# Precision Apiculture – IoT System for Remote Monitoring of Honeybee Colonies

Riste Poposki
*Ss. Cyril and Methodius University*
*Faculty of Computer Science and Engineering*
*Skopje, Macedonia*
riste.poposki@students.finki.ukim.mk

Dejan Gjorgjevikj
*Ss. Cyril and Methodius University*
*Faculty of Computer Science and Engineering*
*Skopje, Macedonia*
dejan.gjorgjevikj@finki.ukim.mk

***Abstract*— Beekeeping practice, being very environmentally dependent, requires the temperature and the humidity in the hive to be in some regular ranges for optimal beehive health and productivity. Since most of the plants and flowers required for beehive prosperity and honey production are usually outside inhabited areas, the beekeeper must travel to the bee colonies to check them, which can be time and resource consuming. In this paper, an end to end remote monitoring and control system for a bee colony is presented. The system is consisted of a web-based system for monitoring and control of the conditions of the hives and IoT system for collecting the sensor measurements and transferring the data. The IoT system is composed of hardware units that are mounted on the beehives, containing temperature, humidity, weight sensors, actuators, and a microcontroller responsible for collecting the measurements and sending the data to the web system. The communication between the hardware unit and the web system uses WiFi or LoraWAN technology, that enables running the device on batteries. The system enables remote monitoring of multiple beehives and can be configured to alert the user via email or push notification if some sensor value is outside of predefined range. The system also enables sending commands to the unit controlling the actuators that can intervene on the beehive closing or opening a ventilation lid.***

***Keywords—beekeeping, IoT, monitoring, LoraWAN, WiFi***

## I. Introduction

The constant progress in the area of internet of things contributes to creating various systems for remote monitoring and control in different areas like agriculture [1], the health sector, smart cities, smart homes, transport etc. The apiculture or beekeeping is organized care and management of bee colonies in human made hives [2].

For the beehive prosperity and successful production of honey, it is very important the hive's internal and external environmental conditions to be in some regular ranges. According to [3] and [4], the hive's internal temperature and humidity are one of the main factors for increased honey production. Honeybees have some natural ways of regulating the temperature and humidity but despite that, sometimes an intervention from the beekeeper is required in order to balance the values. On the other hand, the bee colonies are usually placed outside of areas inhabited with people, preferably in the rural areas and most of the time the beekeeper is not living near the beehives. The threats of animal attack on the hives, as well as possibility of other people stealing the hives are common. Also, when the period for harvesting the honey comes, the beekeeper must manually inspect all the hives and check which ones are ready.

Several monitoring systems that would contribute to the beekeeping process and increase the honey production have been suggested by different researchers. Balta and Dogan in [5] show example of a software architecture for a system that monitors the internal conditions (temperature, humidity) in a beehive. In a similar study [6] Dineva and Atanasova propose a system for monitoring the conditions with focus on different communication technologies for transferring the sensor values. Despite monitoring the internal conditions of the beehive, the authors in [7] propose adding a weight scale to periodically measure the beehive weigh. This weight values help in recognizing which hive has honey to be gathered, or if the hive has been stolen or destroyed by animals.

The aim in this paper was to develop an end to end system for remote monitoring and control of a beehive colonies. An embedded hardware unit with connected sensors would be mounted on the beehives and will periodically send data to a web system. The beehive state including the temperature, humidity and weight of the beehive can be monitored using a web application. The system also provides an option for sending commands to the beehive like changing the data transfer interval. The system can be configured to send alerts to the user if some sensor value is out of range. Special focus will be given to low powered wide area communication technologies like LoraWAN [8] since the device should be able to operate on batteries.

The remainder of the paper is organized as follows: in Section 2 the system architecture is described, with subsections about the general structure, the detailed structure, as well as explanation of the communication between components. In Section 3 the initial implementation of the web system and the prototype of the hardware unit is presented and discussed. Finally, in Section 4 a conclusion and remarks on the future work is given.

