world scenarios, laying the foundation for practical applications in urban parking management.

C. AWS Cloud Experiment

A seamless integration of IoT devices with Amazon Web Services (AWS) is established to enhance real-time environmental monitoring. The process initiates by establishing a connection to AWS, ensuring a robust communication link between the IoT devices and the cloud infrastructure. Leveraging the MQTT test client within AWS IoT Core, a dedicated rule is developed to facilitate the efficient transfer of data through the MQTT protocol [12] – [14]. Subsequently, the data is channeled through the created rule, enabling its storage in the Amazon TimeStream database. The retrieval and display of data from TimeStream are achieved using a specific SQL command, fostering precise data analysis.

The experimental setup accumulates a substantial dataset comprising 3000 measurements within the TimeStream database, providing a comprehensive foundation for subsequent analysis (Fig.7) [15].

Employing a designated SQL command, the dataset is visualized in Amazon Grafana, showcasing a graphical representation of the intricate relationship between time and distance measured in centimeters (Fig.7) [15].

Additionally, a specialized graph is generated, specifically highlighting instances where the measured distance is less than 20cm, offering nuanced insights into critical environmental conditions. This orchestrated integration of AWS services exemplifies a robust framework for IoT-based environmental monitoring, emphasizing the significance of seamless data transfer, storage, and visualization in fostering informed decision-making processes.

Rows returned (3033)

host	measure_name	time	measure_value::double	measure_value::varchar
$(hostname)	distance	2024-03-06 09:11:50.285000000	29.12100029	-
$(hostname)	distance	2024-03-06 09:11:55.475000000	29.10400009	-
$(hostname)	distance	2024-03-06 09:12:00.619000000	34.96900177	-
$(hostname)	distance	2024-03-06 09:12:05.717000000	28.3220005	-
$(hostname)	distance	2024-03-06 09:12:10.962000000	35.00299835	-
$(hostname)	distance	2024-03-06 09:12:16.111000000	53.36299896	-
$(hostname)	distance	2024-03-06 09:12:21.204000000	35.00299835	-
$(hostname)	distance	2024-03-06 09:12:26.306000000	29.10400009	-
$(hostname)	distance	2024-03-06 09:12:31.475000000	34.96900177	-

Figure 7: TimeStream Database Table

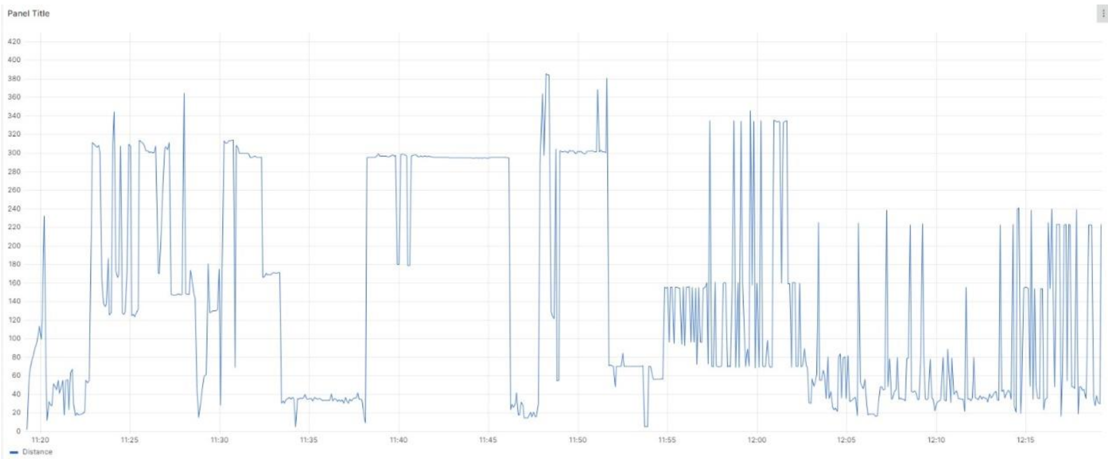

Figure 8: Visualization of the database using Amazon Grafana.

Figure 9: Specialized graph indicating the distance measured is less than 20cm.

VI. Conclusion

In this paper, we introduced an IoT-based smart car parking sensor system that leverages the ESP8266MOD microcontroller and HC-SR04 sensor to address critical challenges in parking management. Our proposed solution seamlessly integrates with cloud services, such as AWS and Arduino Cloud, providing real-time data on parking conditions. The incorporation of dynamic LED indicators on the ESP8266MOD and a safety mechanism based on predefined distance thresholds enhances user experience and safety.

A. Key Contributions

Our research makes significant contributions to the field of parking management systems by addressing key limitations in existing solutions. The integration of the ESP8266MOD and HC-SR04 ensures accurate distance measurements and efficient communication with cloud services. The real-time data transmission through the MQTT protocol and storage in the Amazon TimeStream database and Arduino Dashboard to create a comprehensive parking log(Fig.7) for analysis. The inclusion of dynamic LED indicators and a safety mechanism adds an extra layer of functionality to our system, prioritizing user safety and providing intuitive feedback. The IoT remote app further enhances user interaction and engagement, allowing users to respond to emergency situations promptly.

