Petification: Node-RED based Pet Care IoT Solution using MQTT Broker

Haeram Kim
Computer Science and Engineering
Chungnam National University
Daejeon, Korea
haeram.kim1@gmail.com

Hyejong Kang
Computer Science and Engineering
Chungnam National University
Daejeon, Korea
kanghyejong1001@gmail.com

Sunghan Kim
Computer Science and Engineering
Chungnam National University
Daejeon, Korea
seonghan.kim.cnu@gmail.com

Dukho Choi
International trade / software convergence
Chungnam National University
Daejeon, Korea
dukho.fin@gmail.com

Jihyun You
Cybersecurity
Purdue University
West Lafayette, IN, USA
you62@purdue.edu

***Abstract*—While there are an increasing number of households owning pets, it is challenging for owners who leave home often to take good care of their pets. However, most previous studies of remotely feeding and watering pets use limited featured IoT platforms and do not provide notifications or alerts about how much food and water is left. Thus, an IoT solution "Petification" is proposed in the study, using the open source project Node-RED with MQTT messaging protocol. Petification provides information on device connectivity, remaining food and water and consumption through a web-based dashboard. Additionally, it provides error notification service and remote feeding service. The water supplier and feed machine are attached to the platform to feed and water the pet and scale the weight of the remaining. The load cell, HX711 amplifier, and Raspberry Pi Zero W are mounted to the water supplier and feed machine and an MG90S servo motor is mounted to the Raspberry Pi of the feed machine. However, served amount of the food sometimes mismatches with the desired amount while testing the implementation. Thus the future plan can be to enhance the food gate to serve the exact amount of food.**

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

I. Introduction

As the number of single-person household increases and the raising pet culture spread compared to the past, the number of households raising pets is increasing. The growth of the pet industry is a good example showing this trend: Profit of pet industries have doubled every year over 10 years, from $48.4 billion in 2010 to $109.6 billion in 2020 [1]. Along with this trend, demand for tracking pet wellness is increased accordingly as it is hard to track the pet's status for people who leave their house frequently.

Internet of Things (IoT) has recently received attention for the solution to these challenges. However, despite the various pet care IoT solutions proposed in the past, most of the IoT platforms used have limitations of not being open-sourced. For instance, the most commonly used IoT platform such as "blynk" [2]–[4] has stopped supporting open-source versions, and other common platforms such as "Adafruit IO" [5] and "Freeboard IO" [6] does not support open-source versions. Moreover, while many solutions support tracking food and water consumption or remote feeding, few support features such as device status information and error notification.

The proposed IoT solution "Petification" uses open-source projects for the IoT platform. Node-RED which is a flow-based open-source visual programming tool [7] is used to implement the APIs of the IoT platform. This allows Petification to be able to develop more faster, provide user-friendly user interface to the user, and make use of rich resource ecosystem generated by users. To manage overall message flows, Eclipse Mosquitto which is one of the open-source implementations of the MQTT messaging protocol is used. Using the MQTT protocol as a message broker allows the IoT platform to connect the device to the user in a lightweight way [8].

Petification provides a wide array of information such as device connectivity, the remaining amount of food and water, and amount of consumption visually with the web-based dashboard. It also provides an email and "WhatsApp" messanger based error notification service when the water or food is running out, as well as automatic feeding service. It also provides error notification via email and "WhatsApp" messenger when water or food is running out as well as automatic feeding service. The water supplier and feed machine are attached to the platform to feed and water the pet and to scale the weight of the remaining amount.

II. Related Literature

T. Sangvanloy *et al.* [3] proposed a pet care IoT solution that visualizes the daily food consumption in real-time and automatically feeds the pet according to the scheduled time. The conducted research also provides an automatic serving amount adjustment feature based on the pet's species and

weight. The IoT solution was implemented with ESP32 Wi-Fi micro-controller, servo motor, load cell, and "Blynk" IoT platform.

Y. Chen *et al.* [4] proposed a pet care IoT solution that provides information such as food and water consumption, the number of defecation, and defecation duration. It was implemented using Arduino Uno and ESP8266 Wi-Fi Module with load cell, servo motor, and motion sensor. A mobile application is provided to the user with the help of the "Blynk" IoT platform.

