Petification: Node-RED Based Pet Care IoT Solution Using MQTT Broker

Haeram Kim*, Hyejong Kang*, Sunghan Kim*, Dukho Choi†, Jihyun You‡, Anthony Smith§, and Minsun Lee*

*Department of Computer Science and Engineering, Chungnam National University
†Department of International Trade, Chungnam National University
‡Department of Technology, Purdue University
§Department of Computer and Information Technology, Purdue University
Email: {haeram.kim1, kanghyejong1001, seonghan.kim.cnu, dukho.fin}@gmail.com,
{lee3450, ahsmith}@purdue.edu, mleeoh@cnu.ac.kr

***Abstract*—An increasing number of families have pets, but for owners who often leave home, taking good care of their pets is challenge. Most previous research on remotely feeding and watering pets uses a limited-feature of IoT platform. In this study, we propose an IoT solution, Petification, using the open-source project Node-RED with MQTT messaging protocol. Petification provides information about device connections, leftovers, food and water consumption through a web-based dashboard. It also provides an error notification service and remote feeding service. We mounted the load cell, HX711 amplifier, Raspberry Pi Zero W on the water supplier, and the MG90S servo motor on the Raspberry Pi of the feeding machine. Petification operated as suggested and provided emails or WhatsApp messenger-based error notifications and automatic feeding service when water or food ran out. In addition, we tested the accuracy of the automatic feeding amount of the food and found the average accuracy to be 88.38%. Future work includes modifying the food serving gate design of the machine to improve accuracy.**

***Index Terms*—IoT platform, Node-RED, MQTT, Smart pet care service**

I. Introduction

As more single-person households than in the past and the culture of raising pets spread, more and more families are raising pets. The growth of the pet industry is a good example of this trend. Profits for the pet industry have doubled annually over the past decade, from $48.4 billion in 2010 to $109.6 billion in 2020 [1]. Along with this trend, there is a growing demand for tracking pet health, which is required by pet owners who leave their homes for work or short trips.

Internet of Things (IoT) has recently received attention for a solution to these challenges. Various pet management IoT solutions proposed in the past, and the most commonly used open-source IoT platforms was Blynk [2]–[4]. Unfortunately, they no longer support open-source versions. Other common platforms are Adafruit IO [5] and Freeboard IO [6] and they do not support open-source versions. Additionally, most solutions support tracking food and water consumption or remote feeding but rarely support features such as device status information and error notification.

The proposed IoT solution "Petification" uses an open-source project for the IoT platform. Node-RED, a flow-based open-source visual programming tool [7], is used to implement the APIs of the IoT platform. This allows Petification to develop faster, provide a user-friendly user interface, and leverage the rich ecosystem of resources created by its users. To manage the entire message flow, we use Eclipse Mosquitto, one of the open-source implementations of the Message Queuing Telemetry Transport (MQTT) messaging protocol. Using the MQTT protocol as a message broker allows devices on IoT platforms to connect to users in a lightweight way [11].

The proposed system provides information such as device connectivity, food and water remaining, and consumption visually through a web-based dashboard. It also provides email and WhatsApp messenger-based error notifications and automatic feeding service when water or food runs out.

II. Related Literature

Vania *et al.* [8] proposed an IoT solution that identifies pets via RFID and feeds according to schedule. The IoT platform was developed using Arduino Uno, Node JS, and MySQL database. ESP8266 Wi-Fi Module in the feeder was used to communicate with user's mobile phone via MQTT over SSL/TLS protocol. Chen *et al.* [4] also implemented a solution using Arduino Uno and ESP8266 Wi-Fi Module with load cell, servo motor, and motion sensor. A mobile application was however based on Blynk IoT platform. The system could provide information such as food and water consumption, the number of defecation, and defecation duration. Nogueira *et al.* [9] developed more intelligent system using Raspberry Pi, Python, Telegram cloud service, and IBM Watson Assistant. The system can send alert with photos in case of abnormal situations and provides chat-based intelligent interface.

A pet care IoT solution by Sangvanloy *et al.* [3] visualizes daily consumption in real-time and automatically feeds at the scheduled time. It also adjusts the amount depending on pet's species and weight. The solution was developed using ESP32 Wi-Fi micro-controller, servo motor, load cell, and Blynk IoT platform. Vrishanka *et al.* [10] proposed an automated pet feeder which feed according to the remaining amount of the food in the bowl. Ultrasonic distance sensor and SG90

servo motor were mounted to Arduino Uno R3 to determine the amount remained in the bowl by measuring the distance between the entrance of the feed container and the inside of the bowl.

