Wireless Communication Over LoRaWAN Using STM32 Microcontroller

Submitted by:

Aleena Aamir (408538)
Iman Fatima (420537)
Iqra Shafique (421752)
Syed Suleman Ali (410541)
Talha Razzaq (418013)
Wahab Sohail (418159)

Supervised by:

Dr.Sagheer Khan

Ms.Ayesha Khanum

Date of Submission:
December 30, 2024

College of Electrical and Mechanical Engineering, NUST

1 Abstract

This project proposes a LoRaWAN-based wireless communication system using the STM32 microcontroller for data collection and transmission. With leverage of long range, long power feature of LoRaWAN, this system addresses the limitations of traditional wireless technologies in embedded applications. The system consists of sensors, LoRaWAN gateway and cloud platform for real-time environmental monitoring.

2 Problem Statement

Traditional wireless devices face challenges in transmitting data over long ranges in real-time, while also maintaining reliability and energy efficiency. For applications requiring remote monitoring, scalable and efficient communication systems are crucial to enable accurate and timely data collection.

3 Objective

The primary objectives of this project are:

- Develop a sensor node integrated with STM32 to collect environmental data.
- Implement a LoRaWAN transceiver module to wirelessly transmit sensor data.
- Configure the LoRaWAN gateway to send data in the cloud for visualization in real time.

4 Literature Review

4.1 Introduction to LoRaWAN and STM32 Nucleo WL55JC1

For low-power long-range communication in Internet of Things applications, LoRaWAN (Long-Range Wide Area Network) is a commonly used protocol. It is an affordable option for setting up scalable wireless networks because it can function in unlicensed frequency ranges. A development board called the STM32 Nucleo WL55JC1 incorporates the STM32WL55 microcontroller, which has an inbuilt LoRa transceiver and an ARM Cortex-M4 core. This integration makes LoRaWAN-based systems smaller and easier to construct, which makes them perfect for uses like industrial IoT, smart agriculture, and environmental monitoring.

4.2 Capabilities of STM32 Nucleo WL55JC1 in LoRaWAN Applications

Several benefits are provided by the STM32 Nucleo WL55JC1 for LoRaWAN-based projects:

Integrated LoRa Transceiver: By eliminating the requirement for an external LoRa module, the integrated LoRa transceiver reduces power consumption and hardware complexity.

Dual-Core Architecture: To enable optimal resource allocation, the STM32WL55 has a Cortex-M4 core for high performance activities and a Cortex-M0+ core for energy-efficient operations.

Low Power Consumption: The STM32WL55 is suitable for battery-operated devices due to its sophisticated low-power modes.

Adaptable Development Environment: Pre-certified LoRaWAN stacks are provided, and development is streamlined by the STM32CubeWL software package and the STM32CubeMX configuration tool.

4.3 Integration of STM32 Nucleo WL55JC1 with Environmental Monitoring Systems

The STM32 Nucleo WL55JC1 will be used in the creation of a wireless environmental monitoring system in the proposed project. The viability and advantages of such integration are highlighted in pertinent literature:

Sensor Integration: To interface with environmental sensors, including temperature, humidity, and air quality sensors, the STM32WL55 supports a number of communication protocols (I2C, SPI, and UART). Research indicates that pre-processing sensor data on the STM32WL55 (such as threshold-based warnings or noise filtering) lowers the transmission load on the LoRaWAN network.

Communication using LoRaWAN: The LoRaWAN stack offered by the STM32CubeWL package complies with the LoRa Alliance requirements, guaranteeing compatibility with common LoRaWAN gateways. Research highlights how crucial it is to set up an adaptive data rate (ADR) to maximize power usage and communication range.

Gateway and Cloud Integration: Sensor data is sent to cloud platforms through LoRaWAN gateways for real-time storage and visualization. AWS IoT, ThingsBoard, and The Things Network (TTN) are well-known systems. The STM32WL55's ability to transmit data in JSON format makes integrating with cloud services easier.

4.4 Applications of STM32 Nucleo WL55JC1 in Similar Projects

The STM32 microcontroller is widely used in numerous projects, some of which are briefly discussed here:

Environmental Monitoring The STM32WL55 has been utilized in projects for tracking weather and air quality in both urban and rural areas. Its built-in LoRa transceiver ensures reliable, long-distance data transmission.

Smart Agriculture The STM32WL55 has been utilized in precision agriculture systems to measure temperature, light intensity, and soil moisture. Its low-power operation extends the battery life of field-deployed nodes

Industrial IoT In industrial environments, the STM32WL55 is used to monitor equipment health by transmitting temperature and vibration data to cloud platforms for predictive maintenance.

5 Methodology

5.1 Project Planning and Requirement Analysis

Objective Definition: The project aims to design a wireless communication system leveraging the LoRaWAN protocol and STM32 microcontroller to achieve low-power, long-range communication suitable for IoT applications.

Requirement Analysis: Hardware components needed for the system:

- STM32 WL55 microcontroller.
- Integrated LoRaWAN module.
- Additional peripherals like sensors for data collection.

Software Tools: Chose STM32Cube IDE for firmware development and LoRaWAN libraries for protocol implementation.

5.2 System Design

Architecture Design: Developed a high-level system architecture outlining components such as:

- STM32 WL55 microcontroller for data processing and control.
- LoRa module for long-range wireless communication.
- Sensors to collect data.
- Gateway to transmit data to a central server or cloud.

Protocol Selection: Configured the LoRaWAN protocol for wireless communication over a wide area.

5.3 Software

Firmware Development:

- Used the STM32Cube IDE to write and debug the firmware for the STM32 WL55 microcontroller.

- Configured LoRaWAN libraries for communication, including setting parameters like frequency, data rate, and payload size.

Sensor Data Processing:

- Coded to collect the data from sensors, DHT11 and IR, and pre-process before transmission.

LoRaWAN Communication:

- Implemented the LoRaWAN protocol to establish a connection with the gateway.
- Designed functions to encode data, send packets, and handle acknowledgments or retransmissions.

Application Server

- An application server was integrated with The Things Network (TTN) to display real-time data on a dashboard for efficient monitoring.

5.4 Hardware

Component Selection:

- Chose the STM32 WL55 microcontroller with integrated LoRa module for data extraction and transmission.
- Selected an integrated LoRa module for wireless communication.

Circuit Design:

- Designed the circuit to interface the STM32 microcontroller with the LoRa module and other peripherals.
- Power the system using the Serial wire instead of an external battery or external power source.

5.5 Integration

System Integration:

- Combined hardware and software components into a unified system.
- Tested the communication between the STM32 microcontroller and the LoRa module by performing data transmission.

Gateway Configuration:

- Set up the LoRaWAN gateway to receive and forward data to a network server or cloud.

6 System Implementation

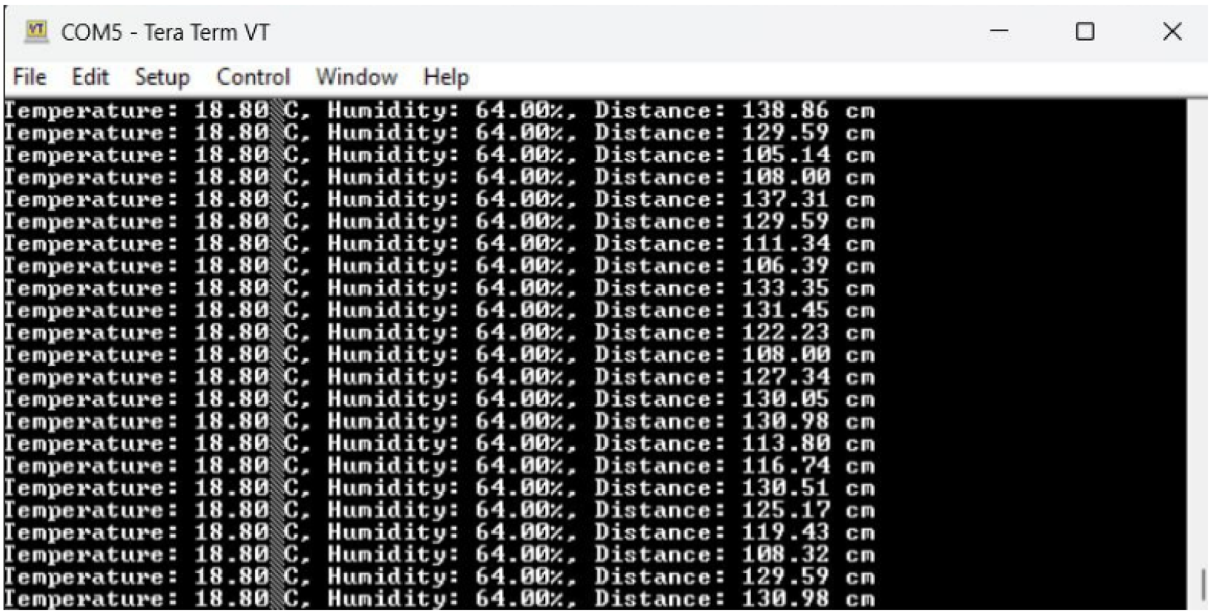
The desired system is implemented step by step, clearly explained below:

6.1 Sensor Data Acquisition

In the very first step, data from sensors is collected over time, by using some built-in commands, to store in variables specified before. The two different sensors- DHT11 and IR, are used to get the Temperature and Humidity of the surroundings, and the distance of object ahead of the IR sensor, respectively, by using a separate function and then displaying them on a console window for user to cross-examine it first.

```
DHT11_Read(&temperature, &humidity);
    distance = Infrared_ReadDistance();

    printf("Temperature: %d C, Humidity: %d%%, Distance: %d
        cm\n", temperature, humidity, distance);
```

The data above is received on the console as following:



```
Temperature: 18.80 C, Humidity: 64.00%, Distance: 138.86 cm
Temperature: 18.80 C, Humidity: 64.00%, Distance: 129.59 cm
Temperature: 18.80 C, Humidity: 64.00%, Distance: 105.14 cm
Temperature: 18.80 C, Humidity: 64.00%, Distance: 108.00 cm
Temperature: 18.80 C, Humidity: 64.00%, Distance: 137.31 cm
Temperature: 18.80 C, Humidity: 64.00%, Distance: 129.59 cm
Temperature: 18.80 C, Humidity: 64.00%, Distance: 111.34 cm
Temperature: 18.80 C, Humidity: 64.00%, Distance: 106.39 cm
Temperature: 18.80 C, Humidity: 64.00%, Distance: 133.35 cm
Temperature: 18.80 C, Humidity: 64.00%, Distance: 131.45 cm
Temperature: 18.80 C, Humidity: 64.00%, Distance: 122.23 cm
Temperature: 18.80 C, Humidity: 64.00%, Distance: 108.00 cm
Temperature: 18.80 C, Humidity: 64.00%, Distance: 127.34 cm
Temperature: 18.80 C, Humidity: 64.00%, Distance: 130.05 cm
Temperature: 18.80 C, Humidity: 64.00%, Distance: 130.98 cm
Temperature: 18.80 C, Humidity: 64.00%, Distance: 113.80 cm
Temperature: 18.80 C, Humidity: 64.00%, Distance: 116.74 cm
Temperature: 18.80 C, Humidity: 64.00%, Distance: 130.51 cm
Temperature: 18.80 C, Humidity: 64.00%, Distance: 125.17 cm
Temperature: 18.80 C, Humidity: 64.00%, Distance: 119.43 cm
Temperature: 18.80 C, Humidity: 64.00%, Distance: 108.32 cm
Temperature: 18.80 C, Humidity: 64.00%, Distance: 129.59 cm
Temperature: 18.80 C, Humidity: 64.00%, Distance: 130.98 cm
```

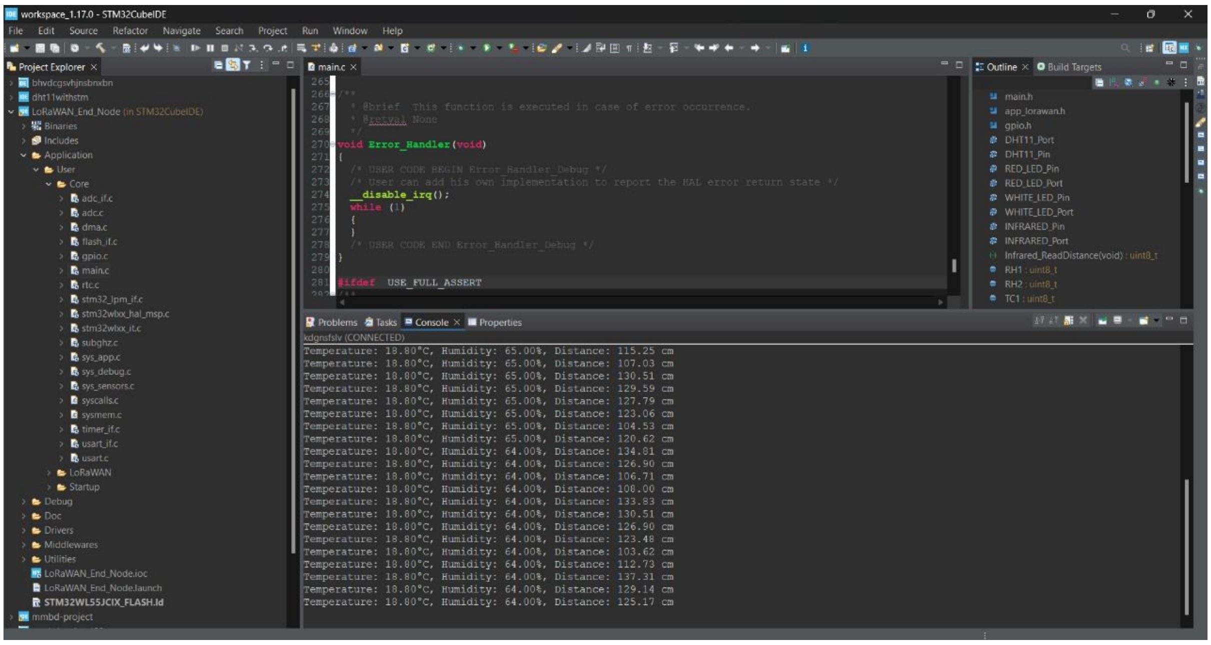
Figure 1: Sensor Output on Tera Term.

6.2 Gateway Communication

For Gateway communication, Dragino gateway will be used and it will be configured on a PC connected to the same network as the Dragino gateway. Here are the detailed steps we followed:

6.2.1 Accessing the Configuration Interface

- Connect the Dragino gateway to power and the Internet.
- Access the Dragino gateway's configuration interface by entering its IP address in a web browser on the PC.



```
/**
  * @brief  This function is executed in case of error occurrence.
  * @retval None
  */
void Error_Handler(void)
{
  /* USER CODE BEGIN Error_Handler_Debug */
  /* User can add his own implementation to report the HAL error return state */
  __disable_irq();
  while (1)
  {
  }
  /* USER CODE END Error_Handler_Debug */
}

#ifdef  USE_FULL_ASSERT
```

```
kdgnsfslv (CONNECTED)
Temperature: 18.80°C, Humidity: 65.00%, Distance: 115.25 cm
Temperature: 18.80°C, Humidity: 65.00%, Distance: 107.03 cm
Temperature: 18.80°C, Humidity: 65.00%, Distance: 130.51 cm
Temperature: 18.80°C, Humidity: 65.00%, Distance: 129.59 cm
Temperature: 18.80°C, Humidity: 65.00%, Distance: 127.79 cm
Temperature: 18.80°C, Humidity: 65.00%, Distance: 123.06 cm
Temperature: 18.80°C, Humidity: 65.00%, Distance: 104.53 cm
Temperature: 18.80°C, Humidity: 65.00%, Distance: 120.62 cm
Temperature: 18.80°C, Humidity: 64.00%, Distance: 134.81 cm
Temperature: 18.80°C, Humidity: 64.00%, Distance: 126.90 cm
Temperature: 18.80°C, Humidity: 64.00%, Distance: 106.71 cm
Temperature: 18.80°C, Humidity: 64.00%, Distance: 108.00 cm
Temperature: 18.80°C, Humidity: 64.00%, Distance: 133.83 cm
Temperature: 18.80°C, Humidity: 64.00%, Distance: 130.51 cm
Temperature: 18.80°C, Humidity: 64.00%, Distance: 126.90 cm
Temperature: 18.80°C, Humidity: 64.00%, Distance: 123.48 cm
Temperature: 18.80°C, Humidity: 64.00%, Distance: 103.62 cm
Temperature: 18.80°C, Humidity: 64.00%, Distance: 112.73 cm
Temperature: 18.80°C, Humidity: 64.00%, Distance: 137.31 cm
Temperature: 18.80°C, Humidity: 64.00%, Distance: 129.14 cm
Temperature: 18.80°C, Humidity: 64.00%, Distance: 125.17 cm
```

Figure 2: Sensor Output on Cube IDE Console.