P. N. Vrishanka *et al.* [9] proposed an automated pet feeder which serves food to the pet according to the remaining amount of food in the food bowl. An ultrasonic distance sensor and SG90 servo motor were mounted to Arduino Uno R3 in this research. An ultrasonic distance sensor is used to determine the remaining food amount by measuring the distance from the entrance of the feed container to the inside of the bowl.

Rogerio Nogueira *et al.* [10] proposed a system that can serve food or water according to pre-scheduled times and amounts. Additionally, the system provides a notification service for abnormal situations with photos and alerts and a "chat-based" intelligent interface to the user. This research is conducted with Raspberry Pi, Python, Telegram cloud service, and IBM Watson Assistant.

Vania *et al.* [11] proposed an IoT solution that identifies each pet via RFID and serves food by schedule. The dog feeder in the research was developed using Arduino Uno and ESP8266 Wi-Fi Module with load cell and DC motor. An Android mobile application is provided to the user and the IoT platform is developed with Node JS and MySQL database. Communication between the device and the user is based on MQTT over SSL/TLS protocol.

III. Methodology

The system architecture for Petification is designed to connect the feed machine and water supplier with the user. Each user and device can communicate bi-directionally through the platform. The feed machine and water supplier are connected to the Access Point (AP) through Wi-Fi. Thus, all messages published from the device are first sent to the AP through Wi-Fi, and then to the platform through the internet. The platform processes the messages and sends the processed data to the users in form of a web dashboard page, email, and "WhatsApp" messages. On the contrary, users can send requests to the platform via the web dashboard. While some requests require only the platform, requests to activate the actuator are sent to the device with the help of the platform. Figure 1 shows the overview of for Petification.

A. Water Supplier

Measuring the weight of the water is done with single load cell which scales up to a maximum of 5kg with 1g accuracy. As shown in figure 3, The load cell is installed between the two plates to measure the pressure applied to an upper plate as shown in figure 2 and is attached to the Raspberry Pi Zero W

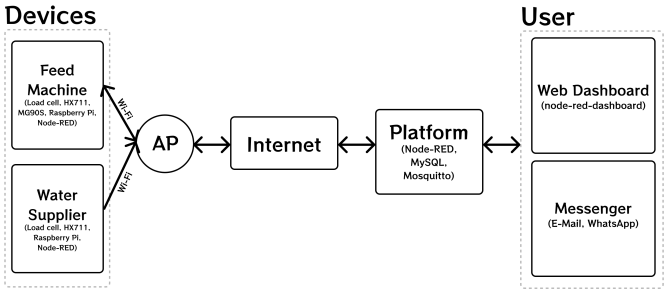

Fig. 1. Petification overview

through the HX711 amplifier which converts the analog signal to the digital signal. To convert the measurement of the load cell to the weight data, the calibration process has proceeded to get the reference unit. The formula for calibration is shown below:

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

A 500mL water bottle which is used as an actual weight of 500g is measured 500 times to get a reference unit in the calibration process.

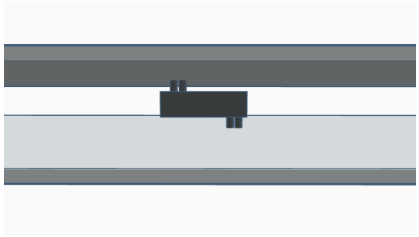

Fig. 2. Installaion of a load cell

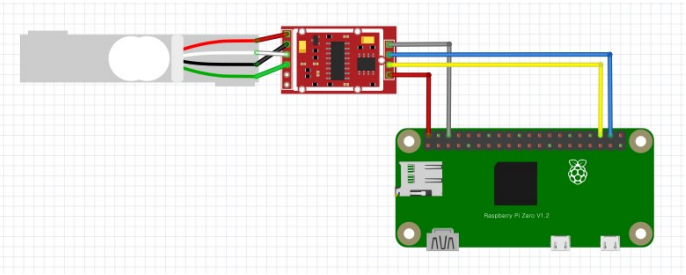

Fig. 3. Circuit diagram of the water supplier

B. Feed Machine

To measure food weight, two load cells with an HX711 amplifiers are installed under the food bowl and container in the same way as the water supplier. To serve the food to the pet, an MG90S servo motor is mounted to the feed machine to control the food gate. As shown in figure 4, the food gate has open/close type; The food is served to the pet in a way that the food rolls down the slope when the food gate is opened.