B. Security Considerations

While our IoT-based smart car parking sensor system demonstrates innovation, our security analysis uncovered vulnerabilities related to data security (Fig. 2). The extraction of sensitive information, including private certificates, keys, and Wi-Fi passwords, from the ESP8266MOD firmware highlights the importance of robust security measures in IoT implementations. Future work should focus on enhancing security practices to safeguard sensitive information in similar applications.

C. Experimental Validation

Our experiments, conducted on both the Arduino Cloud and AWS Cloud platforms, validate the effectiveness of our proposed solution in real-world scenarios. The seamless integration with cloud services, coupled with the interactive features of our system (Fig.6), demonstrates its practical applicability in urban parking management. The visualizations from Amazon Grafana (Fig.8 and Fig.9) showcase the complex relationship between time and distance, providing valuable insights for informed decision-making.

D. Future Work

As we conclude, it is essential to consider future directions for this research. Further refinement of security measures is crucial to ensure the confidentiality and integrity of sensitive data. Additionally, exploring the scalability of the system for large-scale urban deployments and evaluating its performance in diverse environmental conditions would contribute to the continued evolution of smart parking solutions.

In conclusion, our IoT-based smart car parking sensor system represents a significant step forward in addressing the challenges of parking management. By combining technological innovation, real-time data transmission, and user-centric features (Fig.6), our solution offers a robust framework for enhancing the efficiency and safety of parking systems in modern urban environments.

VII. References

[1] D. Eridani, Y. E. Windarto and G. Zaka, "IoT-Based Integrated Parking System Prototype using RFID and HC-SR04," *2023 10th International Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE),* Semarang, Indonesia, 2023, pp. 360-364, DOI: 10.1109/ICITACEE58587.2023.10277099.

[2] Singh, A. K., Arifi, A. A., Harikrishnan, R., Datta, B., & Wagle, S. A. (2022). IoT Based Home Automation Using App & AWS. In *Proceedings of the 2022 International Conference on Advances in Computing, Communication and Applied Informatics (ACCAI)* .DOI: 10.1109/ACCAI53970.2022.9752635

[3] C. Biyik et al., “Smart parking systems: Reviewing the literature, architecture, and ways forward,” *Smart Cities*, vol. 4, no. 2, pp. 623–642, 2021, DOI: 10.3390/smartcities4020032.

[4] T. Perković, P. Šolić, H. Zargariasl, D. Čoko, and J. J. P. C. Rodrigues, “Smart Parking Sensors: State of the Art and Performance Evaluation,” *J Clean Prod*, vol. 262, 2020, DOI: 10.1016/j.jclepro.2020.121181.

[5] National Security Agency. (2019). Ghidra: Software Reverse Engineering Framework. Version 11.0.1. GitHub. https://github.com/NationalSecurityAgency/ghidra.

[6] Espressif Systems. (2020). ESP8266 Technical Reference. Retrieved from https://www.espressif.com/sites/default/files/documentation/esp8266-technical_reference_en.pdf

[7] Arduino. (2017). Arduino Core for ESP8266 Documentation. Retrieved from https://arduino-esp8266.readthedocs.io/en/latest/

[8] CityOS Air. (2020). USB Drivers for NodeMCU v1.0 Documentation. Retrieved from https://cityos-air.readme.io/docs/1-usb-drivers-for-nodemcu-v10

[9] SparkFun Electronics. (2019). HC-SR04 Ultrasonic Sensor Datasheet. Retrieved from https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf

[10] Random Nerd Tutorials. (2021). ESP8266 NodeMCU with HC-SR04 Ultrasonic Sensor Tutorial. Retrieved from https://randomnerdtutorials.com/esp8266-nodemcu-hc-sr04-ultrasonic-arduino/

[11] techiesms. (2021). Using Arduino IoT Cloud with ESP32 | Is it better than New Blynk [Video]. YouTube. Retrieved from https://www.youtube.com/watch?v=rcCxGcRwCVk

[12] how2electronics. (2023). Connecting ESP8266 to Amazon AWS IoT Core using MQTT. Retrieved from https://how2electronics.com/connecting-esp8266-to-amazon-aws-iot-core-using-mqtt/

[13] Guy Tel-Zur - Computing. (2024). IoT demo: MQTT Pub/Sub between a local Raspberry Pi Pico W and AWS IoT Core service (in Hebrew) [Video]. YouTube. Retrieved from https://www.youtube.com/watch?v=j4hwnuJqyh4&t=62s&ab_channel=GuyTel-Zur-Computing

[14] Amazon Web Services. (2024). AWS IoT Developer Guide. Retrieved from https://docs.aws.amazon.com/pdfs/iot/latest/developerguide/iot-dg.pdf

[15] Amazon Web Services. (2021). Visualizing Data in Amazon Timestream using Grafana [Video]. YouTube. Retrieved from https://www.youtube.com/watch?v=pilkz645cs4