6.2.2 Setting the Frequency Plan

- Select the AUS 915-928 MHz frequency plan because it was the most suitable for application.

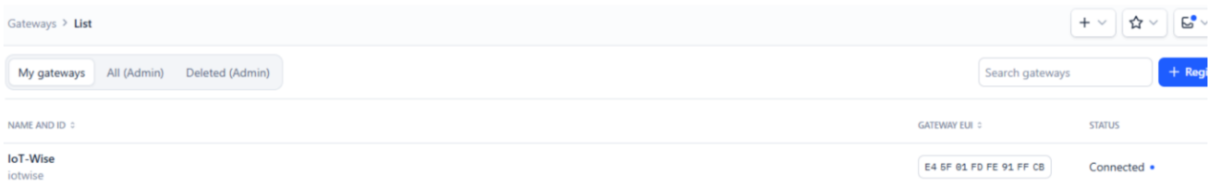
6.2.3 Connecting to The Things Network (TTN)

- Configure the gateway to connect to The Things Network by setting the server address to:

agriculturelorawan.au1.cloud.thethings.industries.

6.2.4 Registering the Gateway on TTN

- Create an account on The Things Network console.
- Register the Dragino gateway by providing its EUI (a unique identifier for the gateway).
- Verify that the gateway status showed as 'connected' on the TTN console.

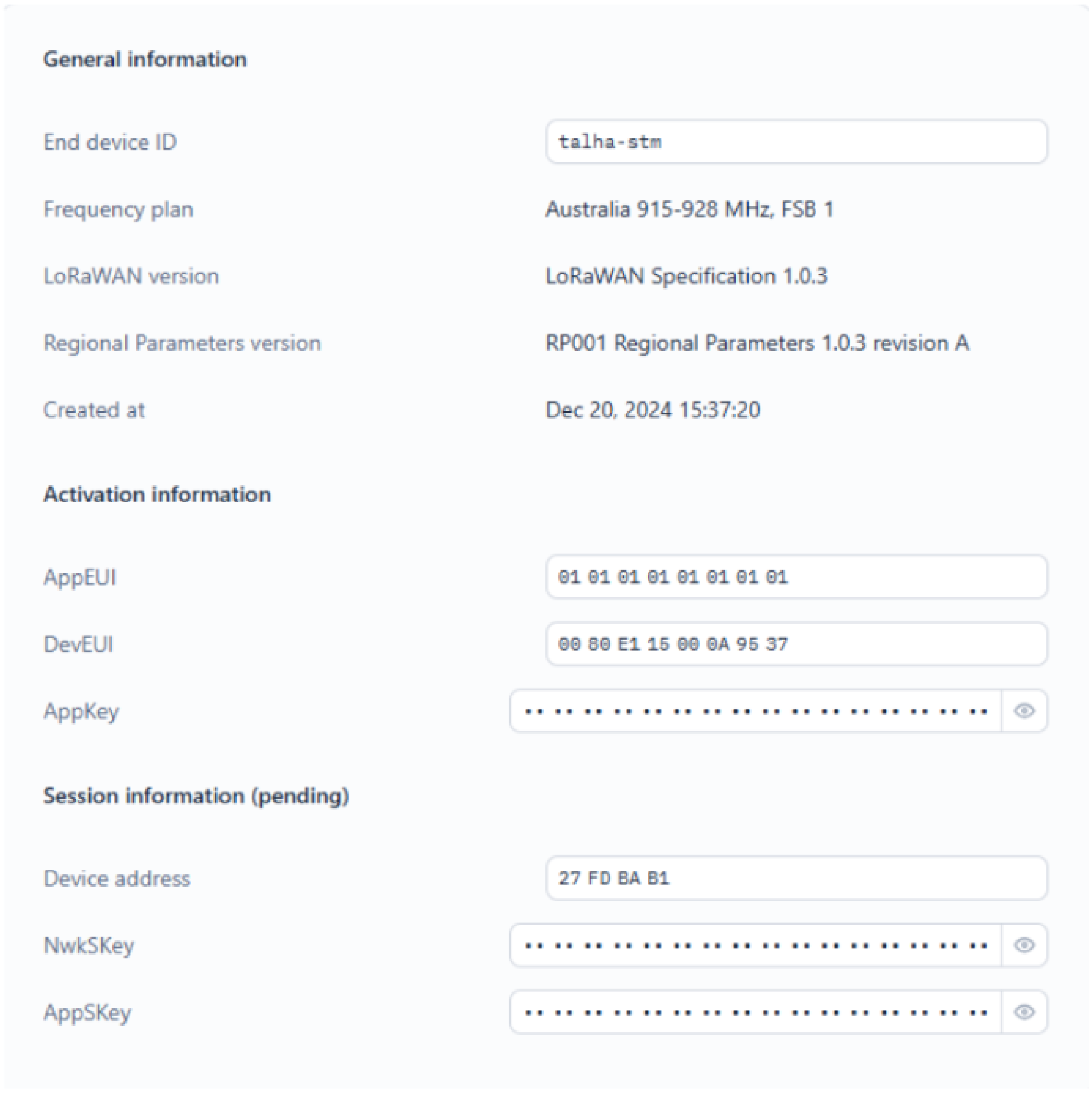


NAME AND ID	GATEWAY EUI	STATUS
IoT-Wise iotwise	E4 5F 01 FD FE 91 FF CB	Connected •

Figure 3: Building the Connection on Gateway

6.3 Creating an application to register an End Device

After configuring the gateway, now create an application on The Things Network (TTN) where one can register their end device. Once the application is created, one needs to register their end device within that application. Since STM32 Nucleo-WL is being used here, its credentials will be used to configure the end device within the application on the TTN console. Then, the information such as frequency plans, APP EUI, DEV EUI, APP KEY, and device address will be set.



General information

End device ID	talha-stm
Frequency plan	Australia 915-928 MHz, FSB 1
LoRaWAN version	LoRaWAN Specification 1.0.3
Regional Parameters version	RP001 Regional Parameters 1.0.3 revision A
Created at	Dec 20, 2024 15:37:20

Activation information

AppEUI	01 01 01 01 01 01 01 01
DevEUI	00 80 E1 15 00 0A 95 37
AppKey	•• •• •• •• •• •• •• •• •• •• •• •• •• •• •• ••

Session information (pending)

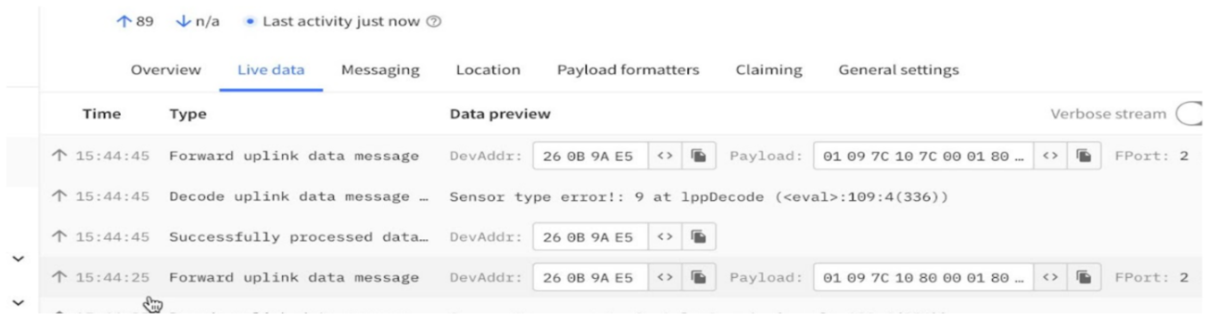
Device address	27 FD BA B1
NwkSKey	•• •• •• •• •• •• •• •• •• •• •• •• •• •• •• ••
AppSKey	•• •• •• •• •• •• •• •• •• •• •• •• •• •• •• ••

Figure 4: Registration of End Device

6.3.1 Data Reception

After the successful integration of the end device within the application on TTN, reception of data will be set through the end device on TTN. Here, as discussed above, DHT11 and IR sensors are used as the source of data, so after configuring the end device it will

send the sensor data towards the gateway and this data will arrive in encrypted form.
As seen in the figure, the encrypted data from end device through uplink is received.