## II. System Architecture

The architecture and intercommunication of the components of the beehive monitoring system should be carefully thought out since the whole system should meet some important requirements such as:

- The user should access the system from any device and from anywhere where internet access is available
- The data from the hives should be available near real time, with possibility to notify the user of some critical conditions
- Many areas where the beehives reside do not have electricity so the hive monitoring system should be operational in these areas
- Operation of the hive monitoring system should not be dependent on constant internet access
- Monitoring the conditions in the hives without altering the natural workflow of the bees, meaning the system should be minimally invasive

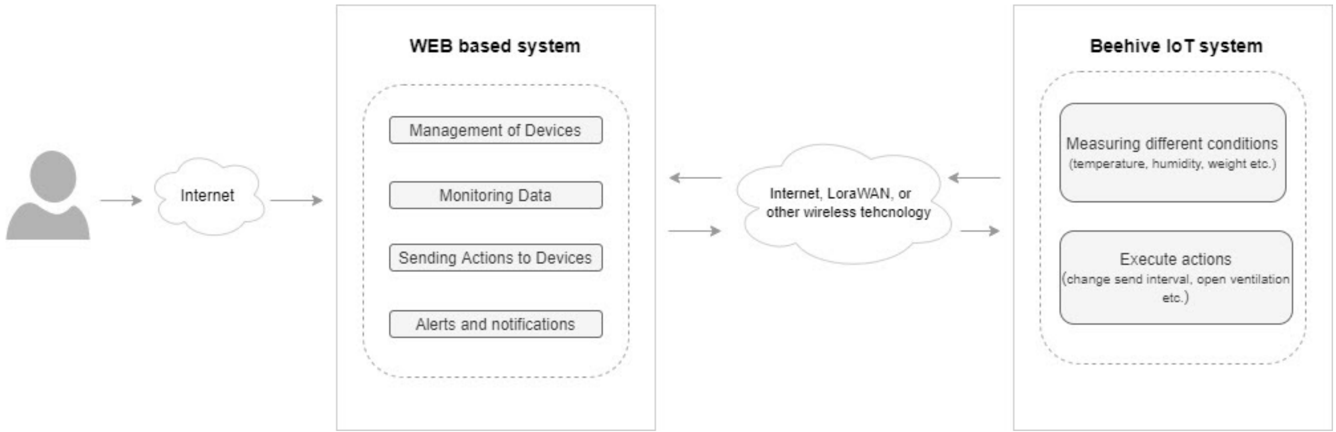


Fig. 1.General structure of the system

### A. General Structure

Based on these requirements, the general system architecture is presented on Fig. 1. The beehive monitoring and control system is composed of two subsystems: a web-based system and an IoT system. The web-based system is hosted on a web server, while the IoT system is placed onsite where the actual beehives are located.

The web system is the main entry point for user interaction. To access and use this system the user only needs to have a device with internet connection and a web browser. Generally, it can be perceived like a traditional rich web application hosted on a web server. Upon visiting the application, the user can interact with the beehive monitoring and control system. Some of the functionalities and modules are:

- Monitoring the beehives' data. This includes viewing the sensor values sent from the devices mounted on the beehives. A tabular and graphical presentation of the data is available.
- Listing and managing all the devices mounted on the beehives. This includes registering new devices, editing their basic information, and adding the device on a dashboard.
- Sending commands to the devices. This is a module where the user can send commands to a device mounted on a particular beehive. The commands can be used to configure the sampling rate and the data transfer interval, or to control actuators as servo motors to increase the ventilation in the beehive. There is also a history view where the status of whether the device executed the command is presented.
- Setting rules and alerts. In this module the user can define rules regarding the sensor values of each beehive. If the rule is valid, the system can send an alert to the user via email or push notification.

The IoT system is composed of the beehives that have smart device mounted on it. The smart device contains a general-purpose microcontroller (MCU) connected to various sensors for measuring some physical parameters of the environment. Some of the sensors monitor the internal conditions inside the beehive and some of them the external conditions around the beehive.

The sensor values are periodically sent to the web-based system on a predefined period. The primary set of parameters that are measured are:

- temperature
- humidity
- pressure
- beehive weight

If the device has WiFi access to the internet, the data can be sent directly to the web system using the HTTP transfer protocol. However, considering the mentioned requirements, constant power supply or internet connection are not always guaranteed. LPWA (Low Power Wide Area) technology like LoraWAN can be very suitable for transferring slow changing data on long ranges using very low power. Devices with LoRa connectivity can generally send long range data to gateways in a 2-5km radius while running on batteries for an extended period of couple of months. Currently we use the LoraWAN wireless technology for transferring the data to the web-based system, but other LPWA technologies can be integrated to the system in the future.