The load cells with an attached amplifier and a MG90S servo motor are attached to a Raspberry Pi Zero W as shown in figure 5.

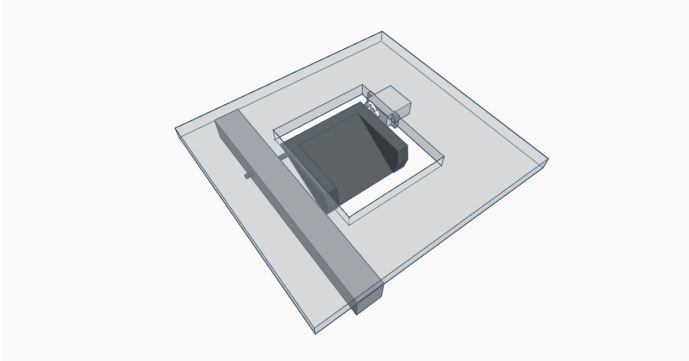

Fig. 4. Food gate of the feed machine

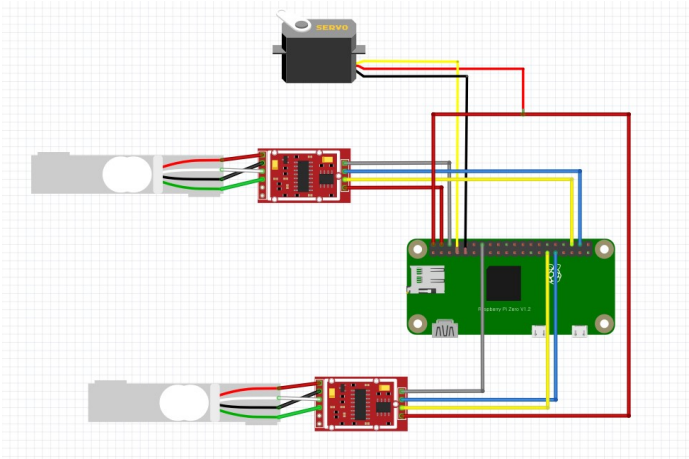

Fig. 5. Circuit diagram of the feed machine

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 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 figure 6.

In order to provide information and notification to the user, the weight of the food and water is measured by the devices. When the measured weight is above a threshold, which is 10g, a message containing the measured weight is sent to the platform. This data message is used to update and show the remaining amount, consumption, and device status to the user. However, when the measured weight is below the threshold,

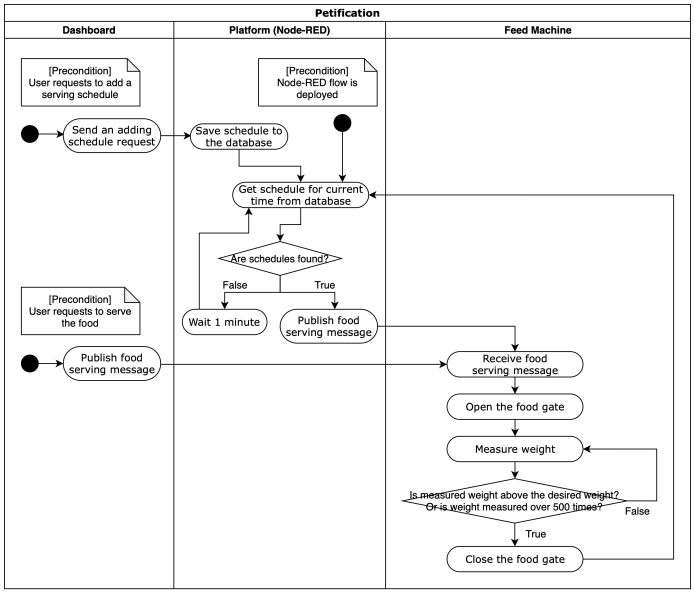

Fig. 6. Activity diagram for user-to-device communication

an error message is sent to the platform to update the device status and to notify the user that the food or water is empty. Figure 7 shows detailed device-to-user flow, and algorithm 1 shows the pseudo-code for calculating comsumption.