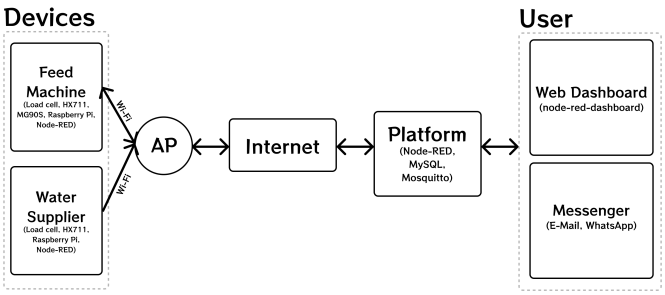

Fig. 1. Petification overview

III. Methodology

The architecture of our proposed system is designed to connect the feed machine and water supplier with the user. Each user and device can communicate in both directions through the platform. The feed machine and water supplier are connected to an Access Point (AP) through Wi-Fi. The messages published by the device are first sent to the AP via Wi-Fi and then to the platform via the internet. The platform processes the messages and sends the processed data to the user in the form of web dashboard pages, emails, and WhatsApp messages. Conversely, users can send requests to the platform through the web dashboard. While some requests require only the platform, actuator activation requests are sent to the device with the help of the platform. Fig. 1 shows the overview of Petification.

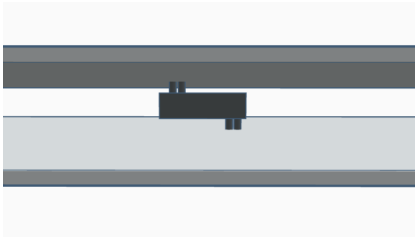

Fig. 2. Installaion of a load cell

A. Water Supplier

Weighing of water is performed with a single load cell that scales up to 5 kg with an accuracy of 1 g. A 500 mL water bottle used with an actual weight of 500 g is measured 500 times to obtain a reference unit during the calibration process. As shown in Fig. 2 the load cell is installed between the two plates to measure the pressure applied to the upper plate. And it is attached to the Raspberry Pi Zero W through the HX711 amplifier that converts analog signals to digital signals. The calibration process was performed to obtain a reference unit for converting a measured value of a load cell into weight data. The calibration formula is as follows:

$$referenceUnit = \frac{average(loadCellMeasurements)}{actualWeight} \quad (1)$$

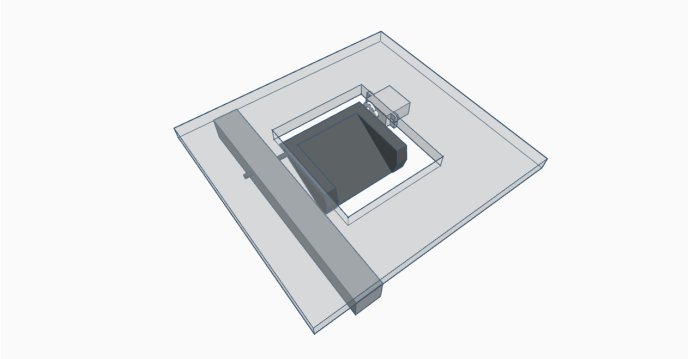

Fig. 3. Food gate of the feed machine

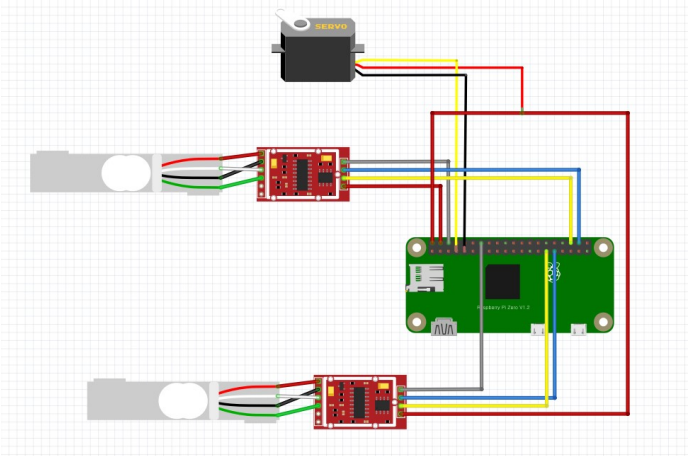

Fig. 4. Circuit diagram of the feed machine

B. Feed Machine

To weigh food, two load cells with HX711 amplifiers are installed under the food bowl and container in the same way as the water supplier. In order to provide food to pets, a MG90S servo motor is mounted on the feed machine to control the food gate. As shown in Fig. 3, the food gate has an open/close type. When the food gate opens, food is served to pets in a way that food rolls down the slope. A load cell with an amplifier and an MG90S servo motor are attached to the Raspberry Pi Zero W as shown in Fig. 4.

