Óbudai Egyetem
Kandó Kálmán Villamosmérnöki Kar
Elektronikai és Kommunikációs Rendszerek Intézet
Műszertechnikai és Automatizálási Tanszék

TUDOMÁNYOS DIÁKKÖRI DOLGOZAT

MULTI-TECHNOLOGY PRESENCE MANAGEMENT SYSTEM

Szerző: **Le Quoc Viet**
villamosmérnök BSc. szak, III. évf.

Konzulens: **Borsos Döníz**
tanársegéd

ABSTRACT

This paper describes the architecture of a LoRa, Wi-Fi, and BLE multi-technology presence tracking system that would track BLE-enabled devices and beacons in space. BLE is a low-power wireless communication technology designed to replace cables in devices needing only small amounts of data transfer. It is optimized for devices with long battery life, such as wearables, smart sensors, and IoT devices. BLE is an integral constituent of IoT and smart devices, enabling their power-efficient communication. Its ecosystem is significant in the applications of smart homes, cities, and healthcare. BLE tracking can be used for asset management and personnel management in manufacturing, healthcare, and logistics to improve experiences, public safety, and emergency responses.

The system will include two Heltec LoRa32 V2 devices: transmitter and receiver. The sender scans its surroundings, detecting BLE devices/beacons in real-time and sends data with LoRa, a communication technology with a large range and low consumption. It also connects to Wi-Fi in order to send data in real-time to ThingSpeak, a data gathering and visualization cloud. This allows the user to monitor in real-time trends of the number of devices detected. It uses LoRa to collect data from the sender and acts as a substitute for local data gathering.

Integrating LoRa for long-distance communication, Wi-Fi for cloud connectivity, and BLE for device detection offers a flexible and scalable solution to industries looking for presence tracking that helps them build better asset management, security, and operational efficiency.

Keywords: *BLE tracking, LoRa, Wi-Fi, Heltec LoRa32 V2, real-time monitoring, ThingSpeak, IoT, device presence, Bluetooth beacons*

TABLE OF CONTENTS

LIST OF ABBREVIATION

Abbreviation	Definition
IoT	Internet of Things
PAN	Personal Area Network
SIG	Special Interest Group
GATT	Generic Attribute Profile
CSS	Chirp Spread Spectrum
Wi-Fi	Wireless Fidelity
GHz	Gigahertz
USB	Universal Serial Bus
HTTP	Hypertext Transfer Protocol
UUID	Universally Unique Identifier
URL	Uniform Resource Locator
ISM band	Industrial, Scientific, And Medical Radio Band
CBC MAC	Cipher Block Chaining Message Authentication Cod
MAC	Media Access Control
AES	Advanced Encryption Standard
AES-CCM	Advanced Encryption Standard-Counter with CBC-MAC
LE	Low Energy
LPWA	Low-Power Wide-Area

WAN	Wide Area Network
BTS	Base Transceiver Station
IP	Internet Protocol
MoCA	Multimedia Over Coax
RTLS	Real-Time Locating Systems
RSSI	Received Signal Strength Indicator
SSID	Service Set Identifier
API	Application Programming Interface
URL	Uniform Resource Locator
SSL	Secure Sockets Layer
TLS	Transport Layer Security

LIST OF FIGURES

1 INTRODUCTION

This paper deals with the design of multi-technology presence tracking equipment that uses LoRa, Wi-Fi, and BLE to monitor and track all BLE-enabled devices and beacons within a given area. BLE is a wireless communication technology meant to bring efficiency in low-power connectivity. It works within the same frequency range as the traditional Bluetooth but is more optimized for devices with long battery life and requiring less data transfer like wearable gadgets, smart sensors, and IoT applications. BLE seems to be playing a very important role in this modern world for enabling IoT and smart device growth by allowing communication between devices without consuming much power, which becomes very essential in smart homes, cities, healthcare systems. With this, tracking BLE devices is of great importance in managing expensive assets or personnel in industries such as manufacturing, healthcare, and logistics. Moreover, this technology also provides immense assistance in improving user experiences in the use of products, public safety, and emergency response from hospital utilization, and data for analytics and decision-making. It includes two Heltec LoRa32 V2 devices: one sender and one receiver. BLE scanning technology by a sender, which it wears, scans surroundings to detect BLE devices and beacons in its vicinity. These devices are detected and tracked in real-time, each of their data being transmitted over LoRa - a long-range, low-power wireless transmission technology suitable for monitoring over wide areas. The sender's device can be connected to Wi-Fi so that real-time information may be uploaded to ThingSpeak, and the data can be sent to the receiver. ThingSpeak provides a cloud-based solution to collect, process, and visualise data. The web application allows users to view trends on a user-friendly dashboard in real-time, such as the number of devices detected. This would provide critical insight into the presence and movement patterns of devices that could be useful in smart environments, asset tracking, and public safety monitoring applications. The receiver device is an alternative means of local data collection; it also collects data from the sender using LoRa. Since the system has been designed to enable dual communication capability both via LoRa and Wi-Fi, the consistency of data relayed over long distances for storage and analysis is always reliable, irrespective of the sender's location. These technologies integrated together have made the system highly flexible in handling both local and remote monitoring situations. It amalgamates various forms of communication within a single package to provide a highly flexible and scalable tracking system that can be applied to create actionable insight in real time. The system can be used in crowd management within

large events and areas to inform on population density and movement patterns in smart city and public safety applications. This would help prevent overcrowding, act on emergency responses in a much-improved manner, and optimize the deployment of resources. For instance, in the case of emergency evacuation, monitoring the movement of people helps the first responders trace missing or stranded people to take effective action in a proper manner. With the expansion of IoT and smart environments, there is ever-growing demand for versatile, scalable, and energy-efficient tracking systems that would be adaptable in different settings. This system brought together BLE for local detection, LoRa for long-range communication, and Wi-Fi for cloud integration, making it the perfect wrap of an end-to-end solution that can meet industrial and public needs in bettering asset management, safety, and operational efficiencies across applications.

2 OVERVIEW OF KEY TECHNOLOGIES

2.1 Overview of Key Technologies

2.1.1 Bluetooth Low Energy (BLE)

Commonly referred to as" Bluetooth smart," Bluetooth low energy (BLE) [1], [2]is the most popular wireless standard for IoT gadgets. The Bluetooth Special Interest Group (SIG) created and continues to support this wireless personal area network (WPAN) technology. [3]BLE stands for Bluetooth Low Energy, a wireless Personal Area Network (PAN) designed to support very low power consumption, with cost and also a similar or somewhat reduced communication range compared to the Classic Bluetooth. The point of this technology, according to the Bluetooth Special Interest Group (SIG), developed and introduced with Bluetooth 4.0 in December 2009, is healthcare, fitness, smart home devices, beacons, and security. BLE uses a similar footprint but operates in the 2.4 GHz the industrial, scientific, and medical radio band (ISM band). It is targeted mainly by the device's capability to broadcast data in small packets, thus enabling it to connect and communicate over a short distance. Unlike Classic Bluetooth, BLE offers optimization for intermittent data transmission rather than constant, bidirectional data communication. The power consumption of BLE is generally 100 times lower than that of Classic Bluetooth. This power efficiency allows BLE devices to operate from the same battery for months and even years. While the earlier Bluetooth Classic offers point-to-point communication in support of audio streaming and data transfer, BLE adds broadcast communications (one-to-all) that are particularly suited to serve indoor localization and contact tracing applications. [4] In order to implement intricate IoT solutions, BLE further offers features for configuring mesh networks (many-to-many communications). BLE introduces new vocabulary for every channel of communication. In the context of point-to-point communication, for example, a device may be a peripheral device that connects to a central device or a central device that actively searches for additional BLE devices to connect to. Devices in the broadcast communication mode can be either broadcasters, which send ads without a connection, or observers, which scan the broadcaster's ads. BLE broadcasters and observers collaborate to build a flooding-based network that links all devices in the mesh

communication mode, which permits many-to-many conversations. Provisioner and unprovisioned devices, node and relay node are new terms for devices in mesh mode.

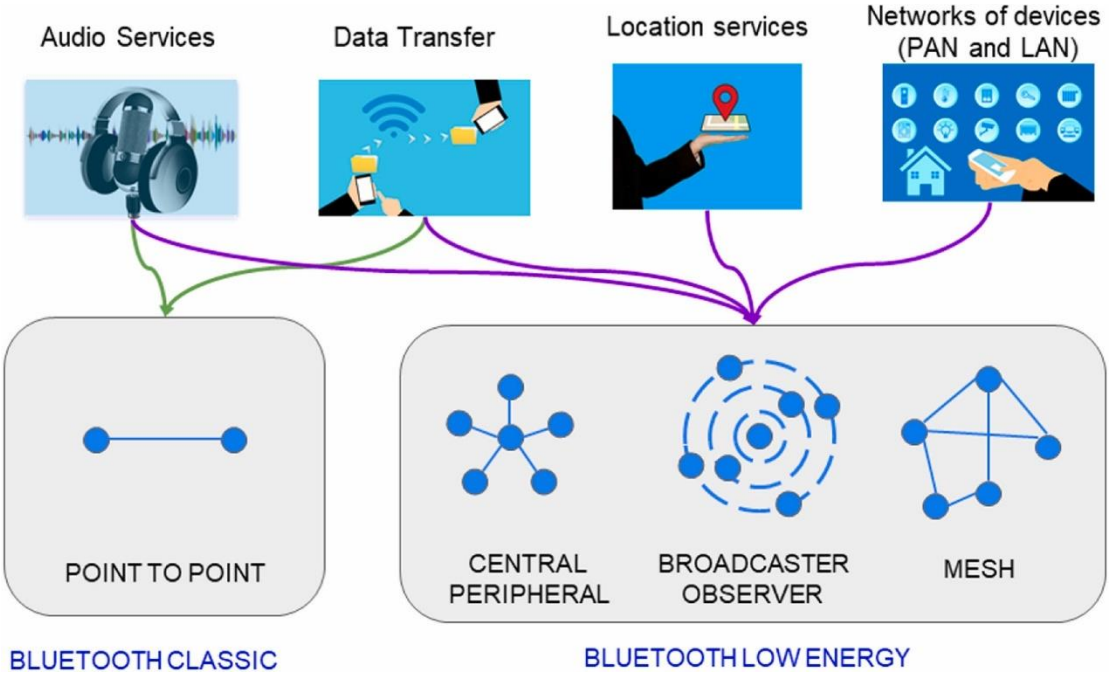

Figure 1 Bluetooth Classic and BLE Comparison [4]

Two types of devices exist in BLE: Bluetooth Smart communicates only with other Bluetooth Smart Ready or other Bluetooth Smart devices; and Bluetooth Smart Ready, which can interface with Bluetooth Smart, Bluetooth Smart Ready, and Classic Bluetooth devices. [5] BLE employs a hierarchical information structure in its data exchange. In BLE, Generic Attribute Profile (GATT) defines how information is formatted and exchanged between devices. The Generic Access Profile (GAP) defines a general topology for how BLE devices interconnect [6] .A BLE device might act as a Broadcaster (advertises but does not allow connections), an Observer (sees advertisements but does not initiate a connection), a Peripheral (advertises and accepts connections), or a Central (sees advertisements and initiate connections) . [7] .The Generic Attribute (GATT) profile defines how data transfer takes place after the establishment of a dedicated connection by the GAP. In addition, it specifies roles for the nodes, where one acts as the client and the other as the server. [8]