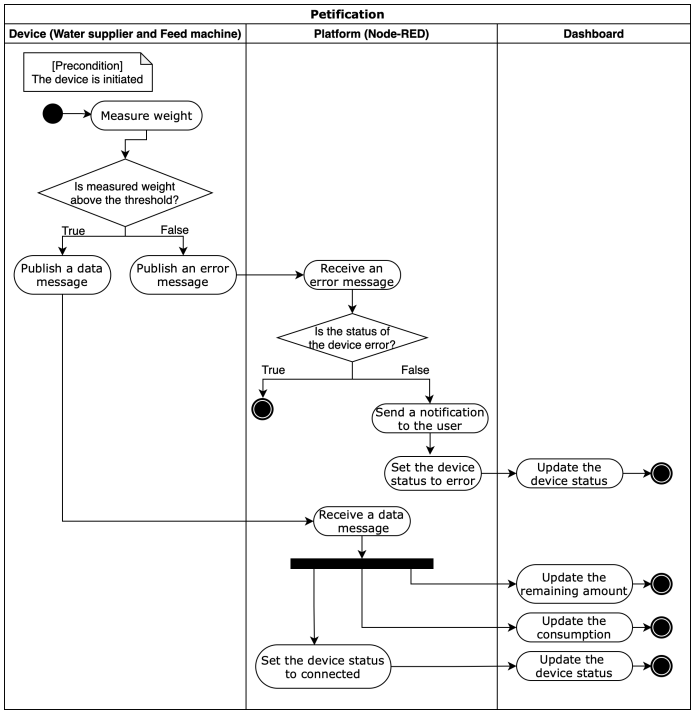

Fig. 7. Activity diagram for device-to-user communication

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

1) Node-RED: Node-RED is a flow-based open-source visual development tool that is easy for developers and non-

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 $precScale \neq null \quad \& \quad prevScale > currScale$ **then**
    $decrease \leftarrow prevScale - currScale$
  return $prevConsumption + decrease$

developers to develop programs [12]. One of the strengths of the Node-RED is its powerful community; Various nodes such as dashboard widgets and database drivers are being distributed through Node Package Manager (NPM). Another strength is that it can run on various environments such as local, Raspberry Pi, Docker, and Cloud Instance. Thus, Node-RED v2.2.1 is installed in not only the platform server but also Raspberry Pi in both water supplier and feed machine.

2) Eclipse Mosquitto MQTT message broker: MQTT (Message Queuing Telemetry Transport) is an OASIS standard protocol and provides lightweight, publish/subscribe messaging transport for IoT [8]. The publish/subscribe model of the MQTT protocol allows the IoT platform to support bi-directional communication in a way that one can subscribe to the "topic" of the message to receive a message published to that "topic" by another. In this research, Eclipse Mosquitto v1.4.15 is used as an MQTT message broker, which is an open-source implementation for 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 RDBMS (relational database management system). In addition to the performance and security perspectives, MySQL is considered a suitable database to manage effective data flows because it has good compatibility with Node-RED. For this reason, MySQL v5.7.37 is integrated as a component of the platform.

IV. IMPLEMENTATION

A. Water supplier

Calibrating and measuring the weight of the water is done by python code. The result of a measurement can be used in Node-RED by using "exec" node, which can execute the python code.

However, even if the calibration has proceeded, there is always the possibility of measurement error. To reduce measurement error after calibration the "smooth" node from the "non-red-node-smooth" node module was used. After weight measurement the "switch" node determines what type of message (weight data or error) is sent by checking if the weight is below the threshold value. The messages are sent after the "function" node prepares the message and the "mqtt out" node publishes the message. The Node-RED flow for publishing weight or error messages is shown in figure 8 and the implemented water supplier is shown in figure 9.