Despite only sending data to the web-based system, the IoT System can also receive some predefined commands from the user. Some of the commands are:

- Changing the interval between taking measurements and data transfers. Most of the time the smart device can remain in low power mode saving the batteries. The most power is used while reading the sensors and transferring the data back to the web-based system.
- Controlling actuators like servo motors to physically move some parts of the beehive. An example can be opening a lid in order to decrease the humidity in the beehive.

Each command is followed by an acknowledgement offering an insight in the commands that have not been delivered, are not executed yet on the device, or have failed.

### B. Detailed Structure

The detailed view of all the components, the external systems and communication between them is given on Fig. 2.

The internal components of the beehive monitoring and control system is composed of web client application, web

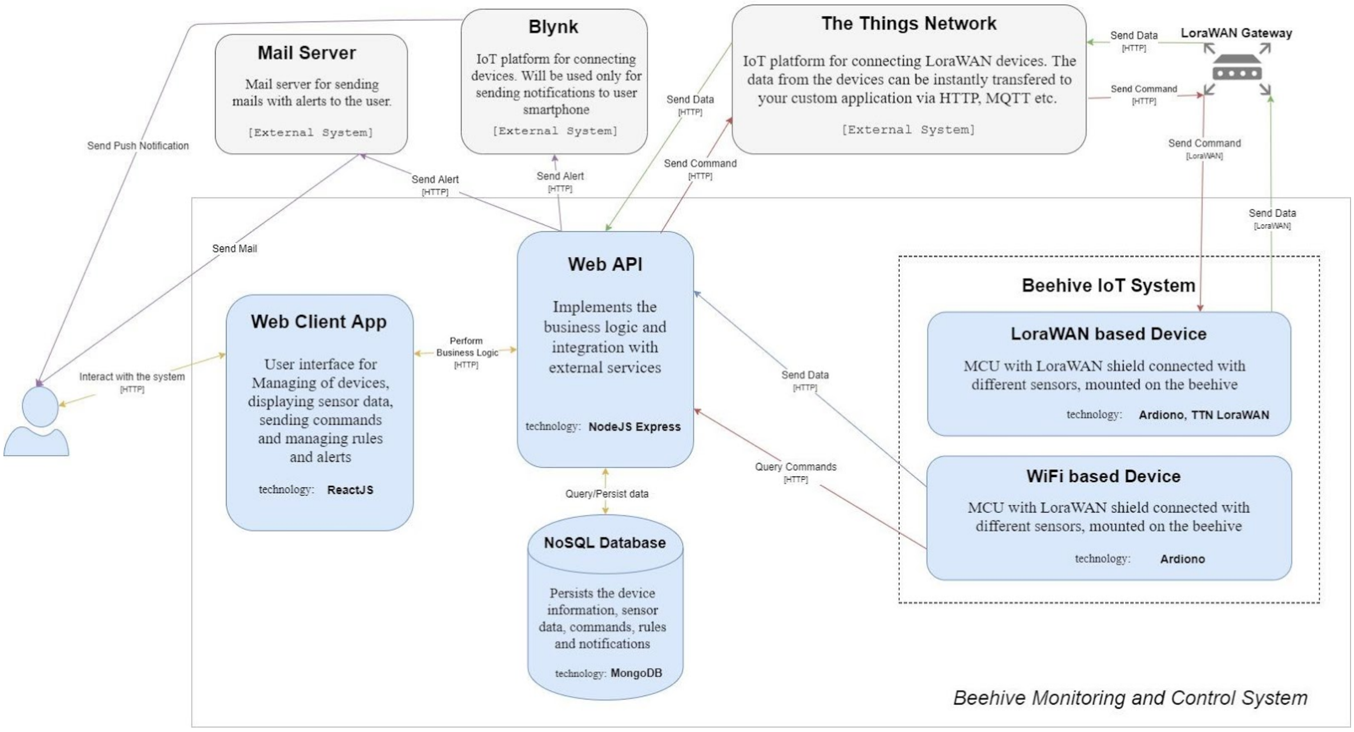

Fig. 2. Detailed structure of the system

API, NoSQL database and IoT devices mounted on the actual beehives.

To implement the LoraWAN support, the system is integrated with the "The Things Network" cloud service [9] while for the alerting module, integration with a mail server and the Blynk IoT platform [10] is used. These are third party external services that provide some functionalities that do not need to be implemented inside the beehive monitoring and control system.