C. IoT Platform

User-to-device communication occurs when the user remotely serve food to the pet, and device-to-user communication occurs when information and notification to the user are provided. To send food serving message, the user can either set the serving schedule or press the button in the dashboard. The serving schedules are stored in the database and checked every

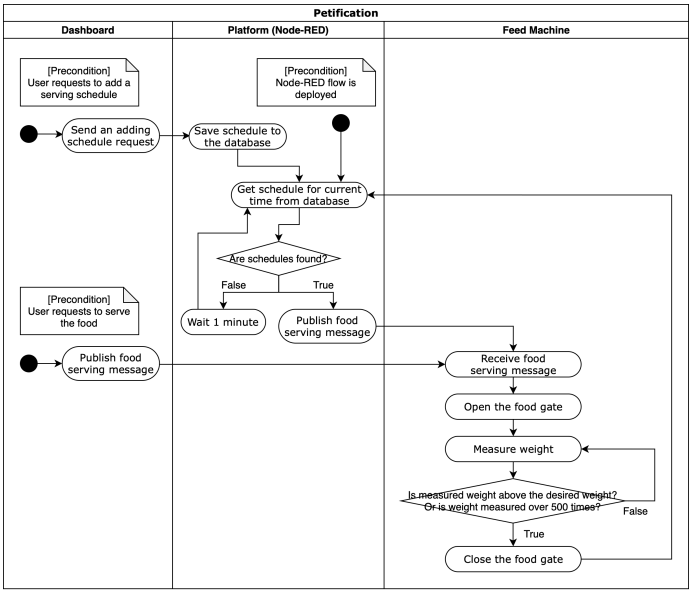

Fig. 5. Activity diagram for user-to-device communication

minute to serve the food when the scheduled time arrives. After the feed machine receives the message, it opens the food gate and repeatedly scales the food bowl to serve the desired amount of food to the pet. The detailed flow for the food serving is shown in Fig. 5.

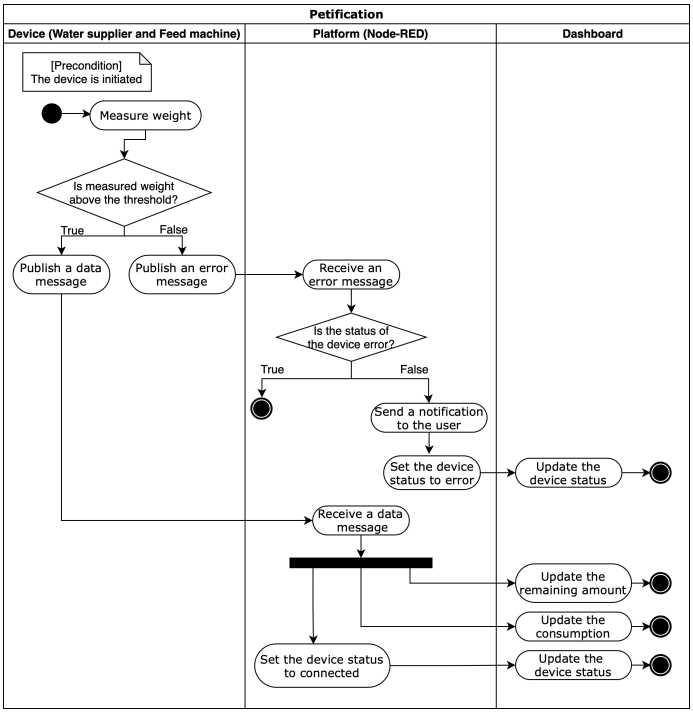

Fig. 6. Activity diagram for device-to-user communication

The device weighs food and water to provide information and alerts to the user. If the measured weight exceeds the threshold value of 10 g, a message containing the measured weight is sent to the platform. These data messages are used to update and display the balance, consumption and device status to the user. However, if the measured weight is below a threshold, an error message is sent to the platform to update the device status and inform the user that food or water is empty. Fig. 6 shows the detailed device-to-user flow and Algorithm 1 shows the pseudocode for calculating consumption.