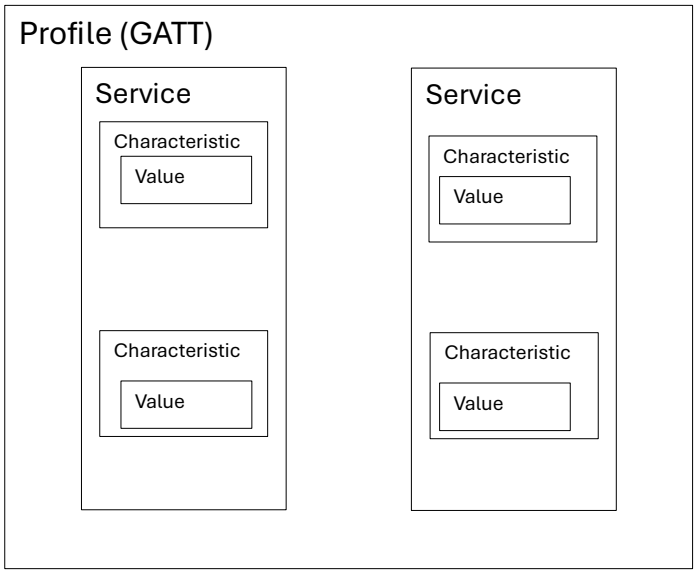

Figure 2:Generic Attribute Profile (GATT) [9]

All BLE peripherals advertise their supported services and characteristics, which may be read or written by a connected central device, like a smartphone or computer, for real-time updates, even temperature sensors sending data to a connected device. BLE profiles define how a certain application should behave; thus, allowing the interchangeability of devices from different manufacturers to work together. Examples include health care profiles such as Blood Pressure Profile and Heart Rate Profile; sports and fitness profiles including Cycling Speed and Cadence Profile and Running Speed and Cadence Profile; and proximity sensing applications such as iBeacons. BLE provides the benefits of low power consumption, consequently low cost due to cheap chipsets; wide compatibility with the main mobile operating systems such as iOS, Android, Windows, and Linux; and even low connection times around 3 ms to start data transfer. Its low data rates—1 Mbps and 2 Mbps are the maximum supported—make BLE quite limited at a basic level. Anyhow, this is coupled with a normal range of hardly over 30 meters in real-world conditions and the fact that being radio-based, BLE signals are subject to interception and possible attacks. There are many applications of BLE in all fields. In a smart home, BLE can allow users to turn lights and thermostats on/off with their smartphones. BLE devices share data with mobile applications, which then display the user's numerous vital health parameters. A fitness tracker or smartwatch may use BLE to broadcast performance data to a paired device for an improved user experience in health and wellness monitoring. It's also a transformative technology, enabling low-power, short-range wireless communication in an increasing

number of applications, from consumer electronics to IoT devices via Bluetooth Low Energy. In nature, it is a major technology in the alleviation of the steadily rising burdens concerning connectivity at high energy efficiency. With the still-progressing nature of this technology, BLE will stay in a critical role in the direction of connected devices and smart solutions in the future.

While BLE is an efficient and low-power technology, it is rather suitable only for short-range communication. In fact, BLE exhibits many limitations with regard to the design and feasibility of presence-tracking systems:

- Range Limitations: BLE normally works in an ideal environment within a range of 30 meters. Indoors this can go further down because of interference caused by walls and furniture, among other obstacles. Tracking devices over a big area or across multi-floor buildings may not be feasible with BLE alone; hence, its complement with LoRa in my system.
- Data Transfer Rate: BLE is limited to 1 Mbps, though with newer standards it reached 2 Mbps. It is thus not useful for transferring large volumes of data. This would not be particularly limiting for presence tracking, which typically would only require minimal data like device IDs, but this would surely limit the applications needing real-time high-bandwidth communication.

In this respect, this project also will integrate LoRa into its system for longer-range, low-bandwidth communication, ensuring that BLE scanning-collected data will be able to be transmitted in a wider area by enabling tracking in real time around larger or more complicated environments. BLE is made to facilitate communication with IoT devices, computationally limited sensors, and very low energy consumption. In BLE, there is a trade-off between performance, security, and privacy concerns over low energy consumption [10] . As BLE protocol has low end-to-end security, an attacker can take over or disrupt a whole network of BLE devices deployed in an industrial automation system. [11]

Security becomes one of the major keys toward which, due to the importance of BLE communications, there has to be a formidable encryption method. Different mechanisms can be in place to secure data in BLE. These are:

- AES-CCM: Advanced Encryption Standard (AES) with Counter with CBC-MAC- This encryption method is used in BLE to secure the sent data packets. It provides

confidentiality and integrity by encrypting the payload and offering message authentication code, Media Access Control (MAC), authenticating the source of the message.

- LE Secure Connections: Introduced by Bluetooth 4.2, LE Secure Connections enhances security during pairing, using Elliptic Curve Diffie-Hellman key exchange for greater protection against eavesdropping and man-in-the-middle attacks.
- Authenticated Payloads: BLE can also use authenticated payloads to ensure that the data exchanged between devices has not been tampered with. In applications where ensuring the integrity of data is of great consequence, this is a very important point.

When implemented, these encryption methods secure sensitive information across transmissions between BLE devices and provide a much more difficult avenue of approach with regard to intercepting or manipulating the data by unauthorized users.

2.1.2 LoRa

LoRa (from "Long Range") is a physical proprietary radio communication technique. [12]It is based on spread spectrum modulation techniques derived from chirp spread spectrum (CSS) technology. [13] It uses chirp pulses to encode information on radio waves, much like bats and dolphins do. LoRa-modulated transmission is robust against disturbances and can be received across great distances. [14]

LoRaWAN is a Media Access Control (MAC) layer protocol on top of the LoRa modulation. It's essentially a software layer that dictates how devices use the LoRa hardware—when, for example, they should transmit and the format of messages. [15]LoRa and LoRaWAN together define a Low-Power Wide-Area (LPWA) networking protocol designed to connect battery-operated devices to the Internet wirelessly in regional, national, or global networks. This technology targets key IoT requirements, including bi-directional communication, end-to-end security, mobility, and localization services. This type of network is distinguished from a wireless wide area network (WAN) by characteristics including low power, low bit rate, and IoT use; the latter is designed to connect users or businesses and carry more data using more power. Data rate in LoRaWAN: 0.3 to 50 kbit/s per channel. LoRa uses license-free sub-gigahertz radio frequency bands: EU868 (863–870/873 MHz) in Europe; AU915/AS923-1 (915–928 MHz) in South America; US915 (902–928 MHz) in North America; IN865 (865–867 MHz) in India; and AS923 (915–928 MHz) in Asia. [16] Applications for LoRa will suit those initializing small chunks of data

at low bit rates. It can also be used to transmit data at a longer range compared with technologies like Wi-Fi, Bluetooth, or ZigBee. These features make LoRa well-suited for sensors and actuators operating in low-power mode. The technology covers the physical layer; other technologies and protocols, like LoRaWAN, cover the upper layers. It can achieve data rates between 0.3 kbit/s and 27 kbit/s, depending upon the spreading factor. LoRa protocol specification is developed by the LoRa Alliance. [17]The end-to-end network protocol architecture is shown below. LoRaWAN's protocol consists of a MAC Layer and an Application Layer, and it operates based on the LoRa physical layer. [18]

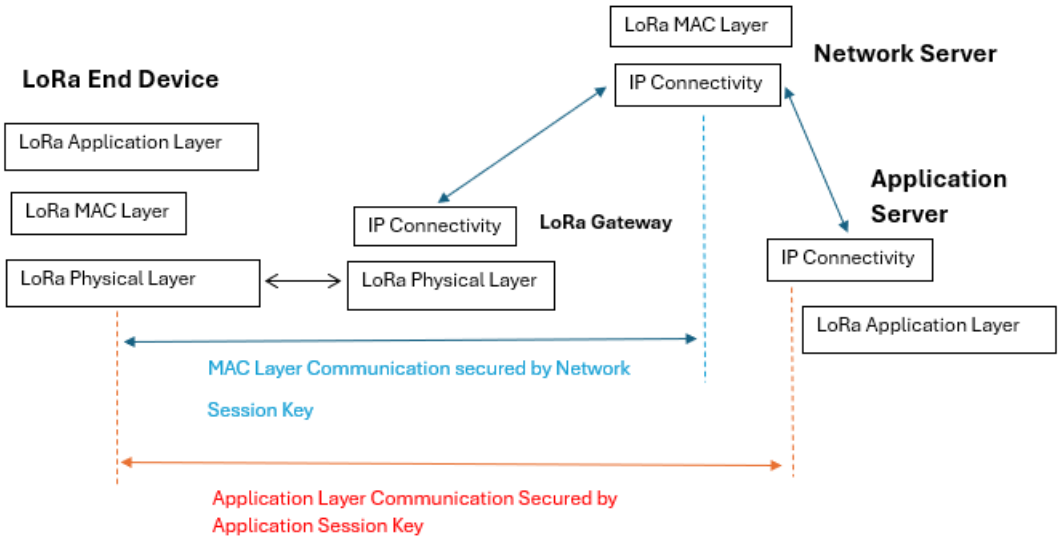

Figure 3 LoRa Network Protocol [19]

The physical layer of LoRa is patent-protected; hence, there's no freely available official documentation, except for the overview published by Sentech and other relevant technical specifications. LoRa uses proprietary spectrum modulation, which is either identical or a derivative of Chirp Spread Spectrum (CSS) modulation. This will permit LoRa to stabilize its data rate by choosing and fixing the spread amount (it can be chosen between 7 and 12). This propagation factor determines the data rate and the sensitivity of the radio, as well as the required energy consumption.

An end node in a LoRa network with a star topology can communicate with the network server by sending messages to several gateways. A message provided by end devices may be received by multiple gateways since an end node is not associated with a specific gateway. An end device and the gateways communicate via LoRa radio access technology.

Standard internet protocol (IP) connections are used to link the gateways and network server.

- End Device: A LoRa End device is a device that sends small amounts of data at low frequencies over long distances. Applications can be smart cities, smart buildings, factory automation, farm automation, and logistics.
- LoRa Gateway: The LoRa base transceiver station (BTS) transfers packets to the network server via IP backhaul or 3G/4G broadband connections after receiving them from the end node via a radio link.
- Network Server: The network server manages the entire network. [20]Upon receiving packets, it removes the redundancy of packets carries out a security check, and then decides the most appropriate gateway to send back an acknowledgment message.
- Application Server: It is the ultimate server where all the data sent by the End Device are post-processed and necessary action is taken.

LoRa is designed for long-distance communication and is very power-efficient; it shall be ideal for IoT applications needing intermittent data transfer. LoRa itself contains a set of limitations:

- Low Data Rate: LoRa operates on a low data rate, usually 0.3 kbit/s and 27 kbit/s. This makes it unsuitable for applications requiring high-speed or a large volume of data transmission. In this project, only the essential 'presence' data of devices is transmitted over LoRa, where it becomes quite possible to perform so under LoRa's data rate constraints.
- Susceptibility to Network Congestion: LoRa operates in license-free frequency bands that are susceptible to potential interference, especially in urban areas with numerous LoRaWAN networks. This can lead to delays and even lost messages, which might have implications for the dependability of a system deployed in a high-density environment.