*1) Web Client App*

This is a single page web application developed using HTML, CSS and the ReactJS framework. It implements the user interface of the system, and exposes the following modules and functionalities:

- Beehive Dashboard - a module where the user can view summary information for the beehives. It includes a section showing the corresponding beehive's minimum, maximum and average sensor values. Underneath this section there is a beehive section where each beehive is represented as a donut chart displaying some sensor values. The last section is a detailed view for the selected beehive revealing all sensor values in detail and displaying a line chart with historical data.
- Control Dashboard - a module where the user can view compact information for some beehives of interest. The view consists of section displaying the basic information of the beehive, section displaying the sensor data as a table and a line chart and section displaying the commands that can be sent to the device.
- Devices Module - a module where all registered devices (beehives) in the system are listed. Some functionalities in this module include creating new devices or adding existing devices to the control dashboard.
- TTN Integration Module - a module where the integration with The Things Network is configured. The user can view all the registered devices on the network and can add new ones. For every TTN device the corresponding beehive device is presented, or if no device is assigned a connection to a device can be configured.
- Alerts Module - a module that is shown in the device detailed view is used for defining rules for each sensor. The predefined rules are:
  - value greater than - the system will send alert if the sensor value is greater than user specified value.
  - value lower than - the system will send alert if the sensor value is lower than user specified value.
  - maximum time without data transfer - the system will send alert if the device has failed to send data for more than prespecified amount of time.

*2) Web API*

This is a backend RESTful web API developed using the NodeJS express framework. It is a central place where the business logic of the system resides. Each independent module is written in a separate endpoint. The following endpoints are available:

- Device Endpoint - This is an endpoint for managing devices. It has CRUD methods for the beehive devices. There is also a method for changing the connected TTN device.
- Data Endpoint - This is an endpoint for the sensor data. It has methods for posting sensor values from the WiFi enabled devices as well as methods for getting and filtering sensor data for some specified device.

- Command Endpoint - This is an endpoint for the device commands. It has methods for sending a selected command to a specified device as well as getting the already sent commands (history). There is also a method for getting only the not acknowledged commands for a particular device. This method is used by the beehive devices that pool the server on a predefined interval.

- Alerts Endpoint - This is an endpoint for managing the rules and alerts for each device. There are methods for creating alerts for certain sensors, as well as defining the rules when this alert will be activated. There is also alerts history endpoint where the last sent alerts with the triggered rules can be retrieved.

- Beehive Summary Dashboard Endpoint - This endpoint offers data summary retrieval for the beehive dashboard.

- TTN Endpoint - This is an endpoint for managing the devices registered on The Things Network. There are CRUD methods for managing each TTN device and a method for getting the TTN application info.

Beside the REST endpoints, the Web API has integration with The Things Network's data endpoint via the MQTT (Message Queue Telemetry Transport). This is the place where actual communication with the LoraWAN devices mounted on the beehives is happening. When a beehive transfers data to the TTN cloud service, the same data gets decrypted and sent to the Web API to be stored in the database.

There is also another integration with the Blynk IoT cloud for sending alerts via push notification and with the google mailing server for sending alerts via e-mail.

*3) NoSQL Database*

The database used by the Web API is NoSQL Mongo database. The data is persisted and queried using the mongoose library for NodeJS which is an ORM (Object Relational Mapper) that eases the whole data related process. The models (entities) created for the system are device, data, command, alert, alert history, and beehive summary dashboard.

*4) Beehive IoT Devices*

The beehive IoT devices are consisted of a microcontroller and sensors connected to it, programmed on the Arduino platform. These devices are mounted on the beehive and measure the conditions inside and outside the hive. Two types of devices have been used:

- Lora32U4 - a development board based on ATmega32u4 microcontroller with a built in Lora 868 MHz Radio module. This module has support for battery-based power supply which can last for a couple of months. On this board, the following sensors and actuators are connected: DHT11 (temperature and humidity sensor), SG90 Micro Servo (minimal servo motor), HX711 (analog to digital converter module connected to 4 strain gauge sensors composing a weight scale). The MCU is periodically reading the sensor values and is transfers the readings using the LoraWAN technology. The data transfer is a broadcast to all accessible TTN gateways nearby. The gateways are transferring the data to the TTN cloud service via the internet. The TTN cloud then sends this data to the web API which saves it in the database making it available to the user. After each transmission there is a small window where the TTN cloud can send some downlink message to the device. This window is used for sending commands to the device.