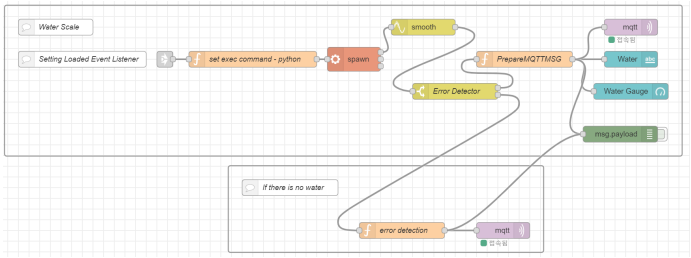

Fig. 8. Node-RED screenshot for publishing weight or error message

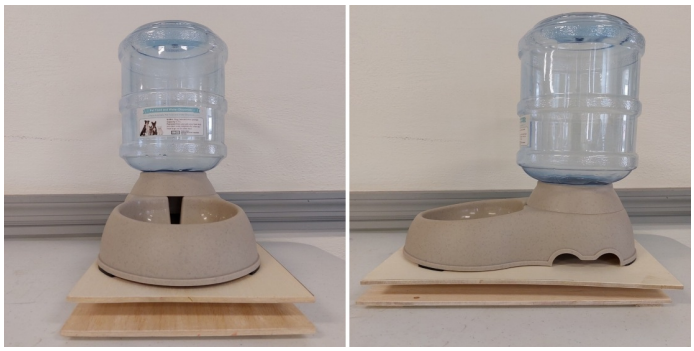

Fig. 9. Result of implementation for water supplier

B. Feed machine

The flow for serving a certain amount of food is implemented in addition to the weight measuring flow, which is similar to the water supplier. Food serving flow begins after a message containing desired serving amount as a message payload arrives. The "join-wait" node of the "node-red-contrib-join-wait" node module waits for a food serving message and returns an object after the message has arrived. 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 is opened for the first loop and is closed when the current weight is above the desired weight or the loop is repeated 500 times. The "pi-gpiod out" node of the "node-red-node-pi-gpiod" node module is used to open and close it. The Node-RED flow for serving food is shown in figure 10 and the implemented feed machine is shown in figure 11.

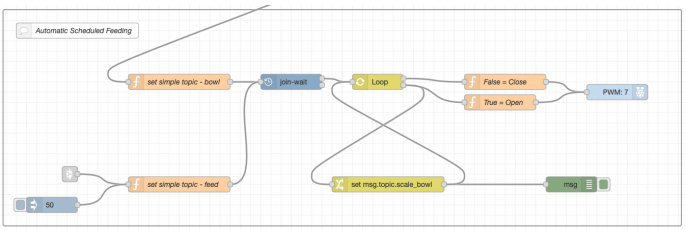

Fig. 10. Node-RED screenshot for automatic feeding

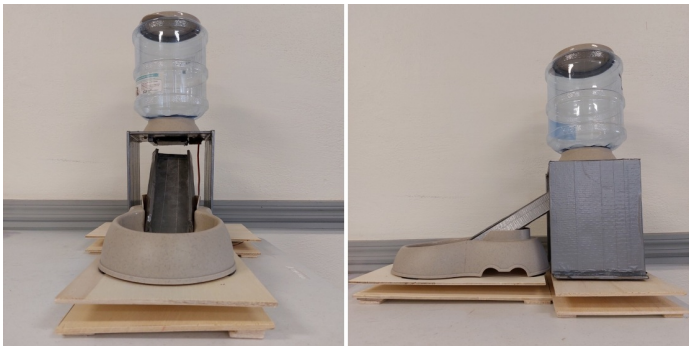

Fig. 11. Result of implementation for feed machine

C. IoT Platform

The platform for Petification uses cloud instances for deployment. On the cloud instance, Mosquitto, MySQL, and Node-RED are installed and the firewall, DNS, and certificate for TLS/SSL communication are configured for networking. In the Node-RED, 7 IoT platform components are implemented as 7 flows. A screenshot for implemented Node-RED flows is shown in Figure 12.