Because LoRa is more suitable for sending minimal data on the presence over longer distances, its possible congestion and low data rate may impact real-time responsiveness in dense environments. To mitigate this, if available, the system uses Wi-Fi to upload to the cloud for good connectivity, ensuring access to data in a reliable way for real-time monitoring.

2.1.3 WI-FI and ThingSpeak

Wi-Fi is a technology that allows many electronic devices to exchange data or connect to the internet wirelessly using radio waves. The Wi-Fi Alliance defines Wi-Fi devices as any "Wireless Local Area Network (WLAN) products that are based on the Institute of Electrical and Electronics Engineers' (IEEE) 802.11 standards". [21]As of 2017, the Wi-Fi Alliance consisted of more than 800 companies from around the world. [22]As of 2019, over 3.05 billion Wi-Fi-enabled devices are shipped globally each year. [23]Wi-Fi uses a number of components from the IEEE 802 protocol family and, being the wireless counterpart of Ethernet, has been developed to function seamlessly with its wired sibling. Compatible devices can network through wireless access points with each other and with wired devices and the Internet. Various versions of Wi-Fi are specified by different IEEE 802.11 protocol standards; radio technologies determine radio bands, maximum ranges, and speeds that may be achieved. The most widely utilized radio bands for Wi-Fi are 2.4 GHz (120 mm) UHF and 5 GHz (60 mm) SHF; later iterations of the standard used the 6 GHz SHF spectrum. There are several channels within these bands. Although networks can share channels, only one emitter within range can be transmitted at a moment on a given channel.

The 802.11 standard identifies several different radio frequency bands that can be used for Wi-Fi communications: 900 MHz, 2.4 GHz, 3.6 GHz, 4.9 GHz, 5 GHz, 6 GHz and 60 GHz bands. These are divided into dozens of channels. Channels are numbered in the standards at 5 MHz spacing within a band (except in the 60 GHz band, where they are 2.16 GHz apart), and the number refers to the centre frequency of the channel. Although channels are numbered at 5 MHz spacing, transmitters generally occupy at least 20 MHz, and standards allow for channels to be bonded together to form wider channels for higher throughput. [24]Wireless communication implies that there is no use of wires in the transmission and reception of information using RF signals. The wireless communication protocols have become very popular in smart home networks because of the easiness in usability and reduced costs in setting up the network and installation of new devices. Some of the benefits associated with wireless communication over wire include:

- Mobility: as no physical linkage is needed to attach a device to a network, the device is free to move; moving the device to another wireless network is also effortless.

- Expandability: adding new devices to a network is easy as long as the maximum number of supported devices is not exceeded. Wireless networks are easy to scale up or down as needed with little or no costs.
- Costs: Setting up a wireless network is quite simple and can often be done without any professional help.
- Flexibility: setting up a wireless network in a new location is as simple as plugging the device into power; this makes it easy to experiment with new devices, or sensor placement.

  The disadvantages of wireless communication include:
- Security: although the current encryption mechanisms are strong, packets do travel through the air and can be intercepted, and possibly decrypted—although it's highly unlikely; most security issues arise when the wireless network is not protected at all, due to lack of proper configuration.
- Data rates: the theoretical speeds of wireless networks are lower compared to wired networks like Ethernet or Multimedia over Coax (MoCA). In practice, however, the data rates are most often adequate for most smart home applications.
- Interference: Wireless networks' (WLANs') signals are susceptible to interference, which lowers the network's quality of service.
- Coverage: theoretically, wireless networks have more coverage in a specific area than wired networks; however, obstacles or poor placement of the devices, [25] can decrease the coverage area, and lead to loss of commands/messages.

Table 1 Commonly used wireless protocols in smart home.

	Wi-Fi 802.11n	**Bluetooth**	**Bluetooth LE**	**ZigBee**	**Z-Wave**	**GlowPAN**
Frequency [GHz]	2.5-5.8	2.402-2.48	2.402-2.48	868/915 MHz , 2.4 GHz	868/915 MHz	868/921
Data rate [Mbps]	450	0.7-2.1	2	20/40 kbps , 250 kbps	10 – 100 kbps	10-40 kbps, 250 kbps
Range [m]	10-100	15-20	10-15	10-100	30-50	10-100
Network site	Thousands (mesh)	8	N/A	65,536	232	250
Network Topology	Star , tree , P2P , mesh	Star	Star	Star , mesh , cluster tree	Mesh	Star , mesh , P2P
Encryption	WPA2	AES-128	AES-128	AES-128	AES-128	AES-128

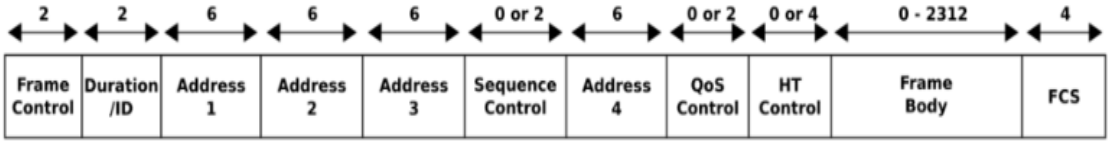

Figure 4 IEEE 802 Communication Stack [26]

Wi-Fi is part of the protocol family IEEE 802. On the data link layer, the data is framed into 802.11 frames. These are much like the Ethernet frames but with some extra address fields. On LANs, for routing purposes, the network addresses are MAC addresses. IEEE 802.11 defines the MAC and PHY specifications of Wi-Fi for modulating and receiving one or more carrier waves with a frequency in the infrared and the 2.4, 3.6, 5, 6,

or 60 GHz frequency bands to carry all the data. They are developed and maintained by the IEEE LAN/MAN Standards Committee (IEEE 802).

Various encryption protocols are applied within the Wi-Fi network for safeguarding the data across the wireless medium from unauthorized access and ensuring integrity. Among them, major encryption standards are:

- WEP- Wired Equivalent Privacy: This is the very first security on Wi-Fi and has basic encryption. It is, however, even today very insecure and can be compromised with ease.
- WPA, or Wi-Fi Protected Access: Replacing WEP, WPA provides stronger security by implementing TKIP, or Temporal Key Integrity Protocol, which dynamically changed the key with each packet.
- WPA2: The upgraded version of WPA, WPA2 implements AES, or Advanced Encryption Standard, for encryption. Thus, it provides a more solid security framework. WPA2 is more common and widely recommended in securing Wi-Fi networks.
- WPA3: The newest and the latest security protocol, WPA3, adds some new benefits-mostly about improved protection against brute-force attacks-and in the same moment provides far better protection for open networks due to opportunistic encryption.

All these protocols share the foundation in that they could make data packets transmitted over Wi-Fi encrypted, thus making the attempts of unauthorized users to intercept and decipher the information quite hard. Strong Wi-Fi security is absolutely essential in protecting sensitive data and ensuring user privacy.

ThingSpeak is an IoT analytics platform service that allows you to aggregate, visualize, and analyse live data streams in the cloud. It facilitates data access, retrieval and logging of data by providing an Application Programming Interface (API) to both the devices and social network websites. ThingSpeak was originally launched by ioBridge in 2010 as a service in support of IoT applications. [27]

ThingSpeak has integrated support from the numerical computing software MATLAB from MathWorks, [28]allowing ThingSpeak users to analyse and visualize uploaded data using MATLAB without requiring the purchase of a MATLAB license from MathWorks. [26]. Data is sensed by sensors or objects, which often take local action. Through

ThingSpeak, data may be transmitted from sensors, instruments, and websites to the cloud and stored in a private or public channel. The default in ThingSpeak is to store the data in private channels, but of course there are also possibilities to show data in public channels if you want to share data with others. Once data are in a ThingSpeak channel, you can immediately analyse and visualize them, calculate new data, or even trigger interactions with social media, web services, and other devices. Besides, under ThingSpeak, storage of cloud data gives you access to your data. This permits the exploration and visualization of data using analytics online. In data, you can find relationships, patterns, and trends. New data can also be calculated. Plots, charts, and gauges can be used to visualize it; ThingSpeak offers tools that facilitate device communication for all of these purposes and more. You can queue up commands for a device to perform and respond to data as it enters a channel, both raw data and newly calculated data.

```
try
{
 const string WRITEKEY = "YOUR_KEY";
 string strUpdateBase = "http://api.thingspeak.com/update";
 string strUpdateURI = strUpdateBase + "?key=" + WRITEKEY;
 string strField1 = "18";
 string strField2 = "42";
 HttpWebRequest ThingsSpeakReq;
 HttpWebResponse ThingsSpeakResp;
 strUpdateURI+= "&field1=" + strField1;
 strUpdateURI+= "&field2=" + strField2;
 ThingsSpeakReq = (HttpWebRequest)WebRequest.Create(strUpdateURI);
 ThingsSpeakResp = (HttpWebResponse)ThingsSpeakReq.GetResponse();
 if(!(string.Equals(ThingsSpeakResp.StatusDescription,"OK")))
 {
 Exception exData = new Exception(ThingsSpeakResp.StatusDescription);
 throw exData;
 }
}
catch (Exception ex)
{
 lblError.InnerText = ex.Message;
 lblError.Style.Add("display", "block");
 throw;
}
```