Algorithm 1 Calculate consumption

procedure CALCCONSUMPTION
  $prevConsumption \leftarrow$ previous *consumption*
  $currScale \leftarrow$ *scale* of received message
  $prevScale \leftarrow$ *scale* of previous record
  $decrease \leftarrow 0$
  if $precScale \neq null \ \& \ prevScale > currScale$ **then**
    $decrease \leftarrow prevScale - currScale$
  end if
  return $prevConsumption + decrease$
end procedure

Three open-source projects are used for the platform: Node-RED, Eclipse Mosquito, and MySQL.

1) Node-RED: Node-RED is a flow-based, open-source visual development tool that makes it easy for developers and non-developers to develop programs [12]. One of Node-RED's strengths is its powerful community; Various nodes such as dashboard widgets and database drivers are being deployed through the Node Package Manager (NPM). Another advantage is that it can be run in a variety of environments, including local, Raspberry Pi, Docker, and cloud instances. Thus, Node-RED v2.2.1 is installed in the platform server as well as the Raspberry Pis in both water supplier and feed machine.

2) Eclipse Mosquitto MQTT message broker: MQTT is an OASIS standard protocol and provides a lightweight publish/subscribe messaging transport for IoT [11]. The publish/subscribe model of the MQTT protocol allows IoT platforms to support bidirectional communication in such a way that by subscribing to a topic of a message, others can receive messages published to that topic. In this study, Eclipse Mosquitto v1.4.15 is used as the MQTT message broker, an open-source implementation of the MQTT protocol 5.0, 3.1.1 and 3.1 versions [13].

3) MySQL: MySQL is a fast, flexible, and easy-to-use open-source database with a relational database management system (RDBMS). In addition to performance and security aspects, MySQL is considered a suitable database for managing effective data flow because of its compatibility with Node-RED. And MySQL v5.7.37 is integrated as a component of the platform.

IV. IMPLEMENTATION AND RESULT

A. Water supplier

Calibrating and measuring the weight of water is done in python code. Measurement results are available in Node-RED using an exec node that can execute python code.

However, there is always a possibility of measurement error even if calibration is performed. To reduce the measurement error after calibration, we used the smooth node of the node-red-node-smooth node module. After weighing, the switch node determines if the weight is below a threshold to determine the type of message being sent (weighted data or error). The messages are sent after the function node prepares the message and the "mqtt out" node publishes the message.

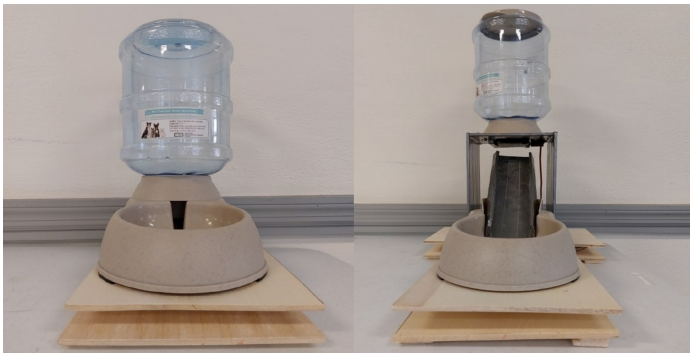

Fig. 7. Water supplier(left) and Feed machine(right)

B. Feed machine

It implements a flow to provide a certain amount of food in addition to the weighing flow, which is similar to a water supplier. The food serving flow starts after a message arrives containing the desired serving amount as the message payload. The join-wait node in the node-red-contrib-join-wait node module waits for a food serving message and returns an object after the message arrives. Then, the current weight is repeatedly measured and compared by the loop node from the node-red-contrib-loop-processing and change node. The food gate opens on the first loop and closes when the current weight is above the desired weight or the loop is repeated 500 times. Open and close using the pi-gpiod out node of the node-red-node-pi-gpiod node module. Both feed machine and water supplier are implemented shown in Fig. 7.

C. IoT Platform

Mosquitto, MySQL, and Node-RED are installed on the cloud instance, and the firewall, DNS, and certificate for TLS/SSL communication are configured for networking. In Node-RED, seven IoT platform components are implemented as seven flows.