- NodeMCU ESP8266 - a development board based on the ESP8266 WiFi module. On this board, a BMP280 temperature and pressure sensor is connected. The sensor values are transferred to the web API periodically on a certain interval via HTTP to the data endpoint. The commands are queried via pooling i.e. sending HTTP request on a certain period and checking the command endpoint for commands waiting to be executed on that device.

*5) External Systems*

The external systems used in this solution include The Things Network cloud service, the Blynk IoT platform and a mail server.

The Things Network is a global public network offering easy integration for the devices that support the LoraWAN protocol. In order to be used, one needs to create a TTN application in the TTN service, after which the registration of the devices can be performed. These devices are called TTN devices, and as described in the previous sections, they can be connected to the already registered devices in the beehive system. For the full communication to work, a gateway is required to be in the reach of the device. A gateway is a hardware device which has permanent internet and power connection equipped with software for capturing the incoming LoraWAN messages and dispatching them via internet to the TTN Network.

The Blynk IoT platform is a large and multi feature platform for developing and implementing different IoT solutions, but in our application only the notification module is used. The user needs to have the Blynk mobile app installed, which enables receiving push notifications from the web application. Whenever an alert is triggered in the system, a notification including information about the alert that was triggered and the device that triggered it is received on the user's smartphone.

For the mail integration, any SMTP server can be used. We have used the Google's Gmail SMTP server, that upon triggering an alert, an email message including the information about the device and the rules that triggered the event is composes and sent to the configured e-mail address.

### *C. Communication between components*

The user starts the interaction with the system from his web browser visiting the web client application. The client application requests resources from the web API using the HTTP protocol. These resources include information for the devices, the actual sensor values, commands history etc. The web API validates the requests and executes some business logic to load the data from the MongoDB database. The payload is returned in JSON format.

The IoT devices can use different communication protocols based on the actual device. The WiFi based devices send their data and query the available commands with HTTP directly with the web API. The LoraWAN based devices use LoraWAN protocol and communicate with the nearby gateways. The gateways than transfer the data to the TTN cloud service, that sends that data further to the web API using

MQTT protocol. The commands for the LoraWAN devices are first sent from the web API to the TTN cloud service that queues all commands and sends them to the appropriate gateways that forward them to the actual LoraWAN devices on their next data transfer.

## III. Prototype And Initial Implementation

The beehive monitoring and control system described in the previous section has been partially implemented and prototyped.

### A. *Software System*

The beehive dashboard module is shown on Figure 3. The first section shows the total number of beehives monitored and the user can select which beehives and which sensors are shown on the page. Next to it is the Min-Max-Avg section showing the information for each sensor. Underneath this is the devices section showing donut gauges for each device. The data shown here can be selected from the Min-Max-Avg section. At the bottom of the screen the device detail section is shown where all the data values for the selected device are shown, as well as line chart showing the variation of the values for the selected sensor.

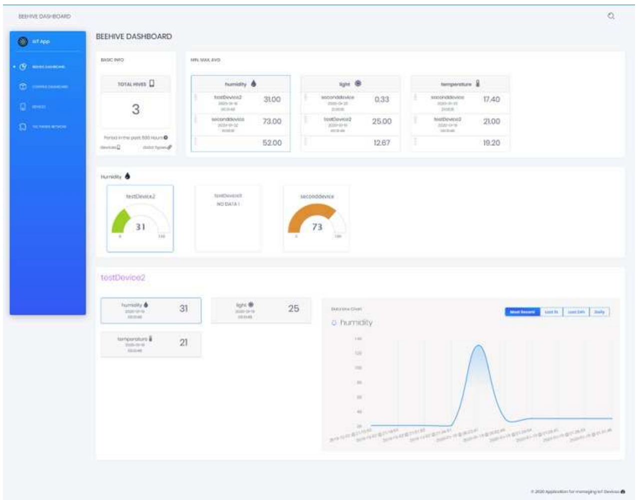


Fig. 3. Beehve dashboard from the Beehive System

### B. *IoT System*

A photo of the principal prototype of the beehive device is shown on Figure 4. It shows the Lora32U4 development board connected to the sensors and actuators on a breadboard. The connected sensors are:

- DHT 11 temperature sensor
- LDR light sensor
- HX711 module connected to a weight sensors
- SG90 Micro Servo actuator
- LED diode for status signaling

The MCU is programmed using the Arduino platform and its main cycle is reading the values from the sensors and sending them to the TTN network. The data transfer interval is initially set to 3 minutes based on the TTN fair access policy. After each data transfer the device checks for possible messages sent back from the TTN network in the downlink window. If downlink message has been received, the MCU decodes the command and executes it. It can be for example moving the servo actuator to open/close a ventilation lid or to control another device.