Figure 5 ThingSpeak Code Example

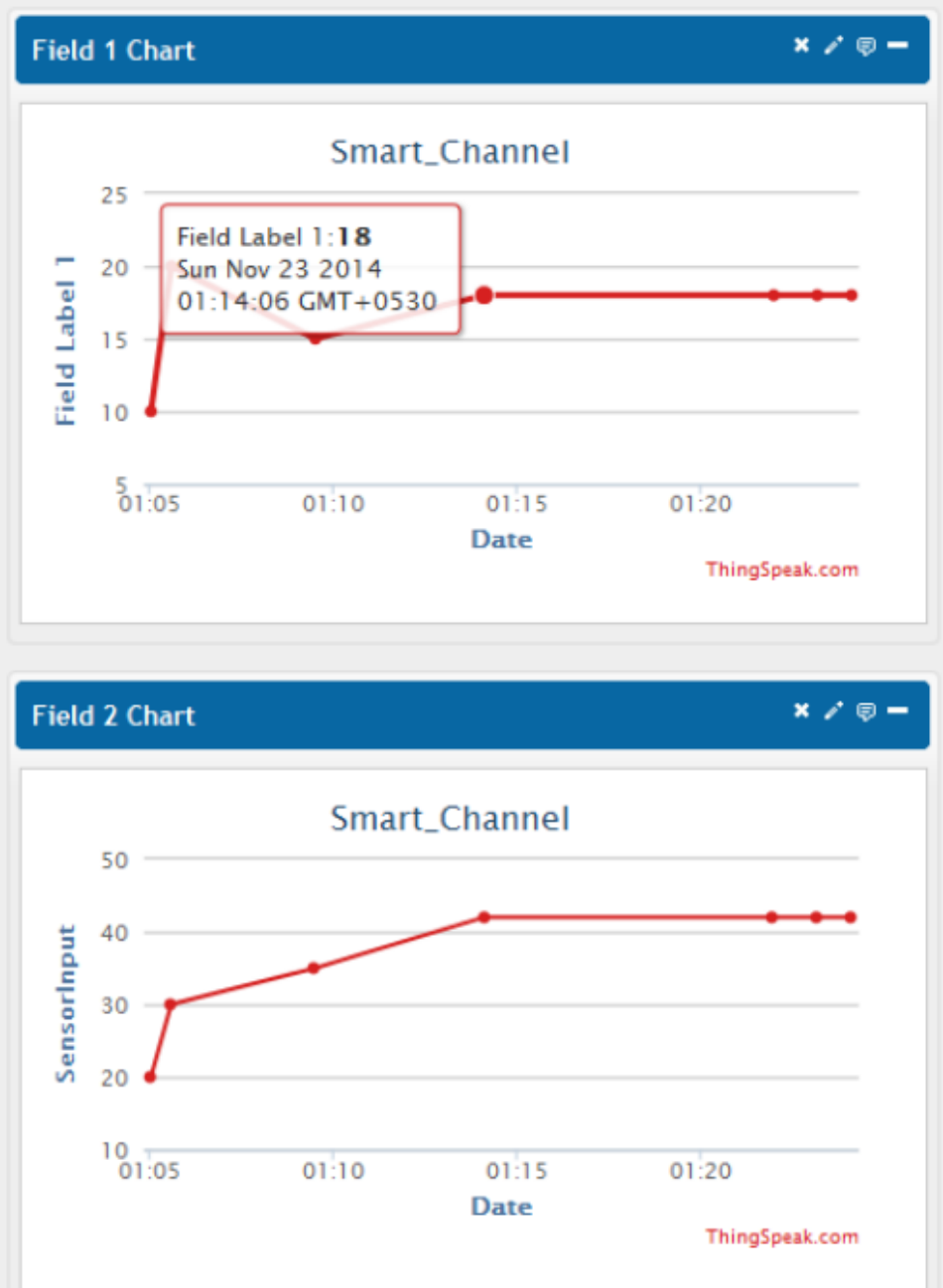

Figure 6 Figures shown in ThingSpeak with provided code [28]

2.2 Existing Systems and Solutions

2.2.1 Wirepas Mesh

Wirepas has developed Wirepas Mesh-a radio frequency-based, fully meshed network architecture for IoT solutions. Within the Wirepas Mesh network, every device independently makes autonomous routing decisions based on the current radio situation. The decentralized architecture of Wirepas Mesh supplies high data transmission reliability and availability, giving coverage across big areas. All of the intelligence of Wirepas Mesh

is decentralized into the network and onto the devices themselves. The devices make autonomous routing decisions based on radio spectrum and energy availability. A central network hub device is not needed in Wirepas Mesh. Wirepas devices can transmit data over multiple hops from one node to another and towards the Cloud and back. [29]For each device, multiple routing options are always present, and several IoT devices may be used in the same network. BLE chips, such as Nordic Semiconductor's nRF52 family and Silicon Labs' EFR32 family, run Wirepas Massive. It actually shares the very same physical layer as the BLE hardware. However, Wirepas Massive overwrites the Bluetooth firmware still keeping the ability to transmit and receive BLE Beacon messages.

Table 2 Comparison between BLE Mesh and Wirepas Mesh [30]

	BLE Mesh	**Wirepas Mesh**
BLE Beacon transmit	Yes	Yes
BLE Beacon scanning	Yes	Yes
Flooding	Yes	Yes
Unlimited hop count	Yes	Yes
Routing	No	Yes
Battery operated routers	No	Yes
Signalling encrypted	No	Yes
Acknowledgements on network level	No	Yes
Number of channels used in one mesh network	3	40

Unlike the great majority of mesh connectivity technologies, BLE Mesh does not use routing. [30]It is called a flooding mesh, which relies on repeaters that re-transmit messages heard from devices in the network. Those repeaters also need to be mains-powered because they consume a lot of energy. These limitations are acceptable, for example, in pure wireless lighting control where devices are mains-powered and the switch command is broadcasted to all devices. [30]But this limitation makes it completely unsuitable for sensor networks bigger than 100 devices. Let me explain. BLE has an in-built BLE Beacon advertisement, which is designed to let devices find one another and make a pair. Precisely this pairing functionality has been repurposed in the BLE Mesh, where there are devices

advertising beacons and receiving and repeating beacon messages with data. Devices will repeat the same message typically three times in three channels, so at least the message is heard somewhere. The data may reach a gateway. To sum it up, Wirepas Massive firmware has adaptive routing and adaptive flooding. This means intelligence inside the network, and that it makes the right decisions on how to carry the message through the network to the gateway. It can handle tens of thousands of devices in one network without any problems. In battery-operated sensor networks, each device extends the coverage; hence, there is no wired or additional wireless infrastructure required with Wirepas Massive. This is one of the huge differences compared to BLE Mesh, where continuous BLE beacon scanning demands mains power for the repeaters. BLE Mesh requires you to build infrastructure to include mains-powered repeaters, which is expensive.

2.2.2 BeaconTrax

BeaconTrax is a leading provider of cloud-based software and technology solutions designed to improve efficiency and increase productivity of businesses. [31] BeaconTrax deals with the development of Bluetooth-based tracking systems. It provides solutions based on Bluetooth Low Energy for asset tracking and management, inventory control, and location services. Systems like these are also widely used in logistics and warehousing, healthcare, and manufacturing for tracking locations of assets in real-time so that facilitation of operations can be done more easily with increased efficiency. For this purpose, the company relies on BLE beacons and sensors to provide trusted tracking solutions for most situations. These include mobile devices, gateways, and software platforms to monitor and manage assets. BeaconTrax is a BLE tracking system designed for asset management, Real-time locating systems (RTLS), and monitoring within confined environments such as warehouses, factories, and hospitals. It uses BLE beacons that can broadcast signals to all BLE-enabled devices in their proximity, where proximity or an exact location could be calculated using the strength of the signal or Received Signal Strength Indicator (RSSI). This system works along with most mobile devices or some sort of gateway hardware, which collects and processes beacon signals and then sends real-time information on the movement of the tracked assets. BeaconTrax is used for inventory management, workflow optimization, and asset tracking visibility in industries for better performance. BeaconTrax has an enormous amount of tracking systems based on beacons utilisation such as: People tracking in Health Care, Manufacturing, construction site, BLE Beacon healthcare tracking solution is a cloud-based software for tracking patients and staff

members in real-time [31] BeaconTrax Beacon Asset Tracking System enables healthcare providers to know where critical assets are located in real-time inside their facility. This includes items such as, beds, wheelchairs, infusion pumps and biometric monitors, among others [32]Staff, and high-risk patients, like elderly patients who may wander, are also easily located in real-time. Regarding manufacturing aspect, with real-time visibility of raw materials, assets, machinery, and tools within a manufacturing plant, their innovative Industrial IoT solutions allow enterprises to automate their production processes. The cloud monitoring solution offers a unified platform for tracking personnel, vehicles, equipment, and assets in many locations. With the Blue-tooth beacon tracking system, you are able to easily manage your inventory, track their movement and receive various security and movement alerts, TraxLinQ location system provides both real time visibility and actionable data on assets and tools within the manufacturing processes. [33]

Besides, it was also possible to collect environmental data using sensor beacons for maintaining temperature and humidity inside the facility. In addition to the devices of beacon and gateway, this software will allow you to track precisely, manage attendance, monitor security, and keep your operations safe at your facilities in healthcare.

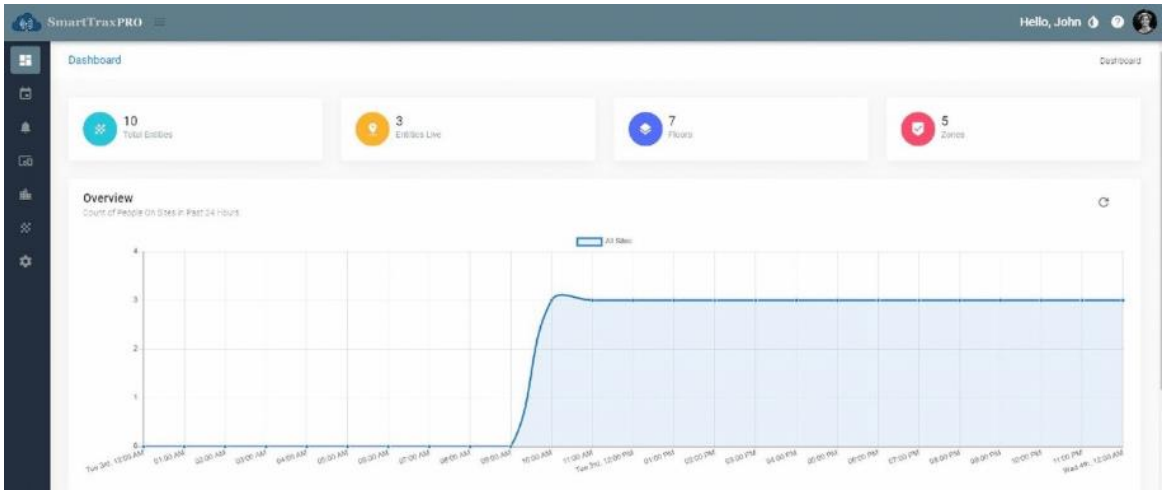

Figure 7 BeaconTrax tracking application based on BLE and beacon tracking [36]

2.3 Challenges And Applications

BLE asset-tracking devices are not very inert to withstand harsh industrial environments and detect rapid movements. The battery-life is further limited to 1-year intervals when sending location data on a weekly basis. Commercial tracking can be somewhat limited using a BLE tracker, but it has considerable advantages indoors and in the case of personal

tracking within an urban environment. BLE tracking has been used successfully in the following industries:

- Medical Healthcare-Trace medical equipment to immediately locate the most critical devices and manage them precisely in health facilities. Example: BLP-blood pressure measurement; HTP-measuring medical temperature devices; GLP-blood glucose monitor; CGMP-continuous glucose monitor
- Retail: Updates the inventory management in real time; helps to avoid stockout and over-inventory. Smart Buildings: BLE technology allows indoor navigation in smart buildings, aids in way finding in big complexes as it also tracks furniture and equipment in their proper usage. HOGP (HID over GATT Profile) allowing Bluetooth LE-enabled Wireless mice, keyboards and other devices offering long-lasting battery life.
- Industrial Applications: Monitoring tools and equipment on a small site; decreasing downtimes; increasing efficiency; and improving control over appropriate tool use and maintenance such as: User Data Service (UDS); Environmental Sensing Profile (ESP)
- Sports and fitness profiles: BCS, CSCP (which mounts on a bike or exercise bike to assess cadence and wheel speed), CPP, etc.