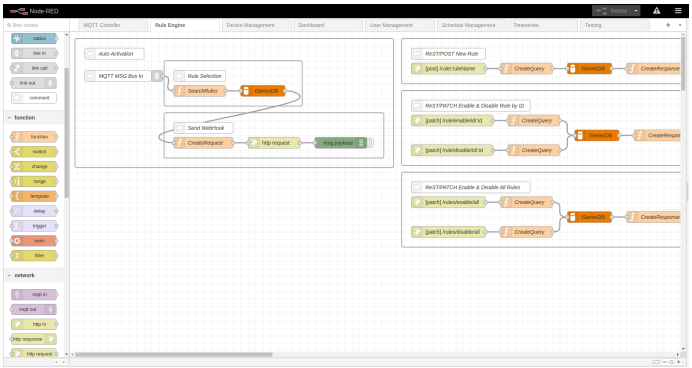

Fig. 12. Node-RED flow screenshot of IoT platform

1) MQTT Controller: MQTT Controller flow in the Node-RED is the gateway for MQTT messages to enter 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 and receive the MQTT message and the ”function” node is used to parse the message and convert it to an object.

2) Rule Engine: The purpose of Rule Engine is to activate actions according to MQTT messages. Logics of the Rule Engine was inspired by “Build Your Own IoT Platform” [14]. For every published message, Rule Engine checks the rule table of the database to find the rules whose message pattern satisfies the message content. Actions that corresponded to the rules are defined as ReST API form, thus activating action will be progressed as sending an HTTP request.

3) Device Manager: The main purpose of the Device Manager flow is to handle devices that are attached to the platform. It provides ReST APIs that can 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.

4) Dashboard: Dashboard Manager provides Graphical User Interface (GUI) to the user. This flow provides visual status of remaining amount and consumption to the user. It also provides buttons to serve food and input areas to set user settings. A screenshot of the implemented dashboard is shown in figure 13.

Fig. 13. Screenshot for Feed Machine tab of the dashboard

5) User Management: The purpose of the User Manager flow is to provide a ReST API for modifying user settings. Users can manage these 4 settings: Enable/disable the notification, Email address, WhatsApp account information, and timezone where the user lives. Also, a ReST API for sending email and ”WhatsApp” message 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 is the main functionality of this flow. Every minute, the schedule engine checks the schedule table of the database and executes the actions that are scheduled to be activated at that time. Similar to the rule engine, each corresponded action is defined as a ReST API form, thus an HTTP request is sent when the action is executed.

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

V. Experiment and Testing

Based on the results of the implementation, testing is conducted only on quantitatively verifiable results, which is accuracy testing of the food serving feature. For testing, the difference of 5 desired weight and the average of actual served weight is compared.

Desired Weight (g)	Average Served Weight (g)	Accuracy Rate (%)	Error Rate (%)
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

Fig. 14. Testing result of automatic feeding

Figure 14 shows the testing results for automatic feeding. The maximum error rate for the solution was 33.87%, and the minimum error rate was 1.28%. The reason for the error is that the food is often stuck in front of the food gate and the food serving speed is too fast for the load cell to detect. It is a limitation for the feed machine and can be solved if the design is modified correctly.

VI. Conclusion

In summary, a pet-care IoT solution "Petification" that can remotely take care of pets is proposed. The IoT platform for Petification is implemented using Node-RED and Eclipse Mosquitto MQTT message broker to take advantage of open-source. The water supplier is supported in the Petification to supply water to the pet and report the current amount of the remaining water. The feed machine is also supported to serve a certain amount of food to the pet and report the current amount of the remaining food. Web-based dashboard is provided to allow the user to track the consumption, remaining amount, and connectivity of each device. Users can also receive notification by email or "WhatsApp" messenger when the food or water of the devices is running out. Current limitation of Petification is the relatively low accuracy in food serving, due to the design of the food gate and sensitivity of the load cell. Thus future plans involves adressing these issues to improve accuracy of the serving amount. Improving funcitonality could also be improved by adding different devices such as cameras so that users can visually track the pets.

VII. Acknowledgement

"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 Planing & Evaluation) in 2021"(2021-0-01435) The authors of this study are grateful to Professor Minsun Lee of Chungnam National University, Professor Eric T. Matson, Professor Anthony H. Smith, and Teaching Assistant Minji Lee of Purdue University for helping us participate in the project.

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] MQTT [Online]. Available: https://mqtt.org

[9] 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.

[10] 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/

[11] 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.

[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