1) MQTT Controller: The MQTT controller flow on Node-RED is the gateway for MQTT messages to go to Node-RED. The main purpose of this flow is to parse the topic and payload of the incoming MQTT message to provide useful information such as the MQTT username and client ID to other flows. The mqtt-in node is used to subscribe to and receive MQTT messages, and the function node is used to parse the message and transform it into an object.

2) Rule Engine: The purpose of the Rule Engine is to activate actions based on MQTT messages. Rule Engine's logic is inspired by "Building Your Own IoT Platform" [14]. For every message published, the Rule Engine checks the rules table in the database to find a rule whose message pattern meets the message content. The action corresponding to the rule is defined in the ReST API format, so the action is activated by sending an HTTP request.

3) Device Manager: The Device Manager flow is to handle devices attached to the platform. It provides a ReST API to add new devices to the platform, modify the status of devices, and delete devices. It also provides functionality for publishing food serving messages to specific devices.

Fig. 8. Screenshot for Feed Machine tab of the dashboard

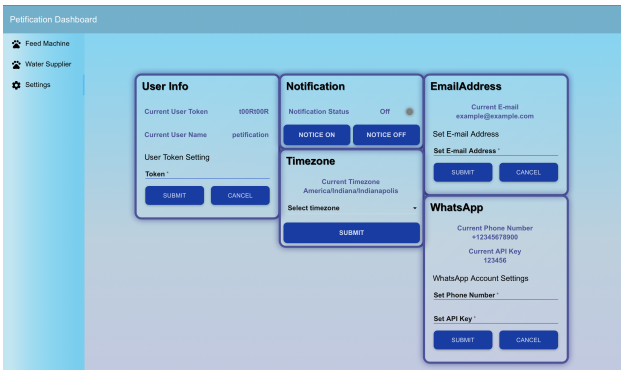

Fig. 9. Screenshot for User Setting tab of the dashboard

4) Dashboard: This provides users with a graphical user interface (GUI). The dashboard consists of three tabs: Feed Machine tab, Water Supplier tab, and User Settings tab.

- *Feed Machine Tab:* Under the Feed Machine tab, Device Information widget, Feed Machine Statistics widget, and Serve Food widget are provided. As all the functionalities this tab provides are device specific, selecting a device is included in the Device Information widget and is displayed as a dropdown. The Feed Machine Statistics widget provides leftover food and food consumption. The amount of leftover is displayed in a gauge and food consumption is displayed in a line graph. Users can check all the feeding schedules in the table format, add a new

schedule by user input interface, and delete the existing schedule by clicking the row of the table. Fig. 8 shows the Feed Machine tab on the Dashboard.

- *Water Supplier Tab:* Similar to the Feed Machine tab, the Water Supplier tab provides two widgets: Device Information widget and Water Supplier Statistics widget. The mechanism for each widget is identical to the Feed Machine tab.
- *User Settings Tab:* The User Settings tab provides five widgets: User Information, Timezone settings, Notification settings, E-Mail address settings, and WhatsApp settings. Notification settings, E-Mail address settings, and WhatsApp settings widgets are to control error notification. Users can turn notifications on or off with the Notification settings widget and decide which accounts receive email and WhatsApp messages. Fig. 9 shows the screenshot of the implemented User Settings tab.

5) User Management: The user managing flow is to provide a ReST API for modifying user settings. Users can manage 4 settings: enable/disable notifications, email address, WhatsApp account information, and the time zone in which the user resides. Also, a ReST API for sending email and WhatsApp messages is included in this flow using the "email" node of the node-red-node-email node module and the WhatsApp-bot node of the node-red-contrib-whatsappbot node module.

6) Schedule Engine: Handling and executing schedules are key features of this flow. Every minute, the scheduling engine checks the schedule table in the database and runs jobs scheduled to be active at that time. Similar to the rule engine, each corresponding action is defined in the ReST API format, and an HTTP request is sent when the action is executed.

7) Time-series Manager: All published MQTT messages are stored in a time series data table, and managing the stored messages is the primary purpose of the Time-series Manager flow. Time-series Manager flows provide APIs to other flows, such as dashboard flows, to get and utilize time series data.