Apple AirTag is about the size of a quarter. It is excellent for personal item BLE tracking because this is a convenient solution for the consumer who already uses an Apple device, for example, an iPhone. Apple AirTags communicate location when within proximity to a neighbouring Apple iPhone as part of the Find My network. While this enables users to view the location of their connected asset, limitations exist for accessing location data if assets are in a remote location. Accessibility of data is further limited by AirTag data being available to a user only via a connected Apple device. [34] This inhibits publishing and accessing location information to other sources, which is essentially a must in commercial settings. Commercially speaking, there needs to be asset visibility; therefore, solutions which cannot constantly depend on infrastructure systems-such as Bluetooth gateways-to send out location data become paramount. Besides location, tracking organizations for high-value assets also need to have further data and insights, including tamper, theft, and impaction detection. This is usually very limited with BLE tags, while GPS devices can do much more.

Besides, durability and lifetime are also important features for an organization when choosing a tracking technology. BLE asset-tracking devices have limited capabilities to resist the harsh industrial atmosphere and detect rapid movements. Their battery-life lasts just for 1 year sending location data once in a week. Many organizations need asset location information more than once a day, which in the case of BLE tracking devices would result in frequent battery changes. Although the actual cost of an Apple AirTag battery is inexpensive, it is often not practical to physically locate and replace batteries once the assets are deployed, especially if tracking a large number of assets.

Therefore, BLE tracking is associated with the following drawbacks: Limited Range: It is effective only for short-range tracking that also can be only a couple of hundred feet, which restricts its application in the large-scale or outdoor scenarios. Interference: Signals can be interfered with by other Bluetooth devices to affect its accuracy and reliability. Security Vulnerabilities: There is no proper security measure associated that may be resulting in BLE network vulnerability to unauthorized access and data breaches. Dependence on BLE-enabled devices: BLE-enabled devices and infrastructure are needed to track. That again creates difficulties for tracking in certain environments.

3 IMPLEMENTATION AND TESTS

3.1 System Architecture and Component Configuration

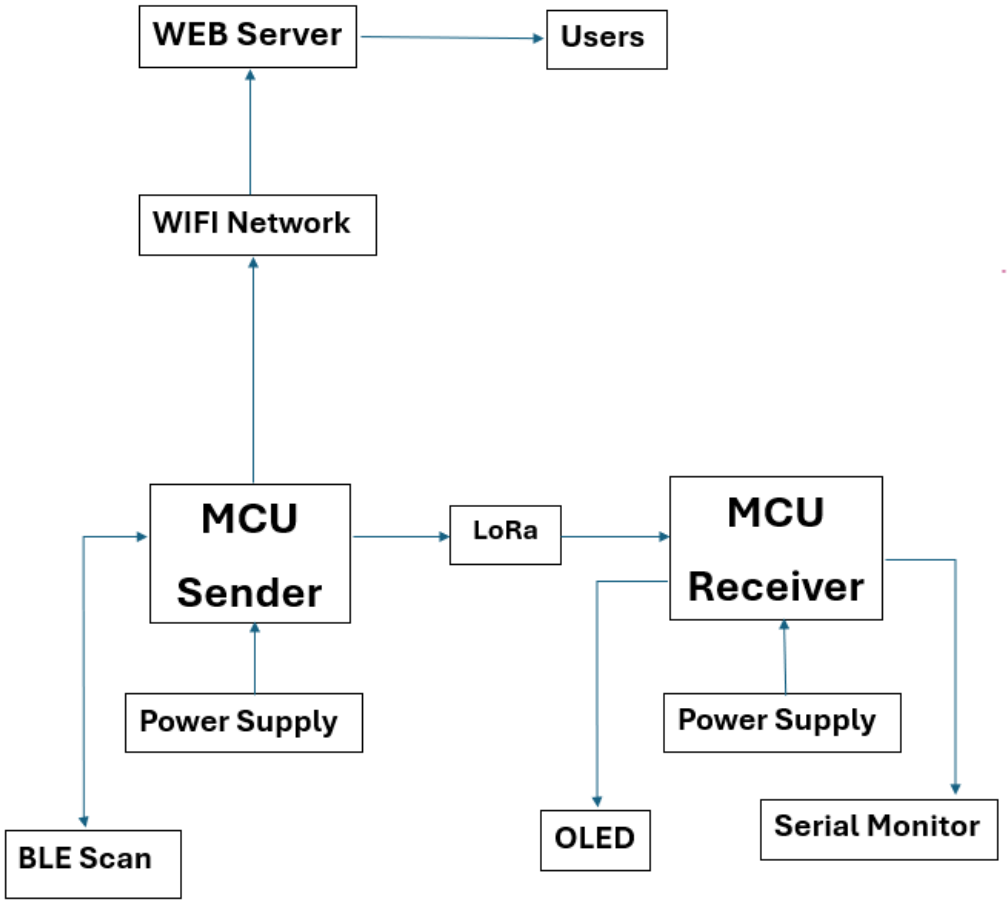

Figure 8 Logical System

This system's logical design shows how various components come together to enable presence tracking via BLE, LoRa, Wi-Fi, and a web server. The MCU sender, powered by a portable power bank, performs a BLE scan of devices in the environment and further transmits the collected data over LoRa to the MCU receiver. The receiver is battery-powered using a laptop and displays data on an OLED screen, logging to a serial monitor for further analysis and debugging. The sender also connects to a Wi-Fi network, uploading the presence data onto a web server accessible by users remotely. It allows continual low-power operation of the sender, real-time data monitoring and transmission, and uses the special strengths of LoRa for long-range communication and Wi-Fi for data storage on a cloud platform. Portable power sources—the power bank for the sender and the laptop for

the receiver—guarantee flexibility and mobility in the deployment of this presence-tracking system.

3.1.1 Mobile BLE Tracking Unit (Sender Device)

The core microcontroller of the sender device's hardware setup is the Heltec LoRa32 V2, where BLE scans are performed and sent via LoRa and Wi-Fi. Wi-Fi LoRa 32 is an IoT dev-board built and designed by Heltec Automation TM; it's a highly integrated product based on ESP32-S3 + SX1262. The item has available Wi-Fi BLE and LoRa capabilities but also includes Li-Po battery management systems and 0.96" OLEDs.

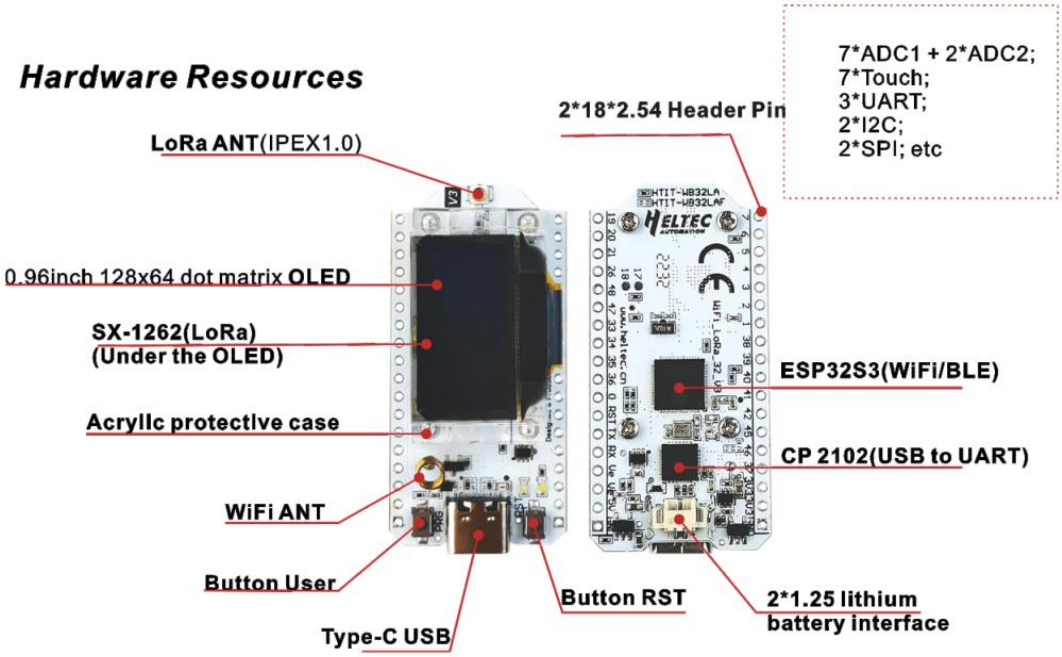

Figure 9 Heltec Lora32 V2.1 [35]

Table 3 Power Supply of Heltec LoRa32 V2.1

Power supply mode	**Minimum**	**Typical**	**Maximum**	**Company**
Type-C USB (>= 500 mA)	4.7	5	6	V
Lithium battery (>= 250 mA)	3.3	3.7	4.2	V
5 V pin (>= 500 mA)	4.7	5	6	V

The Heltec LoRa32 V2.1 can be run on a standard power bank that can charge a mobile phone, much like most ESP32-based boards. The Heltec LoRa32 V2.1 has an onboard voltage regulator, allowing the board to be powered over its USB port with 5 V input, which is the standard for most power banks. Heltec LoRa32 V2.1 is designed to be powered through its USB port, which usually accepts 5 V; hence, USB outputs found in power banks will be perfect. The onboard voltage regulator will safely convert the 5 V USB power from the power bank down to the 3.3 V needed by the ESP32 chip and other components on the board. Provided that my power bank is able to charge an iPhone, it really ought to be able to supply sufficient current-at least 1 A, but desirably 2 A or more-for the Heltec LoRa32 V2.1, which generally consumes less than 1A even during high-power LoRa transmissions.

Using BLE, LoRa, and Wi-Fi all at once on the Heltec LoRa32 V2.1 board, the power consumption is higher; the input voltage remains 5 V via USB. Every technology required a supplementary current when it is activated, especially for data transmission. BLE would draw about 15-30 mA during normal operation (that is, during scanning or low-power operation), though this is subject to variation. LoRa can consume around 120-150 mA when it transmits at maximum power. Wi-Fi is power hungry and draws 200-250 mA of current during active transmission, especially at high power modes. When all three are working simultaneously, peak current use can be in the range of 400-500 milliamps or higher. The actual use may be subject to alteration depending on various factors such as transmission time intervals, power levels, and duty cycles of each type of communication. Since the power bank is to charge an iPhone, it should provide at least 1 A at 5V, which shall be sufficient in this case. In such cases, however, a power bank with an output current rating of 2 A or higher is preferable for stability and prevention of possible restarts or power drops at peak loads.

3.1.2 Stationary Data Reception Unit (Receiver Device)

Standard across all USB ports, its output voltage is 5 V, while actual current output may vary with port and device load; some USB 3.0 ports are capable of higher currents up to 1.5 A, or higher on specific charging ports. Therefore, this chip may work perfectly based on the data coming from the previous sender device, Heltec Lora32 v2.1.

3.2 Software Design and Implementation Framework

3.2.1 Bluetooth Low Energy (BLE) Device Scanning and Identification

Scanning for BLE devices in this project is implemented through the Arduino BLE library that can detect devices around it, including beacons. It initializes the BLE device first, then creates an object to manage the scanning. The scanning parameters are set to active scanning; the device will request responses from advertisements around it. The callback function is invoked for every detection of an advertised device during a scan. It retrieves essential information from that, like device names, addresses, and service Universal Unique IDentifier (UUIDs), which get logged. The implementation also contains the ability to find Eddystone URL beacons where the decoded Uniform Resource Locator (URL) is retrieved and stored for transmission. It provides an efficient scanning mechanism with the capability to monitor all the BLE devices present in its vicinity, which forms an important part of the project's presence tracking objectives.