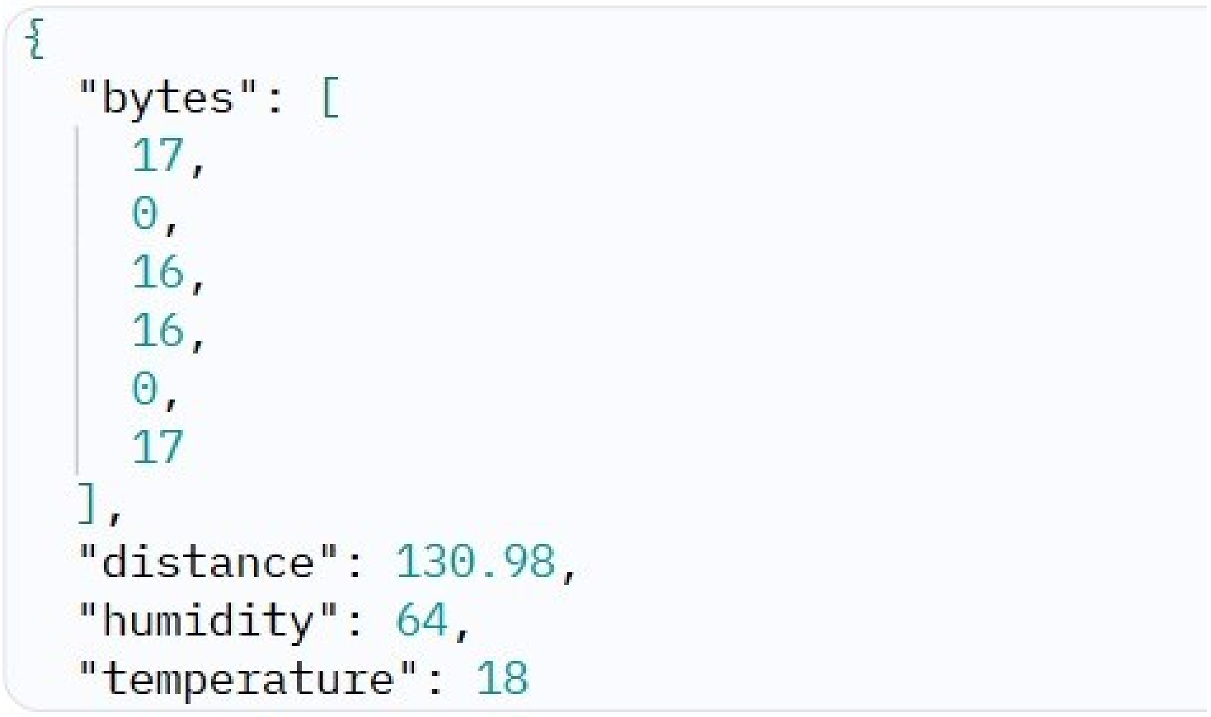


```
Time      Type                             Data preview
15:44:45  Forward uplink data message      DevAddr: 26 0B 9A E5   Payload: 01 09 7C 10 7C 00 01 80 ...   FPort: 2
15:44:45  Decode uplink data message ...   Sensor type error!: 9 at lppDecode (<eval>:109:4(336))
15:44:45  Successfully processed data...   DevAddr: 26 0B 9A E5
15:44:25  Forward uplink data message      DevAddr: 26 0B 9A E5   Payload: 01 09 7C 10 80 00 01 80 ...   FPort: 2
```

Figure 5: Encrypted Data Reception on TTN

Payload Formatters of TTN to decode this encrypted data and get the actual data, is successfully setup. Here is the custom Javascript code that decodes the data:

```
function decodeUplink(input) {
 let bytes = input.bytes;
 let temperature = null;
 let humidity = null;
 let distance = null;
 let warnings = [];
 let errors = [];
 try {
   if (bytes.length >= 6) {
     temperature = (bytes[0] << 8) | bytes[1];
     humidity = (bytes[2] << 8) | bytes[3];
     distance = (bytes[4] << 8) | bytes[5];
     temperature = temperature / 100;
     humidity = humidity / 100;
     distance = distance / 100;
   } else {
     errors.push("Payload length is less than expected. Expected
         6 bytes, got " + bytes.length);
   }
 } catch (error) {
   errors.push("Error decoding payload: " + error.message);
 }
 return {
   data: {
     temperature: temperature,
     humidity: humidity,
     distance: distance,
     bytes: bytes
   },
   warnings: warnings,
   errors: errors
 };
}
```



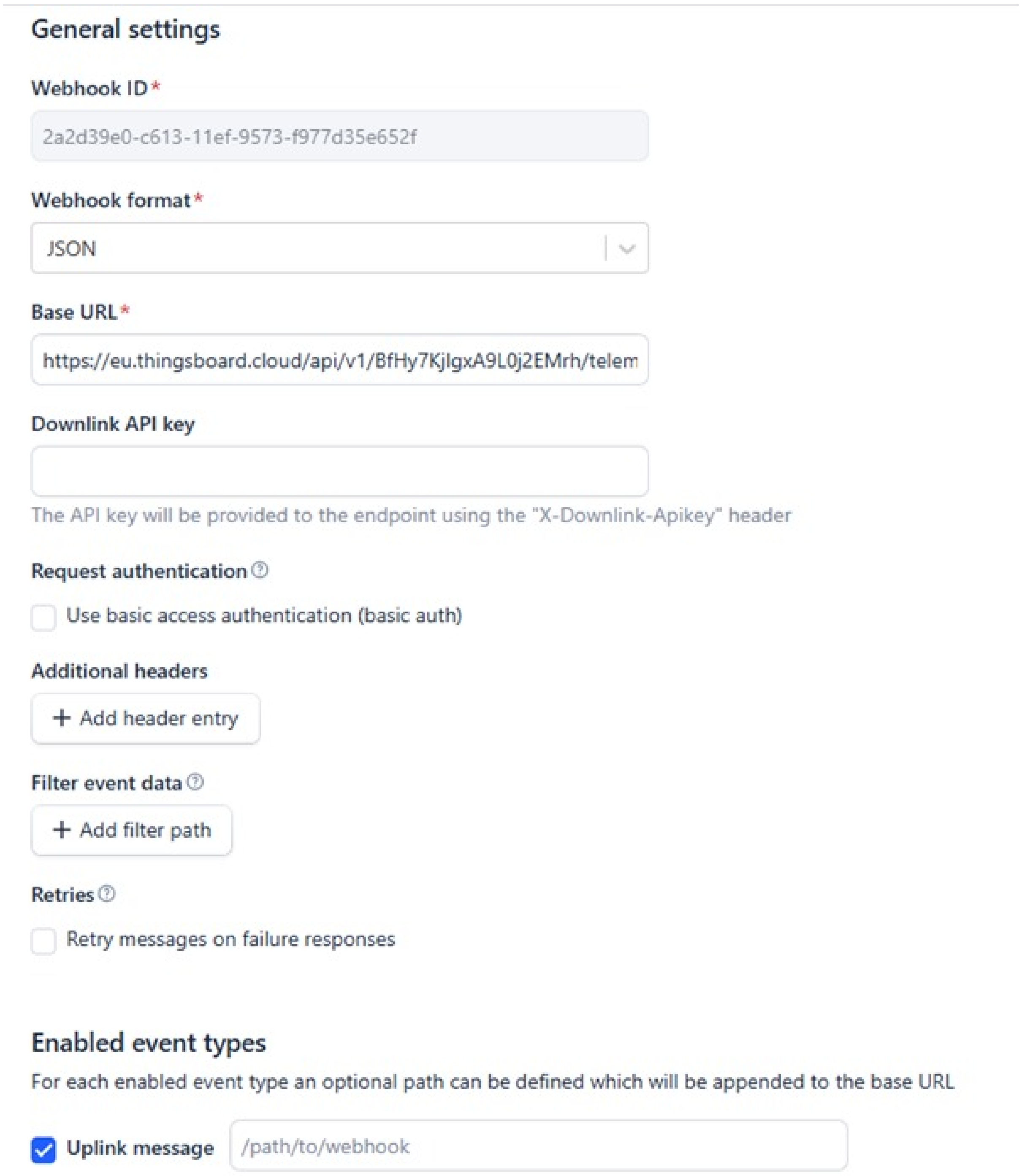
```
{
  "bytes": [
    17,
    0,
    16,
    16,
    0,
    17
  ],
  "distance": 130.98,
  "humidity": 64,
  "temperature": 18
```

Figure 6: Decoded Test Payload

6.4 Connecting TTN with Application Server

For the purpose of connecting The Things Network with the Things Board, navigate to the integrations and add a webhook from there.