TABLE I
TESTING RESULT OF AUTOMATIC FEEDING

Desired Weight(g)	***Average Served Weight(g)***	***Accuracy (%)***	***Error (%)***
15.00	15.66	95.63	4.37
20.00	20.26	98.73	1.28
50.00	57.60	84.80	15.20
60.00	80.32	66.13	33..87
200.00	193.19	96.60	3.40
Average Rate		88.38	11.62

D. Automatic Feeding Test

One of the features of this solution is the automatic food feeding. We evaluated the accuracy of the automatic food serving feature. The difference between the five desired weights and the average actual served weight are compared in Table I. The solution has an average accuracy rate of 88.38%. The reason for the error is that food often gets stuck in front of the food gate, and the transfer speed is too fast for the load cell to detect. This is due to the limitation of the feed machine and can be resolved by modifying the design of the machine.

V. CONCLUSION

In summary, we proposed the pet-care IoT solution "Petification" that can take care of pets remotely. Petification supports tracking food and water consumption and remote feeding with the web-based dashboard. We also demonstrated that the solution provided device status information and error notification. The advantages of the open-source projects are to develop faster, and leverage the rich ecosystem of resources created by its users. We expect to add more features to extend Petification by leveraging open-source advantages. Current limitation of Petification is the relatively low accuracy of automatic food serving due to the design of the food gate and the sensitivity of the load cell. Therefore, future work includes improving the accuracy of servings by addressing these issues.

ACKNOWLEDGMENT

This research was supported by the MSIT(Ministry of Science and ICT), Korea, under the National Program for Excellence in SW) supervised by the IITP(Institute of Information & communications Technology Planning & Evaluation) in 2021"(2021-0-01435)

REFERENCES

[1] M. Hanson. "Pet Industry Statistics" spots.com. https://spots.com/pet-industry-statistics/ (accessed Jan. 25, 2022).
[2] Accessed: Feb. 1, 2022. [Online]. Available: https://www.instructables.com/IOT-Pet-Feeder-Using-the-Blynk-Mobile-App-an-ESP82/
[3] T. Sangvanloy and K. Sookhanaphibarn, "Automatic Pet Food Dispenser by using Internet of Things (IoT)," 2020 IEEE 2nd Global Conference on Life Sciences and Technologies (LifeTech), Kyoto, Japan, Mar. 10-12, 2020.
[4] Y. Chen and M. Elshakankiri, "Implementation of an IoT based Pet Care System," 2020 Fifth International Conference on Fog and Mobile Edge Computing (FMEC), Paris, France, Apr. 20-23, 2020.
[5] Accessed: Feb. 4, 2022. [Online]. Available: https://iotdesignpro.com/projects/google-assistant-controlled-iot-pet-feeder-using-esp8266
[6] Accessed: Feb. 5, 2022. [Online]. Available: https://create.arduino.cc/projecthub/circuito-io-team/iot-pet-feeder-10a4f3
[7] Node-RED [Online]. Available: https://nodered.org/about/
[8] Vania, K. Karyono and I. H. T. Nugroho, "Smart dog feeder design using wireless communication, MQTT and Android client," 2016 International Conference on Computer, Control, Informatics and its Applications (IC3INA), Tangerang, Indonesia, Oct. 3-5, 2016.
[9] R. Nogueira, H. Araújo and D. Prata. (Apr. 2019). Robot Chow: Automatic Animal Feeding with Intelligent Interface to Monitor Pets. International Journal of Advanced Engineering Research and Science. [Online]. Available: https://ijaers.com/detail/robot-chow-automatic-animal-feeding-with-intelligent-interface-to-monitor-pets/
[10] P. N. Vrishanka, P. Prabhakar, D. Shet and K. Rupali, "Automated Pet Feeder using IoT," 2021 IEEE International Conference on Mobile Networks and Wireless Communications (ICMNWC), Tumkur, Karnataka, India, Dec. 3-4, 2021.
[11] MQTT [Online]. Available: https://mqtt.org
[12] N. B. Kamarozaman and A. H. Awang, "IOT COVID-19 Portable Health Monitoring System using Raspberry Pi, Node-Red and ThingSpeak," 2021 IEEE Symposium on Wireless Technology & Applications (ISWTA), Shah Alam, Malaysia, Aug. 17-17, 2021.
[13] Eclipse Mosquitto [Online]. Available: https://mosquitto.org/
[14] A. Tamboli, "Build Your Own IoT Platform," in *Apress*, 1st ed, 2019