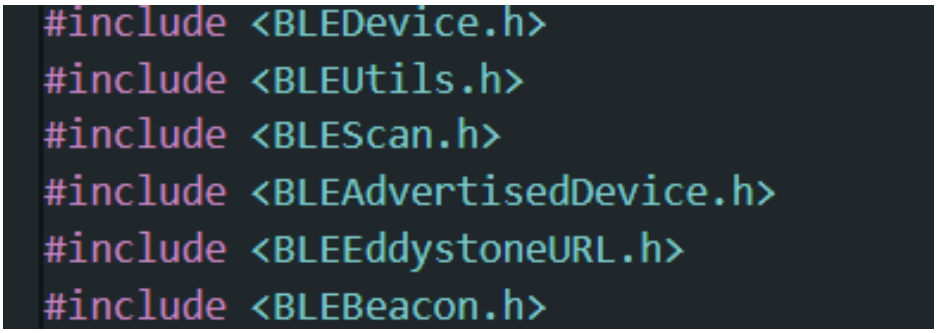

Figure 10: Libraries for BLE

```
int scanTime = 10; // Scan duration for each BLE scan
BLEScan *pBLEScan;
int deviceCount = 0; // Flag variable to track the number of device count
String txData; // Declare a String varible to store information of detected devices
```

Figure 11 Variables utilized in BLE codes

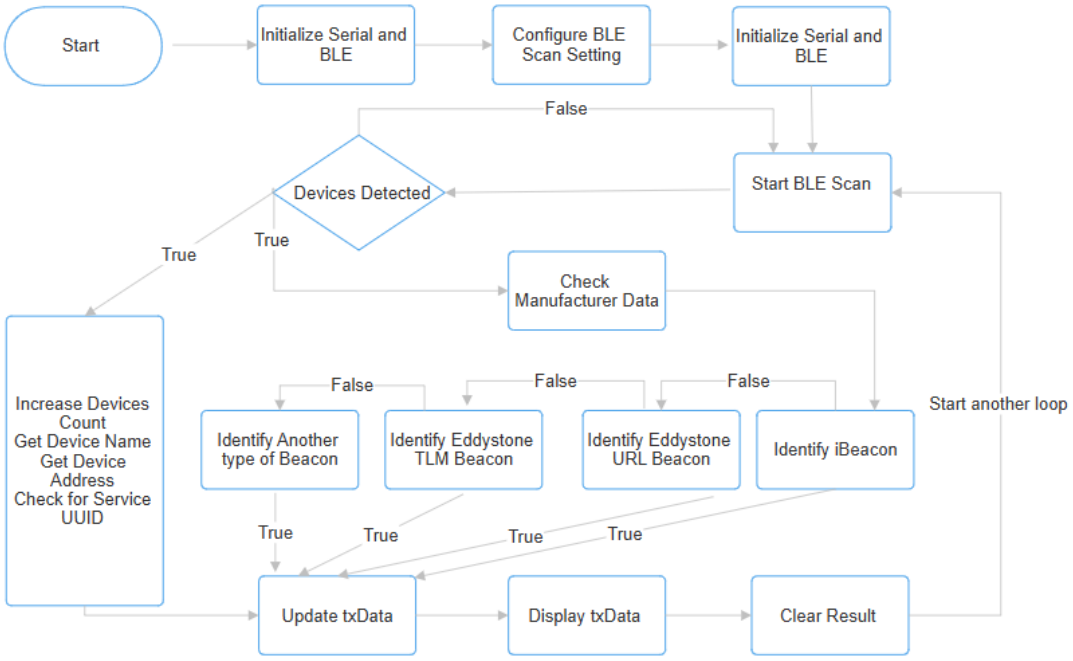

Figure 12 BLE Workflow

The following code is intended to scan and log key information from BLE devices in proximity to allow some form of presence detection. The idea here would be that we might expect this code to do and set up for capture the expected information from devices: device name and address. Each time a BLE device is discovered, we will extract and log its name-if present-and address. This allows the retrieval of a unique identifier of individual devices in the area; Service UUIDs: In case the device advertises its service UUID, then it will appear here. Knowing some service UUIDs will allow you to have a hint about what for the device is used. For example, there is a standard UUID for heart rate monitors, while environmental sensors have different UUIDs; Manufacturer Data: This chunk of data often contains particular information on the device type and model. Notably, Apple devices use specific bytes in their manufacturer data for iBeacon identification; thus, this code checks whether the manufacturer data starts with the bytes 0x4C and 0x00, indicating that it is an Apple iBeacon. ; Identification of Beacon: It precisely identifies the following: iBeacons: If the manufacturer data fits Apple's format, it logs the finding of an iBeacon. Eddystone Beacons: It checks every advertisement's FrameType to detect Eddystone URL beacons-those carrying URLs-and Eddystone TLM beacons, which contain telemetry data. Decoded Eddystone URLs If there is any kind of Eddystone URL beacon, it decodes and logs the

URL that is currently being broadcast. This could come in handy in cases where the beacons link to a website or some other online resource. This code will enable real-time presence tracking of BLE devices coming within range. Finding the devices based on their advertised data, this code delivers on the following: Device Discovery and Identification: This will allow for the detection of all nearby BLE devices and capture key identifying information about each. This is really important in situations you want to monitor BLE devices entering or leaving a defined area. Beacons Recognition: This code will be able to identify and segregate regular beacon types, specifically iBeacons and Eddystone beacons. This can be used in tracking or proximity-based applications where a few beacons may bear some special tasks, such as location-based services or events. Gathering Data for Analysis: This data collection may serve as the ground for analysing the presence patterns and density of BLE devices in an area. For instance, several detections of the same device may hint at it being stationary or regularly visiting, while brief detections may just be passersby.

3.2.2 Data Transmission via LoRa for Long-Range Communication

Data transmission via LoRa in the project has been enabled by a configuration and event-driven mechanism of parameters to ensure solid communication over long distances. The initial configurations of the LoRa module are done with essential parameters like frequency, transmission power, bandwidth, spreading factor, and coding rate. Setting these

```
21  #define RF_FREQUENCY                                868000000 // Hz
22  #define TX_OUTPUT_POWER                             5        // dBm
23  #define LORA_BANDWIDTH                              0         // [0: 125 kHz,
24  #define LORA_SPREADING_FACTOR                       7         // [SF7..SF12]
25  #define LORA_CODINGRATE                             1         // [1: 4/5,
26  #define LORA_PREAMBLE_LENGTH                        8         // Same for Tx and Rx
27  #define LORA_FIX_LENGTH_PAYLOAD_ON                  false
28  #define LORA_IQ_INVERSION_ON                        false
29
30  #define RX_TIMEOUT_VALUE                            1000
31  #define BUFFER_SIZE                                 256
32  // Increased buffer size for BLE data
33
34  char txpacket[BUFFER_SIZE];
35  char rxpacket[BUFFER_SIZE];
36  double txNumber;
```

Figure 13 LoRa Configuration

configuration settings is important for optimizing the transmission range and data fidelity so that a system can ensure effective communication under difficult conditions.

After successful initialization of the LoRa module, the system enters a processing loop where data from the BLE scanning operations is aggregated. Upon fulfilment of predefined criteria, such as the detection of a certain number of BLE devices, the gathered data is then organized into a packet ready for transmission; this packet carries information about the recognized devices, such as the quantity and other identifying attributes of interest. It is using Radio.Send, which will send the framed packet over the LoRa network in bytes to efficiently communicate with a device on the receiving end. Once the transmission is done, an event callback is invoked with an event confirming that this send action of data has been accomplished and the system is ready for the next sends. This, without doubt, enriches the whole project with some functionality-like BLE tracking data communication over a long-range using LoRa technology. That's quite vital in presence tracking across environments that could not be ignored.

The above procedure is for the Sender device; as for the receiver devices, the packet sent via LoRa would be extracted by using LoRa and Arduino.h library functions.

In the receiver code, depacketizing LoRa packets from the sender involves the almost careful extraction and interpretation of every segment in the incoming data. Upon receiving a packet, the receiver listens on the specified LoRa frequency and retrieves the data payload sent from the sender device. Each packet is structured such that the receiver identifies among others, BLE device information, and any specific identifier included in the payload. Parsing functions are then used at the receiving device to break down the contents of the packet and map them to predefined variables or data structures that assure accurate reading and interpretation of each element. For example, in the case of BLE scan results sent by the sender, the parameters that can be extracted from the receiver include device identifier, signal strength, and type. Depacketizing is important in ensuring that data is actually

decoded reliably and further processed for storage, analysis, or display on a connected platform such as ThingSpeak.

```
void OnRxDone(uint8_t *payload, uint16_t size, int16_t rssi, int8_t snr) {
    // Store RSSI and size
    int16_t receivedRssi = rssi; // Changed to use the passed rssi instead of local rssi
    rxSize = size;

    // Copy received payload to rxpacket
    memcpy(rxpacket, payload, size);
    rxpacket[size] = '\0'; // Null-terminate the received string

    // Print received packet as a string
    Serial.printf("\r\nReceived packet \"%s\" with rssi %d, length %d\r\n", rxpacket, receivedRssi, rxSize);

    // Debug: Print raw received data in hex format
    Serial.print("Raw received data in hex: ");
    for (int i = 0; i < rxSize; i++) {
        Serial.printf("%02X ", (uint8_t)rxpacket[i]);
    }
    Serial.println();

    // Process and print the received data
    String receivedData = String(rxpacket);
    Serial.println("Received Data:\n" + receivedData);

    // Extract specific fields from the received data
    if (receivedData.indexOf("Device name:") != -1) {
        String deviceName = receivedData.substring(receivedData.indexOf("Device name:") + 12);
        deviceName = deviceName.substring(0, deviceName.indexOf('\n'));
        Serial.println("Extracted Device Name: " + deviceName);
    }
```

Figure 13 Receiver device OnRxDone Function to extract the packet sent

```
    if (receivedData.indexOf("Device address:") != -1) {
        String deviceAddress = receivedData.substring(receivedData.indexOf("Device address:") + 15);
        deviceAddress = deviceAddress.substring(0, deviceAddress.indexOf('\n'));
        Serial.println("Extracted Device Address: " + deviceAddress);
    }

    // Optionally, handle additional fields from the sender data
    if (receivedData.indexOf("Found ServiceUUID:") != -1) {
        String serviceUUID = receivedData.substring(receivedData.indexOf("Found ServiceUUID:") + 18);
        serviceUUID = serviceUUID.substring(0, serviceUUID.indexOf('\n'));
        Serial.println("Extracted Service UUID: " + serviceUUID);
    }

    if (receivedData.indexOf("Manufacturer Data:") != -1) {
        String manufacturerData = receivedData.substring(receivedData.indexOf("Manufacturer Data:") + 18);
        manufacturerData = manufacturerData.substring(0, manufacturerData.indexOf('\n'));
        Serial.println("Extracted Manufacturer Data: " + manufacturerData);
    }

    lora_idle = true; // Set LoRa back to idle
```