The figure given below shows the details of the webhook that has been scheduled to forward the uplink messages to the application server. The base URL is basically the token of the device that we have created on the things board that will receive data.



General settings

Webhook ID*

2a2d39e0-c613-11ef-9573-f977d35e652f

Webhook format*

JSON

Base URL*

https://eu.thingsboard.cloud/api/v1/BfHy7KjlgxA9L0j2EMrh/telem

Downlink API key

The API key will be provided to the endpoint using the "X-Downlink-Apikey" header

Request authentication

☐ Use basic access authentication (basic auth)

Additional headers

Filter event data

Retries

☐ Retry messages on failure responses

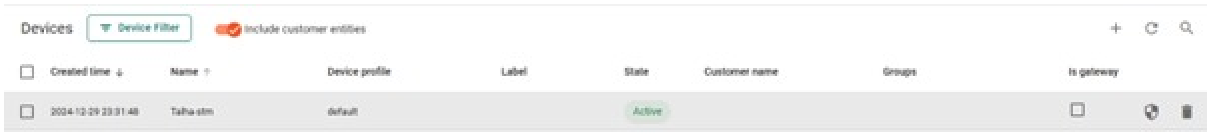
Enabled event types

For each enabled event type an optional path can be defined which will be appended to the base URL

☑ Uplink message /path/to/webhook

Figure 7: Application

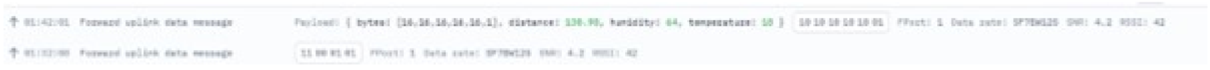
After successfully connecting The Things Network (TTN) with ThingsBoard, the device will show an active connection on ThingsBoard. This is shown in the figure below:



Created time	Name	Device profile	Label	State	Customer name	Groups	Is gateway
2024-12-29 23:31:48	Talha-stm	default		Active			☐

Figure 8: Device Status

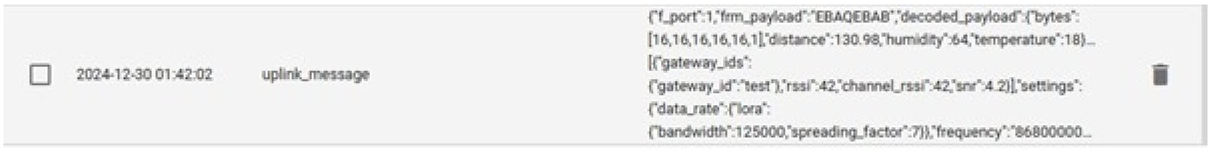
Once the connection is successful, TTN will send the uplink data received from the sensors to the ThingsBoard device. The figure below shows the forwarding of uplink data:



```
↑ 01:42:01  Forward uplink data message    Payload: { bytes: [16,16,16,16,16,1], distance: 130.98, humidity: 64, temperature: 18 }  10 10 10 10 10 01  FPort: 1  Data rate: SF7BW125  SNR: 4.2  RSSI: 42
↑ 01:32:00  Forward uplink data message    11 00 01 01  FPort: 1  Data rate: SF7BW125  SNR: 4.2  RSSI: 42
```

Figure 9: Forwarding of uplink Data

This data will be displayed in the device telemetry, as shown below:



☐	2024-12-30 01:42:02	uplink_message	{"f_port":1,"frm_payload":"EBAQEBAB","decoded_payload":{"bytes":[16,16,16,16,16,1],"distance":130.98,"humidity":64,"temperature":18}...[{"gateway_ids":{"gateway_id":"test"},"rssi":42,"channel_rssi":42,"snr":4.2}],"settings":{"data_rate":{"lora":{"bandwidth":125000,"spreading_factor":7}},"frequency":"86800000...	

Figure 10: Forwarding of uplink Data

A dashboard was also created to visualize the data and keep them updated, as they will reflect the real-time data coming from the uplink.

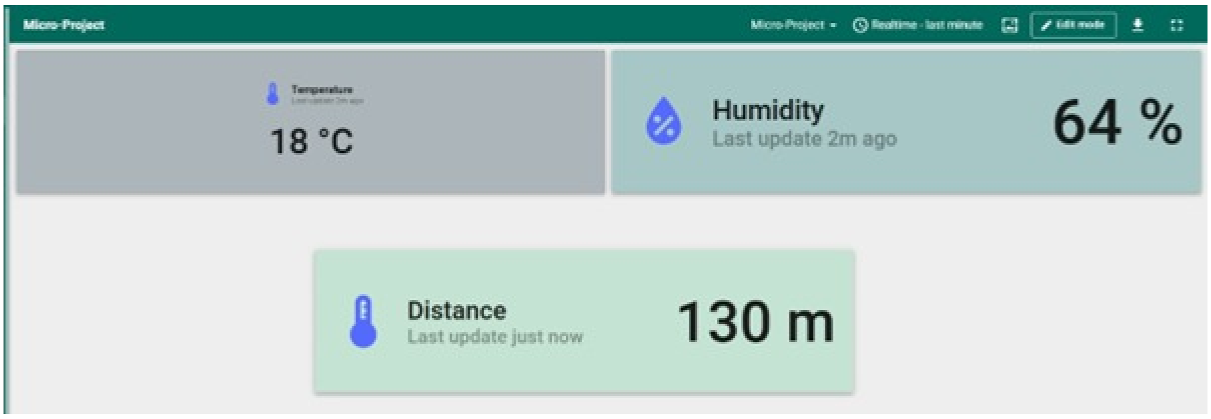


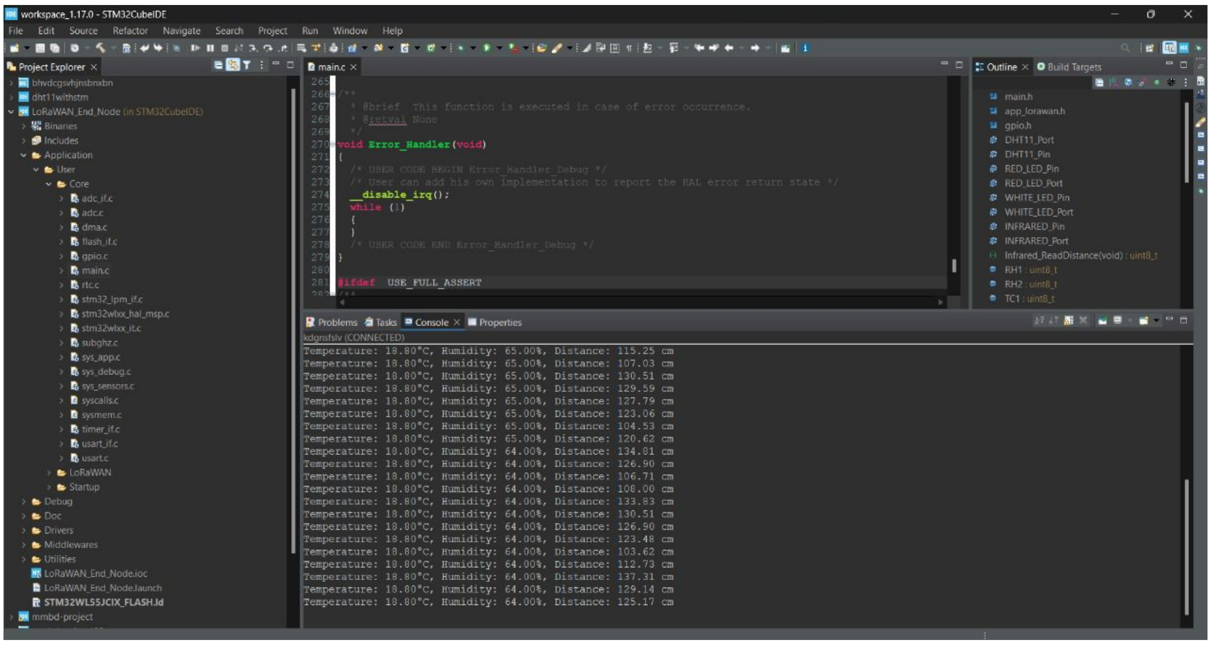
Figure 11: Forwarding of uplink Data

7 Testing and Validation

The main objective of testing and validation is to verify the correct operation of the system, including successful data acquisition from the two sensors and reliable data transmission over LoRaWAN using STM32 to a built-in gateway. So, the snippets below correctly depict the testing phase:

7.1 Sensor Integration Testing

First of all, the data extraction from sensors integrated with STM32, and the display of their values, depict the validation of the embedded system being created.