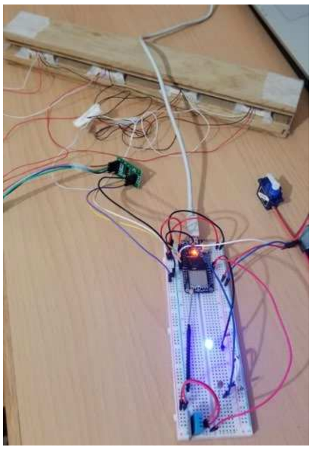

Fig. 4. Prototype of the beehive sensor device

## IV. Conclusion

In this paper, an end to end system for monitoring and control of a beehive colonies is presented. It contains a web-based solution that exposes a web application where the user monitors and controls the conditions of the bee colonies. The application shows different graphical and tabular representations of the sensor values for each beehive, beehive groups, summary view of the minimum, maximum and average values across all available beehives. There is also possibility to define rules for beehive parameters that enable sending alerts via mail or push notification. The other part of the system is the hardware IoT units containing sensors. The sensors measure some crucial parameters for the beehive like the internal and external temperature and humidity and the beehive weight. These values are sent via the LoraWAN or WiFi connection to the web system where they are persisted. Such system will enable the beekeeper to have near real-time information about the state of his bee colonies, reducing the need for travel to physically inspect the beehives.

For the future work, this prototype will be mounted on a real beehive adequately protected from the external influences (rain, sun) and we plan to perform experiments with real beehive colonies. Use of additional sensors that can give some insight of intensity of the bee activity, like microphones and light sensors that can count the bees that leave or enter the hive will also be considered. Extending the system to support other LPWAN technologies like NB-IoT or SigFox and even the GPRS network communication is also planned. As for the web-based system, a standalone mobile application can be developed and integrated with the web API. This will eliminate the need for the integration with the Blynk platform and will also provide easier way for the user to interact with the system through its mobile device.

## Acknowledgment

This research was partially funded by the Faculty of Computer Science and Engineering, Ss. Cyril and Methodius University in Skopje.

## REFERENCES


[1] Z. K. Aldein Mohammeda and E. S. Ali Ahmed, “Internet of Things Applications, Challenges and Related Future Technologies”, World Scientific News, vol. 67, No. 2, pp. 126-148, 2017.

[2] S. E. McGregor, “Beekeeping”, Encyclopaedia Britannica, https://www.britannica.com/topic/beekeeping, [Accessed March 2020].

[3] E. E. Southwick, G. Heldmaier, “Temperature Control in Honey Bee Colonies”, BioScience, vol. 37, no. 6, pp. 395–399, June 1987.

[4] H. F. Abou-Shaara, A. A. Al-Ghamdi, A. A. Mohamed, "Tolerance of two honey bee races to various temperature and relative humidity gradients", Environmental and Experimental Biology, vol. 10, pp. 133–138, 2012.

[5] A. Balta, S. Dogan, G. O. KOCA and E. Akbal, “Software Modeling of Remote Controlled Beehive Design”, International Conference on Advances and Innovations in Engineering (ICAIE), Elazig, Turkey, May 2017.

[6] K. Dineva and T. Atanasova, “Model of Modular IoT-based Bee-Keeping System”, The 2017 European Simulation and Modelling Conference, Lisbon, Portugal, October 2017.

[7] S. Gil-Lebrero, F.J. Quiles-Latorre, M. Ortiz-López, V. Sánchez-Ruiz, V. Gámiz-López, J.J. Luna-Rodríguez, “Honey Bee Colonies Remote Monitoring System”, Sensors, vol. 17, no. 55, 2017.

[8] “LoRaWAN 1.1 Specification”, LoRa Alliance Technical Committee, October 11, 2017.

[9] “The Things Network”, [Online] https://www.thethingsnetwork.org/, Accessed: March 2020.

[10] “Blynk IoT”, [Online], https://blynk.io/, Accessed: March 2020.