Figure 14 Receiver Device OnRxDone part 2

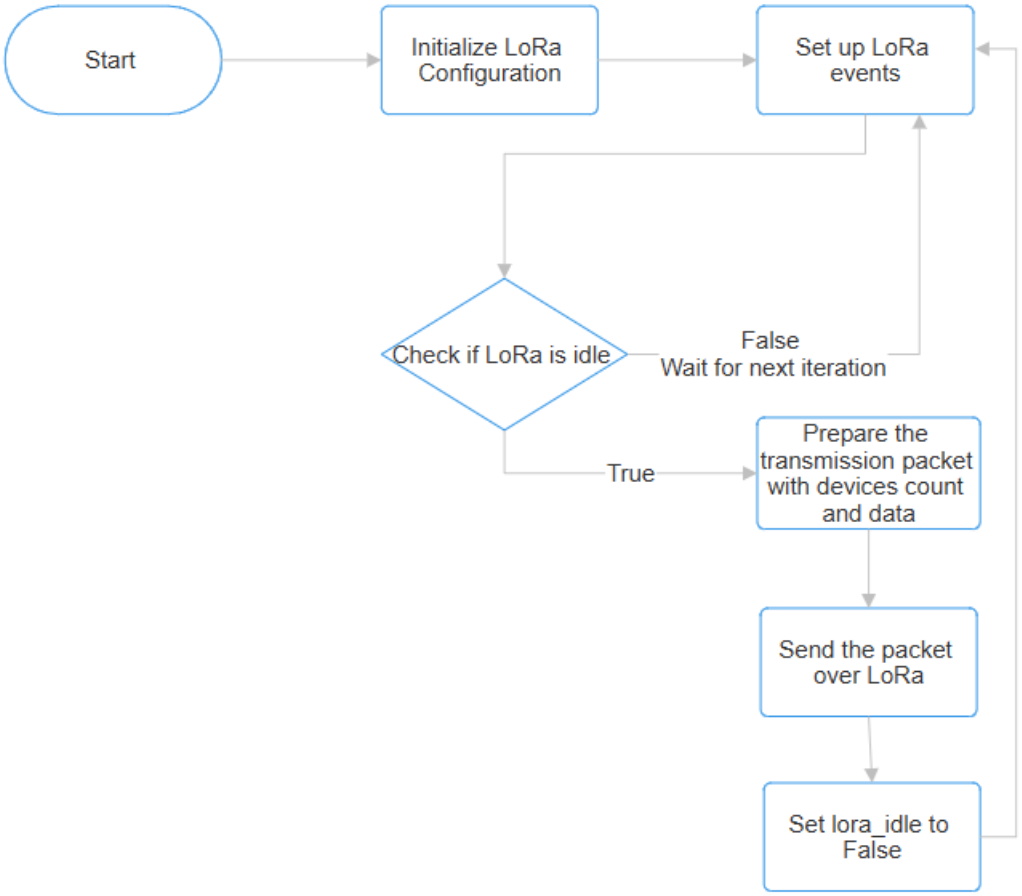

Figure 14 LoRa Sender Workflow

In addition, inside the BLE scan, the encryption of data can be provided for the sensitive data received. For instance, we can use Advanced Encryption Standard (AES). The algorithm of it is symmetric, which means that the same key will be used in encryption and decryption; this further really means that a single secret key is shared between the sender and the receiver. This type of encryption is used in various application: Applications like WhatsApp, Signal, and Telegram use AES to encrypt messages, ensuring privacy in communications. The HTTPS protocol uses AES for encrypting data transmitted between web browsers and servers, providing secure online transactions and communications. The AES encryption encrypted the data prepared to send by steps:

- The secret key and the initialization vector must be in blocks of 16 bytes for the AES-128 encryption process.
- Data to be Encrypted: This forms the actual data that needs to go over in plain text. For this example, we will use a simple string, "SensitiveData".

- Padding: If the data is not in multiples of the AES block size, then it needs to be padded into full block size.
- Encryption: The plaintext is turned into ciphertext by using the AES algorithm and the secret key. Indeed, AES operates by means of multiple rounds of substitution, permutation, and mixture of the data.
- Ciphertext: This finally results in encrypted data, now called ciphertext, that will be unreadable without the key. This is what will be sent over the network. Transmission: This encrypted data-ciphertext-is then sent out securely.

The below code snippet is the addition of AES encryption to the Sender device code .

```
153    if (lora_idle == true && !txData.isEmpty()) {  // Only send if there are valid data to transmit
154        delay(1000);
155        txNumber += 0.01;
156        char encryptedData[BUFFER_SIZE];
157        aesLib.encrypt(encryptedData, (const byte *)txData.c_str(), txData.length(), (const byte *)encryptionKey, (const byte *)iv);
158
159        snprintf(txpacket, BUFFER_SIZE, "Devices found: %d\n%s", deviceCount, encryptedData);
160        Serial.printf("Sending packet: \"%s\"\n", txpacket);
161        Radio.Send((uint8_t *)txpacket, strlen(txpacket));
162        lora_idle = false;
163    }
164    Radio.IrqProcess();
```

Figure 14 AES encryption addtion to the loop of sending packets of data

```
#include <AESLib.h>
#include <openssl/rand.h>

AESLib aesLib;
const char *encryptionKey = "YourSecretKey12";
const char *iv = "YourIV1234567890";

RAND_bytes(key, sizeof(key));
RAND_bytes(iv, sizeof(iv));
```

Figure 13 AES encryption integrated

3.2.3 Wi-Fi Network Integration for Cloud Data Upload to ThingSpeak

The Wi-Fi configuration used in this project connects the device with the ThingSpeak IoT platform for sharing data in real time. This is very important for further analysis of the collected data. First, there should be the establishment of a device connected to a network

provided by a certain service set identifier (SSID) and password. Once such connection is achieved, an IP address is allocated, which is confirmed through serial output for verification purposes. This will form the backbone of the connection to send data to ThingSpeak, where the individual reading-in this case, BLE device count-will be sent via an HTTP GET request. Data is formatted as an HTTP query with an API key and other particular data fields so that ThingSpeak can capture and log each reading on the channel. It reads the response incoming from ThingSpeak and prints that into the serial monitor to give feedback whether communication was successful or not-that the data is successfully transmitted. The setup here shows an IoT device that is capable of implementing Wi-Fi and cloud platforms for continuous monitoring and logging of data in order to enable data analysis and generation of insights. The code here implements the mechanisms to handle the response of the server and check whether there is good communication with the ThingSpeak server. This integration with Wi-Fi capabilities not only allows real-time data transmission but also adds to the works of the Bluetooth tracking system: continuous data analysis and monitoring, this time through ThingSpeak.

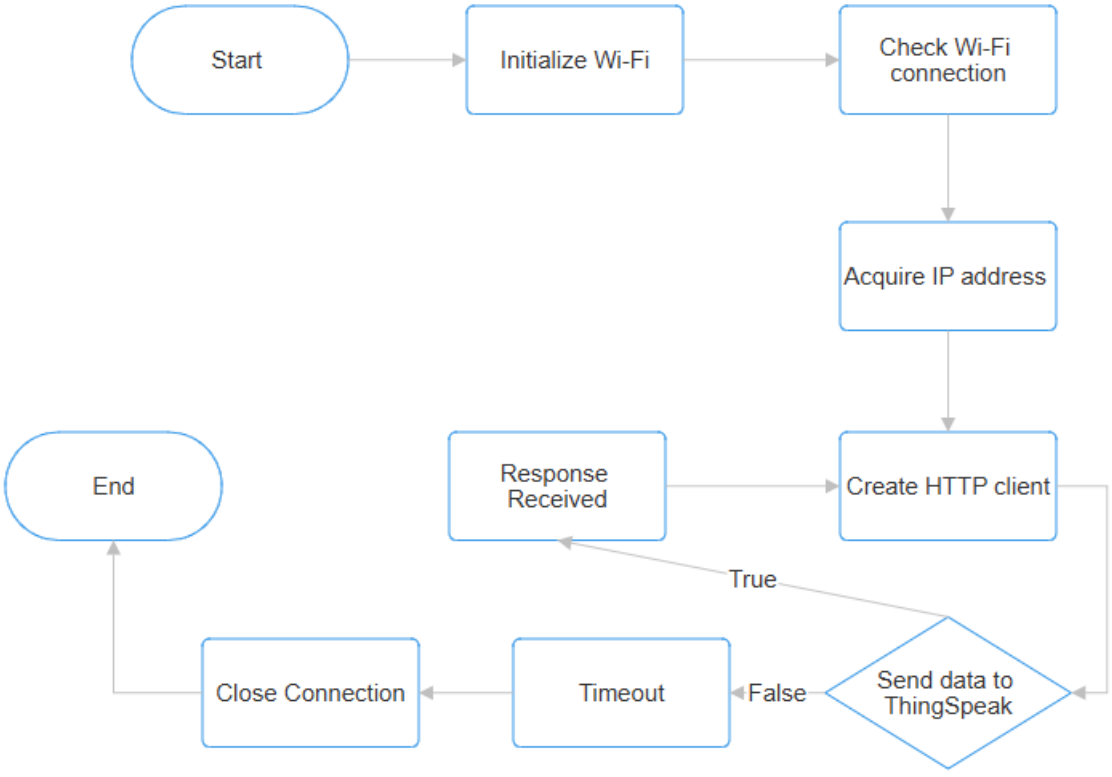

Figure 15 Wi-Fi Workflow

If your ThingSpeak channel is private, then clients would need your Read API Key to access the data through the ThingSpeak API. You will want to share this key in one of several ways that keeps it secure. This key allows clients to make GET requests in order to programmatically pull data, or to view via custom dashboards tailored to their needs. To make this easier, it's worth letting your clients know that their application can leverage the ThingSpeak RESTful API to access your data. This way, clients could effectively include your data in their applications. With that in mind, each client will need to have your Channel ID and Read API Key in order to do so.

If your ThingSpeak channel is set to be public, then the Public View URL is found on the main page of your channel. You should copy this URL and communicate it with your clients or viewers. Through this link, it will be enabled to view in real-time the latest visualized data. Also, all of the historical charts can be accessed without logging in. Also, if you have your own website, then you are able to embed your ThingSpeak charts directly. Each chart provided by ThingSpeak has the option for Embed Code. It is mostly provided on the Visualization tab of your channel. You will just need to copy the HTML that is being provided and paste it into your website so that it shows real-time data to viewers.