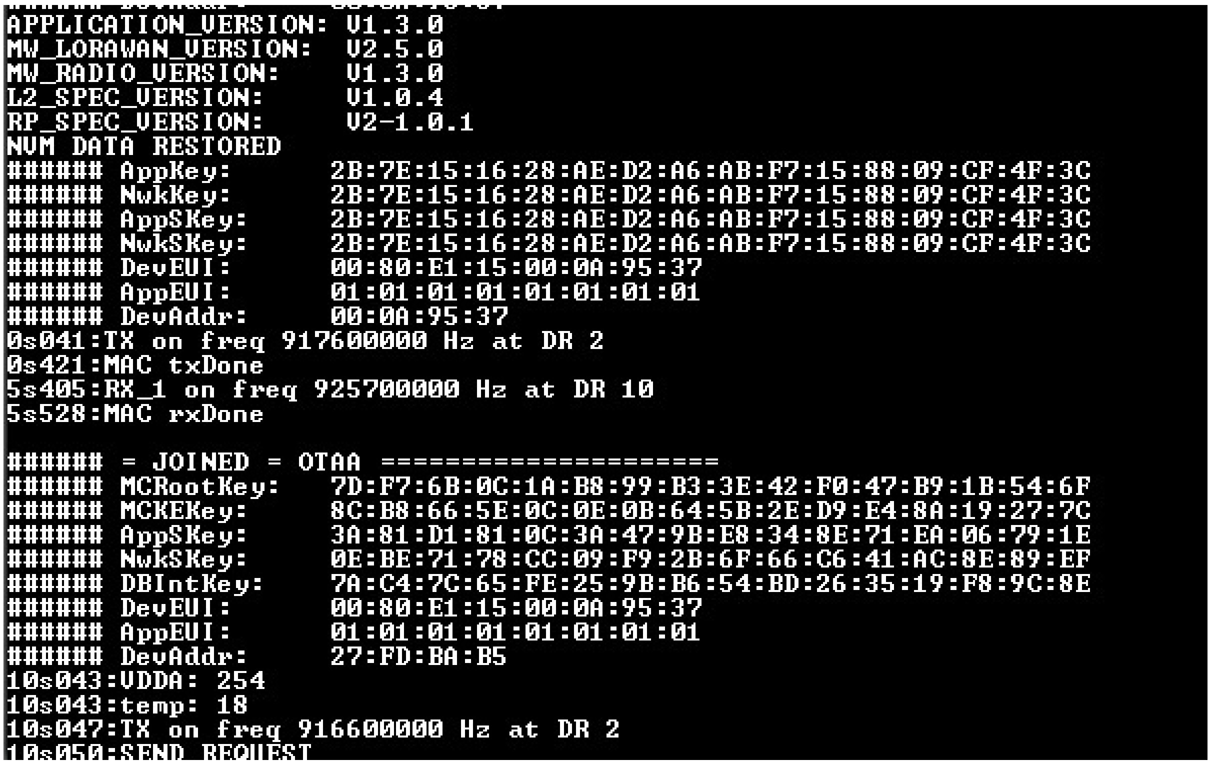
```c
/**
  * @brief  This function is executed in case of error occurrence.
  * @retval None
  */
void Error_Handler(void)
{
  /* USER CODE BEGIN Error_Handler_Debug */
  /* User can add his own implementation to report the HAL error return state */
  __disable_irq();
  while (1)
  {
  }
  /* USER CODE END Error_Handler_Debug */
}

#ifdef  USE_FULL_ASSERT
```

```
kdgnsfslv (CONNECTED)
Temperature: 18.80°C, Humidity: 65.00%, Distance: 115.25 cm
Temperature: 18.80°C, Humidity: 65.00%, Distance: 107.03 cm
Temperature: 18.80°C, Humidity: 65.00%, Distance: 130.51 cm
Temperature: 18.80°C, Humidity: 65.00%, Distance: 129.59 cm
Temperature: 18.80°C, Humidity: 65.00%, Distance: 127.79 cm
Temperature: 18.80°C, Humidity: 65.00%, Distance: 123.06 cm
Temperature: 18.80°C, Humidity: 65.00%, Distance: 104.53 cm
Temperature: 18.80°C, Humidity: 65.00%, Distance: 120.62 cm
Temperature: 18.80°C, Humidity: 64.00%, Distance: 134.81 cm
Temperature: 18.80°C, Humidity: 64.00%, Distance: 126.90 cm
Temperature: 18.80°C, Humidity: 64.00%, Distance: 106.71 cm
Temperature: 18.80°C, Humidity: 64.00%, Distance: 108.00 cm
Temperature: 18.80°C, Humidity: 64.00%, Distance: 133.83 cm
Temperature: 18.80°C, Humidity: 64.00%, Distance: 130.51 cm
Temperature: 18.80°C, Humidity: 64.00%, Distance: 126.90 cm
Temperature: 18.80°C, Humidity: 64.00%, Distance: 123.48 cm
Temperature: 18.80°C, Humidity: 64.00%, Distance: 103.62 cm
Temperature: 18.80°C, Humidity: 64.00%, Distance: 112.73 cm
Temperature: 18.80°C, Humidity: 64.00%, Distance: 137.31 cm
Temperature: 18.80°C, Humidity: 64.00%, Distance: 129.14 cm
Temperature: 18.80°C, Humidity: 64.00%, Distance: 125.17 cm
```

Figure 12: Sensor Output on Cube IDE Console.

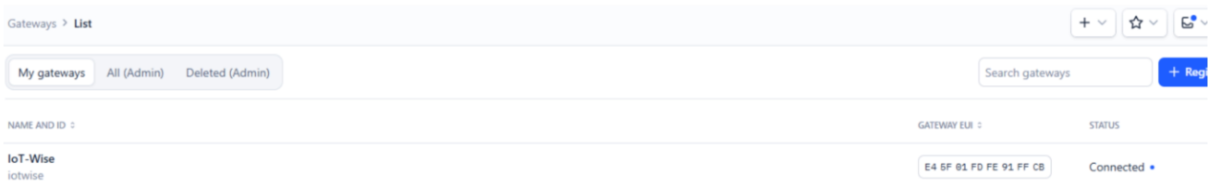
7.2 LoRaWAN Communication Testing

Then, for communication with TTN, both the connection status from Gateway and OTA activation status on Tera Term depicts the successful connection between the embedded system and TTN.



```
APPLICATION_VERSION: V1.3.0
MW_LORAWAN_VERSION:  V2.5.0
MW_RADIO_VERSION:    V1.3.0
L2_SPEC_VERSION:     V1.0.4
RP_SPEC_VERSION:     V2-1.0.1
NVM DATA RESTORED
###### AppKey:      2B:7E:15:16:28:AE:D2:A6:AB:F7:15:88:09:CF:4F:3C
###### NwkKey:      2B:7E:15:16:28:AE:D2:A6:AB:F7:15:88:09:CF:4F:3C
###### AppSKey:     2B:7E:15:16:28:AE:D2:A6:AB:F7:15:88:09:CF:4F:3C
###### NwkSKey:     2B:7E:15:16:28:AE:D2:A6:AB:F7:15:88:09:CF:4F:3C
###### DevEUI:      00:80:E1:15:00:0A:95:37
###### AppEUI:      01:01:01:01:01:01:01:01
###### DevAddr:     00:0A:95:37
0s041:TX on freq 917600000 Hz at DR 2
0s421:MAC txDone
5s405:RX_1 on freq 925700000 Hz at DR 10
5s528:MAC rxDone

###### = JOINED = OTAA =====================
###### MCRootKey:   7D:F7:6B:0C:1A:B8:99:B3:3E:42:F0:47:B9:1B:54:6F
###### MCKEKey:     8C:B8:66:5E:0C:0E:0B:64:5B:2E:D9:E4:8A:19:27:7C
###### AppSKey:     3A:81:D1:81:0C:3A:47:9B:E8:34:8E:71:EA:06:79:1E
###### NwkSKey:     0E:BE:71:78:CC:09:F9:2B:6F:66:C6:41:AC:8E:89:EF
###### DBIntKey:    7A:C4:7C:65:FE:25:9B:B6:54:BD:26:35:19:F8:9C:8E
###### DevEUI:      00:80:E1:15:00:0A:95:37
###### AppEUI:      01:01:01:01:01:01:01:01
###### DevAddr:     27:FD:BA:B5
10s043:VDDA: 254
10s043:temp: 18
10s047:TX on freq 916600000 Hz at DR 2
10s050:SEND REQUEST
```

Figure 13: OTA Activation on Tera Term

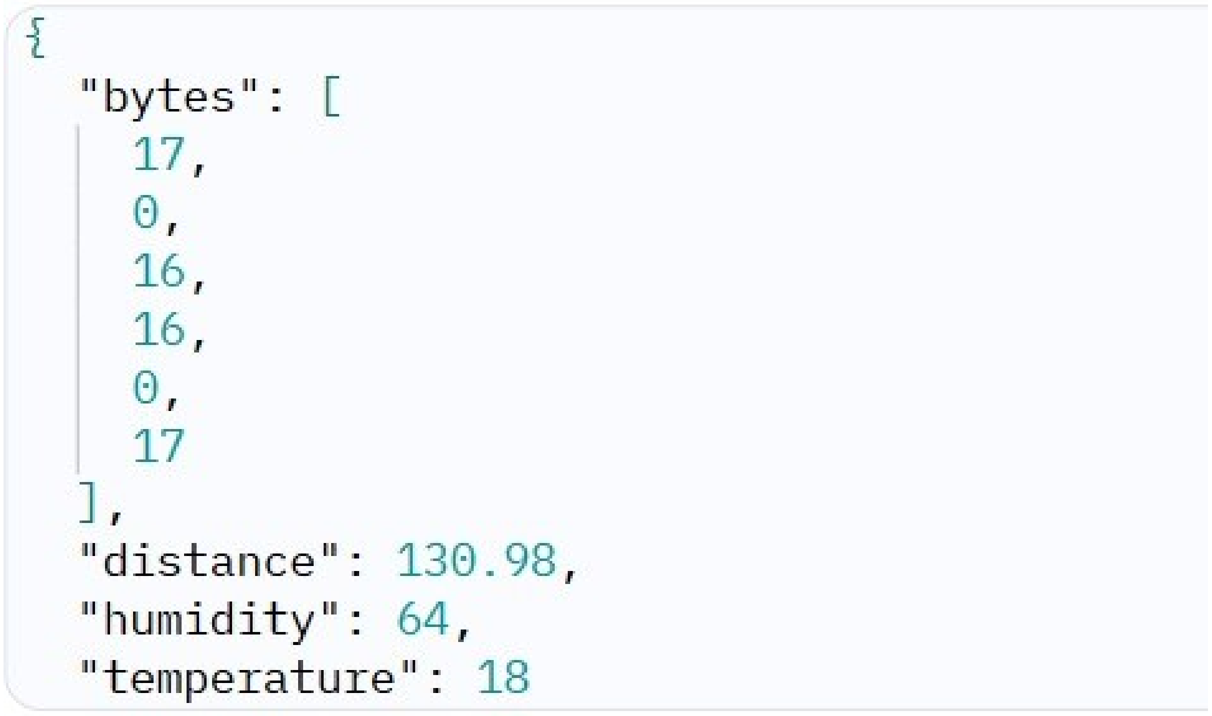


NAME AND ID	GATEWAY EUI	STATUS
IoT-Wise iotwise	E4 5F 01 FD FE 91 FF CB	Connected ●

Figure 14: Successful Connection on Gateway

7.3 Data Integrity Testing

Display of decoded data on TTN ensures the successful communication between system and Cloud, hence falling in to the requirements.



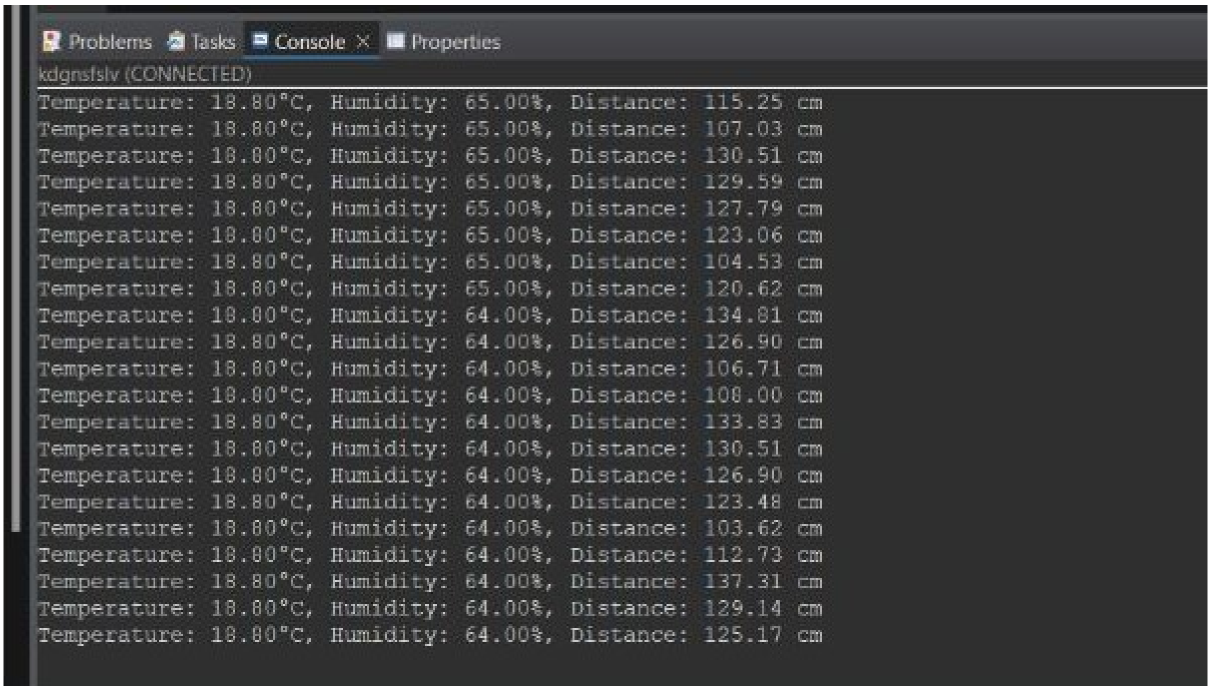
```
{
  "bytes": [
    17,
    0,
    16,
    16,
    0,
    17
  ],
  "distance": 130.98,
  "humidity": 64,
  "temperature": 18
```

Figure 15: Decoded Data at Gateway

8 Results and Discussion

8.1 Sensor Output

Successfully transmitted temperature and humidity data from the integrated sensors to the built-in gateway.



```
kdgnsfslv (CONNECTED)
Temperature: 18.80°C, Humidity: 65.00%, Distance: 115.25 cm
Temperature: 18.80°C, Humidity: 65.00%, Distance: 107.03 cm
Temperature: 18.80°C, Humidity: 65.00%, Distance: 130.51 cm
Temperature: 18.80°C, Humidity: 65.00%, Distance: 129.59 cm
Temperature: 18.80°C, Humidity: 65.00%, Distance: 127.79 cm
Temperature: 18.80°C, Humidity: 65.00%, Distance: 123.06 cm
Temperature: 18.80°C, Humidity: 65.00%, Distance: 104.53 cm
Temperature: 18.80°C, Humidity: 65.00%, Distance: 120.62 cm
Temperature: 18.80°C, Humidity: 64.00%, Distance: 134.81 cm
Temperature: 18.80°C, Humidity: 64.00%, Distance: 126.90 cm
Temperature: 18.80°C, Humidity: 64.00%, Distance: 106.71 cm
Temperature: 18.80°C, Humidity: 64.00%, Distance: 108.00 cm
Temperature: 18.80°C, Humidity: 64.00%, Distance: 133.83 cm
Temperature: 18.80°C, Humidity: 64.00%, Distance: 130.51 cm
Temperature: 18.80°C, Humidity: 64.00%, Distance: 126.90 cm
Temperature: 18.80°C, Humidity: 64.00%, Distance: 123.48 cm
Temperature: 18.80°C, Humidity: 64.00%, Distance: 103.62 cm
Temperature: 18.80°C, Humidity: 64.00%, Distance: 112.73 cm
Temperature: 18.80°C, Humidity: 64.00%, Distance: 137.31 cm
Temperature: 18.80°C, Humidity: 64.00%, Distance: 129.14 cm
Temperature: 18.80°C, Humidity: 64.00%, Distance: 125.17 cm
```

Figure 16: Compiled Sensor Data on TTN

8.2 Hardware Assembly

Successfully connected the hardware components to STM32 Microcontroller and their working.

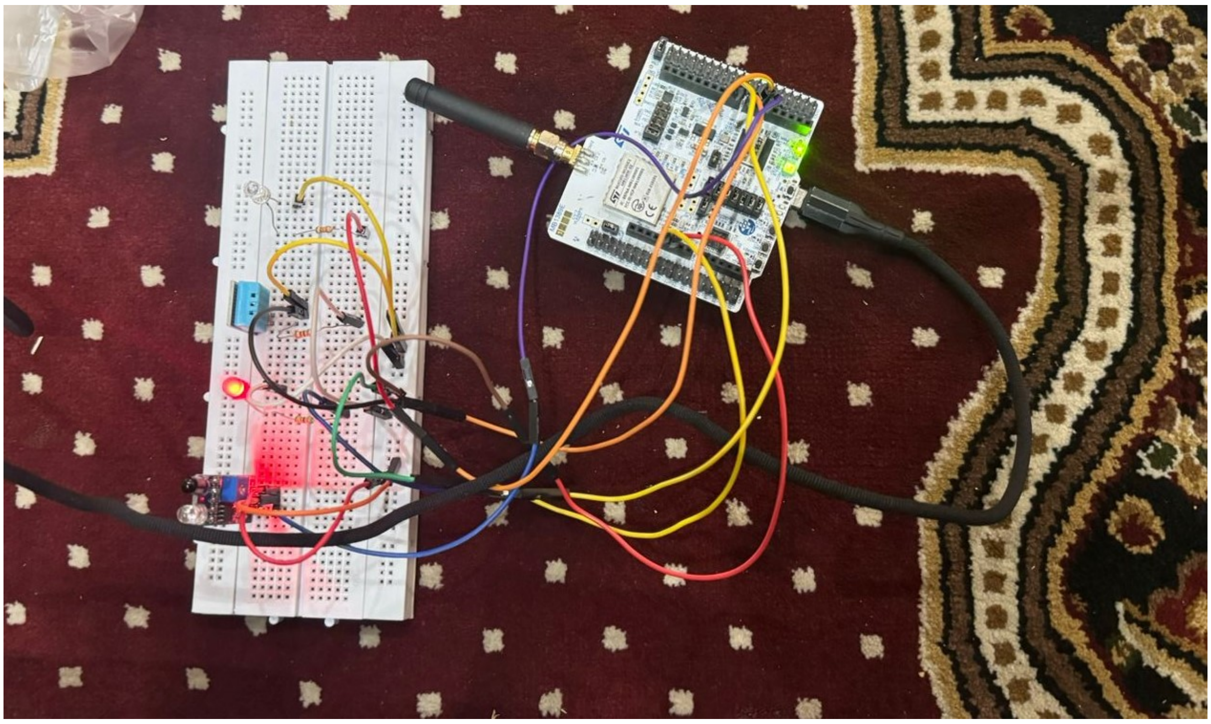
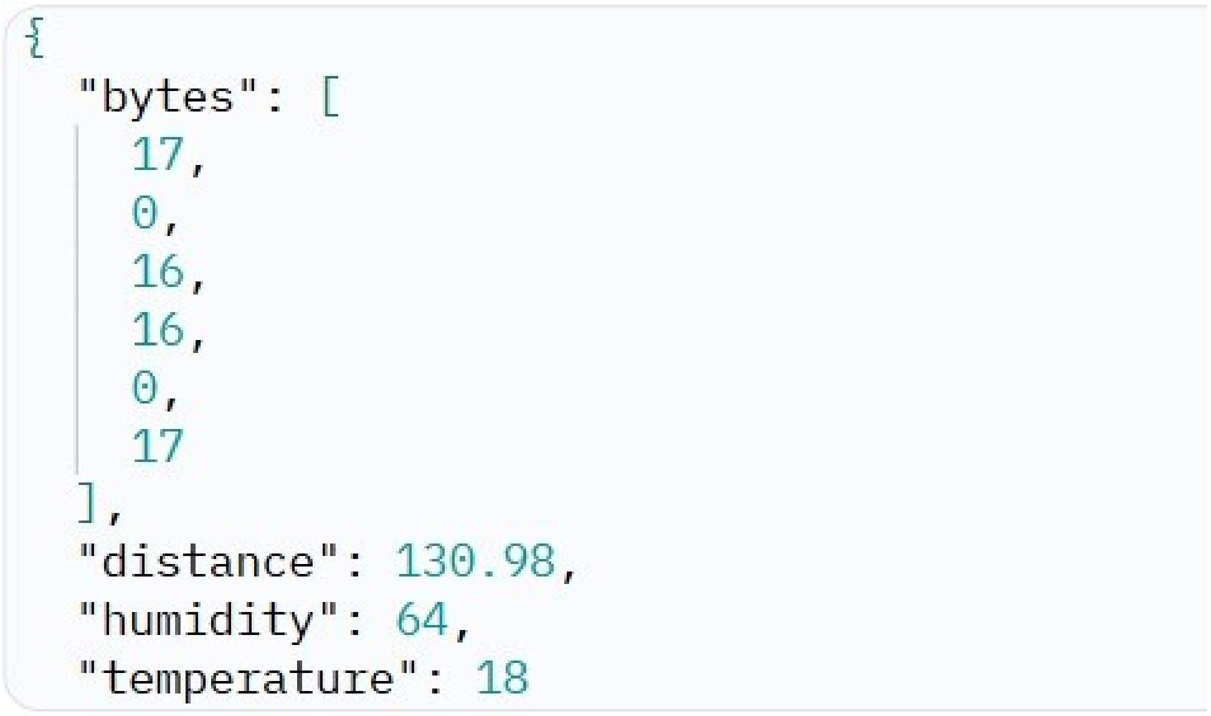


Figure 17: Hardware Integrated Components with STM32

8.3 Assembly

The display of transmitted data on TTN ensures the successful data extracting from sensors connected.



```
{
  "bytes": [
    17,
    0,
    16,
    16,
    0,
    17
  ],
  "distance": 130.98,
  "humidity": 64,
  "temperature": 18
```

Figure 18: Compiled Sensor Data on TTN

9 Challenges and Solution

Power Optimization While the STM32WL55 is designed for low-power applications, careful configuration of sleep modes, transmission intervals, and duty cycles is required to maximize battery life.
Network Scalability Duty cycle rules and gateway capacity affect how scalable LoRaWAN networks are. Large-scale installations require effective network planning.
Data Rate Limitations Applications needing frequent or massive data transmissions may find LoRaWAN unsuitable due to its low data rate. This restriction can be lessened by optimizing data compression and prioritizing techniques.

10 Future Work

Advanced Power Management Investigating energy-harvesting methods (such as solar panels) to supplement STM32WL55-based systems' battery power.
Edge Computing In order to process data locally and lessen the need for frequent cloud communication, edge analytics is being implemented on the STM32WL55.
Security Enhancements Enhancing security protocols, like firmware upgrades and secure boot, to shield STM32WL55-based devices from online attacks.

11 Conclusion

With its dual-core architecture and integrated LoRa transceiver, the STM32 Nucleo WL55JC1 provides a strong foundation for creating LoRaWAN-based wireless communication systems. The goal of this project is to create a scalable and effective environmental monitoring system by utilizing its low-power characteristics and adaptable development environment. The STM32WL55's ability to overcome the drawbacks of conventional wireless technologies and open the door for creative IoT solutions is highlighted by the insights from the body of existing literature.

12 References

- LoRa Alliance, "Official information about the LoRaWAN protocol and its applications," [Online]. Available: https://lora-alliance.org

– STMicroelectronics, ”Resources, datasheets, and tutorials for STM32 microcontrollers,” [Online]. Available: https://www.st.com

– The Things Network, ”Documentation about LoRaWAN, gateways, and IoT integration,” [Online]. Available: https://www.thethingsnetwork.org

– IEEE Xplore Digital Library, ”Academic articles, papers, and case studies on LoRaWAN, STM32, and IoT,” [Online]. Available: https://ieeexplore.ieee.org

– Hackster.io, ”Projects and tutorials related to STM32, IoT applications, and LoRaWAN,” [Online]. Available: https://www.hackster.io

– Adafruit Learning System, ”Tutorials on sensors, microcontrollers, and LoRaWAN integration,” [Online]. Available: https://learn.adafruit.com

– Arduino Project Hub, ”Projects related to LoRaWAN and STM32 microcontrollers,” [Online]. Available: https://create.arduino.cc/projecthub

– AWS IoT Core, ”Integrating LoRaWAN applications with cloud platforms,” [Online]. Available: https://aws.amazon.com/iot-core