Regarding the encryption and data security problems, the Wi-Fi Encryption (WPA) is already handled when you connect to the network using the WiFi.begin(ssid, password) method and the SSL/TLS (Transport Layer Security) encryption while sending data to a server, usually over an HTTPS connection. Secure Sockets Layer (SSL) is the abbreviation for Secure Sockets Layer and refers to a protocol used in securing communication over a computer network, especially over the internet. It is used to establish an encrypted link between a client, typically a web browser or IoT device, and a server, which could also be a website or a cloud service. SSL ensures that any data sent by the client and the server is encrypted; hence, it protects it from eavesdropping, tampering, and forgery. SSL encrypts data in transit between the client and the server in such a way that any interception by any other entity would make it illegible. That helps protect sensitive data, such as passwords, credit card details, or personal information.

```
#include <WiFiClientSecure.h>
WiFiClientSecure client;
```

Figure 17 Library included for SSL encryption for HTTPS connection.

```
    if (WiFi.status() == WL_CONNECTED) {
    client.setInsecure();  // Disable certificate validation (for testing purposes, use proper certs in production)

    if (client.connect(host, httpPort)) {
        String url = "/update?api_key=" + writeApiKey + "&field1=" + String(deviceCount);
        Serial.print("Requesting URL: ");
        Serial.println(url);

        client.print(String("GET ") + url + " HTTP/1.1\r\n" +
                     "Host: " + host + "\r\n" +
                     "Connection: close\r\n\r\n");

        // Read response from the server
        readResponse(&client);
    }
    client.stop();
}
```

Figure 16 SSL encrytion before sending data over Thingspeak

It responds with its public key-on an SSL certificate-when a client connects to the server. With this public key, the client then generates a secret key to be used in symmetric encryption. Only the server can decrypt this because it has the private key. Once the client and server have agreed on a shared secret key, all other communications are encrypted using symmetric encryption. This is very fast compared to public-key encryption. Establishment of an SSL connection between client and server is called the "handshake":

- In this step, the client and the server agree on what encryption algorithms will be used.
- The server responds with its certificate in SSL.
- The client then verifies this certificate, and if valid, it associates the public key with it in order to initiate a secure connection.

4 SYSTEM EVALUATION AND VALIDATION PROCEDURES

4.1 Experimental Setup and Environmental Conditions for System Evaluation

The system was tested under controlled conditions so that proper and reliable performance evaluations could be performed. The hardware configuration consists of two Heltec LoRa32 V2 devices: one acts as a sender connected with a power bank, while the other acts as a receiver connected with a computer. Devices have been strategically positioned to keep the distances during the tests. BLE scanning, LoRa transmission, and Wi-Fi connectivity were some of the configuration packages used to establish respective parameters that enhanced efficiency. Based on the functional testing of the system, many test scenarios were built up: indoor and outdoor conditions of variable BLE device densities. The experiments were conducted at different locations, including infrastructure such as buildings and other sources of interference like other wireless networks. Other ambient conditions that affect signal propagation, such as the time of day and weather conditions, were recorded. Also recorded were possible interference from surrounding wireless devices and physical obstacles. Data collected from the experiments included BLE scan results, successful LoRa transmissions, response times, and reliability of coverage in general. The tests were performed for defined durations so that something for which a good experiment is desired or expected would take place. This comprehensive approach in the creation of the experimental setup and conditions was aimed at the robust testing of the system's capability to allow transparency and reproducibility in the results.

4.2 Assessment of System Performance Metrics and Operational Efficacy

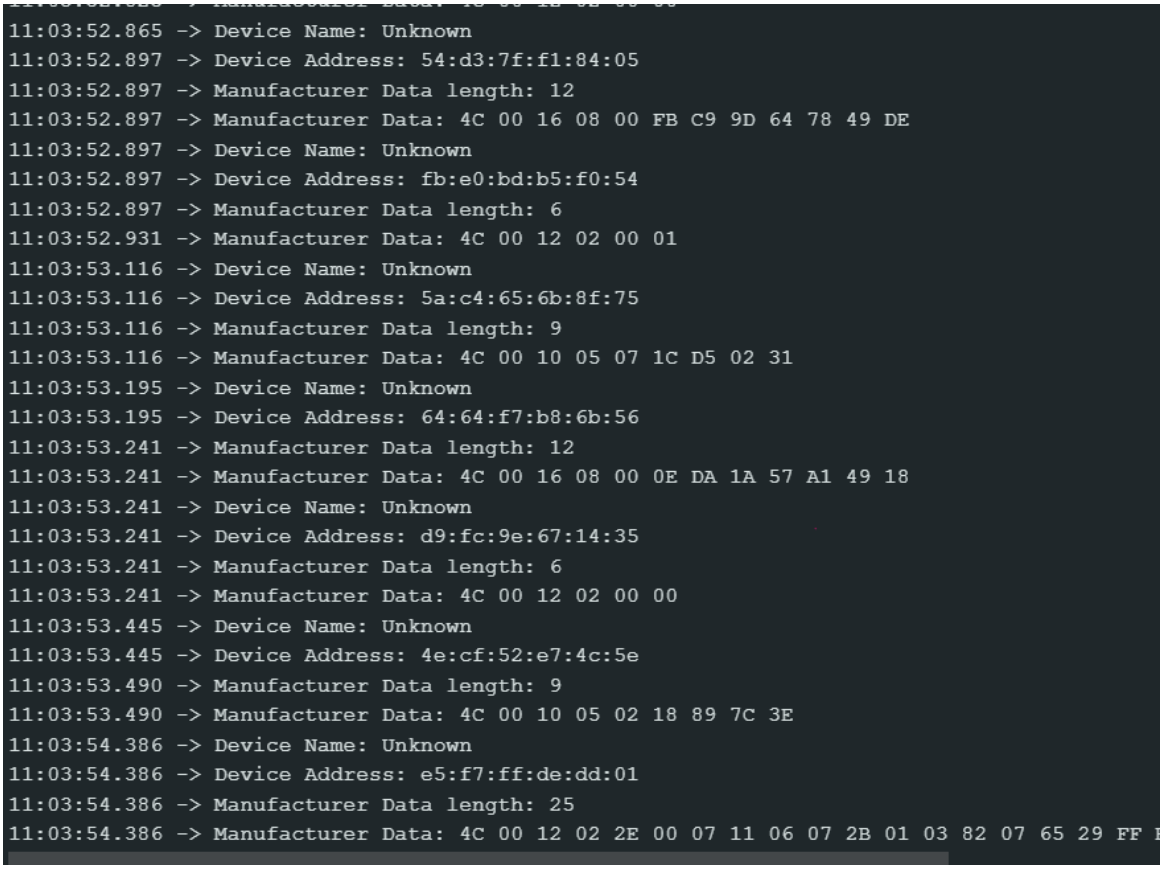

Figure 18 The results of Devices found from Sender device.

Above are code snippets representing BLE scanning, processing of detected devices, and data sending using LoRa in this project. In the given configuration, the ESP32 Heltec LoRa32 V2 module is programmed to initialize a BLE scan and discover the devices around it. It includes the name of the device, address, UUID, and beacon type. The discovered devices have been categorized on the basis of their service UUID and manufacturer data, with results prepared for transmission in a formatted string over LoRa. These will be transmitted after scanning is complete, where there is an aggregate packet of data transmitted over LoRa by ESP32 to the receiver device for processing and analysis.

This would expectedly be an output that includes the number of devices found, including details like device name and address, among other extra data that is further explained to include manufacturer data or beacon type. In actuality, the output, as shown in the image, contains the address and manufacturer data for different devices detected. Some devices

have known details like name and service UUID, while others can only disclose the manufacturer data or the beacon type. The actual output given is what is expected-the BLE scanning and data processing portion is functioning as it should. As an example, the hexadecimal manufacturer data here shown carries some information about the device characteristics which might be useful in applications around presence tracking.

This is the sender devices conducted in the outdoor location which is the space between the 3 Obudai Universities at Tavaszmezo 5, Budapest, Hungary. The sender was powered and started tracking in the centre of 3 buildings and the receiver is connected to the laptop inside the yard behind Building A (approximately 60 meters), so it is blocked through the university walls. The experiment tested and given results shown that:

- Outdoors, the range depends upon the existence or nonexistence of obstacles like trees, buildings, and vehicles. LoRa and BLE signals can have the maximum possible range only in totally open areas without obstacles, while with lots of obstacles, their range is decreased. As the space surrounded by buildings provided a small number of trees. Such range is good for tracking all the BLE devices in within 30 meters and sent the data successfully to the receiver.
- Temperature and rain or humidity, it follows. LoRa is generally resistant, but with high humidity, it could reduce strength over a big area.
- This is because BLE scanning/LoRa communication may not be interfered with by as many electronic sources in outdoor settings, and thus may yield a better, closer-to-accurate detection of presence.

Code provided herein is a LoRa-based receiver program that receives BLE device information sent by a LoRa sender and further processes the received packets. Herein, the receiver initializes a LoRa module and configures it in RX mode to receive incoming packets from the sender. The Wi-Fi setup section is commented out for now but can easily be used in the future to send this data across to ThingSpeak-an open IoT analytics platform. The receiver listens for LoRa packets, each one containing the details of the BLE devices scanned by the sender, including device name, address, service UUIDs, and manufacturer data. Whenever a packet is received, the function OnRxDone is invoked that retrieves some of the information, like the device name and its address within the received packet and prints them onto the serial monitor.

```
11:13:40.700 -> Received packet "Devices found: 71
11:13:40.732 -> Device name: IQOS ILUMA
11:13:40.732 -> Device address: c0:5a:03:b3:bf:01
11:13:40.732 -> Found ServiceUUID: daebb240-b041-11e4-9e45-0002a5d5c51b
11:13:40.732 -> Manufacturer Data length: 4
11:13:40.732 -> Manufacturer Data: 23 2 0 0
11:13:40.732 -> Device name: Unknown
11:13:40.732 -> Device address: 44:53:07:00:80:2f
11:13:40.732 -> Found Servi" with rssi -37, length 255
11:13:40.732 -> Received Data:
11:13:40.732 -> Devices found: 71
11:13:40.732 -> Device name: IQOS ILUMA
11:13:40.732 -> Device address: c0:5a:03:b3:bf:01
11:13:40.763 -> Found ServiceUUID: daebb240-b041-11e4-9e45-0002a5d5c51b
11:13:40.763 -> Manufacturer Data length: 4
11:13:40.763 -> Manufacturer Data: 23 2 0 0
11:13:40.763 -> Device name: Unknown
11:13:40.763 -> Device address: 44:53:07:00:80:2f
11:13:40.763 -> Found Servi
11:13:40.763 -> Extracted Device Name: IQOS ILUMA
11:13:40.763 -> Extracted Device Address: c0:5a:03:b3:bf:01
11:13:40.763 -> Extracted Device Name: Unknown
11:13:40.763 -> Extracted Device Address: 44:53:07:00:80:2f
11:13:40.763 -> Into RX mode
```

Figure 19 Result of Received packet from receiver device - At school.

The result would be a listing of BLE devices detected by the sender, in a format that shows the name and address of each device among other details like Service UUID and manufacturer data. Every entry of a BLE device should be in structured format with clear delimiters that demarcate different attributes of each device. The program should extract and display information about each device in sequence, hence making easy identification of the detected devices possible.

The actual output, provided by the figure below, partially matches with the expected output. The receiver successfully prints the BLE device details it has captured from the LoRa packet. We can notice that there is more than one device: one with the name "IQOS ILUMA", and an unknown device (without name). For each detected device, the output shows attributes, including the device name, device address, Service UUID, and manufacturer data length.

The output prints the extraction of the device name and its address, which it prints as "Extracted Device Name" and "Extracted Device Address," respectively. However, the output seems a bit redundant, probably due to the type of received packet and its way of

structuring the data to be shown. Also, for each device, it shows its Service UUID and manufacturer data properly but not well-aligned, hence it may not give a fine reading.

In other words, the code does a good job of fetching details regarding BLE devices, but it would look even better with a little modification. The differences between expected and real output are minor, and for the most part, they concern formatting rather than functionality.

In our BLE presence tracking system based on LoRa, information each BLE device advertises actively will be captured and displayed by the receiver but not always extracting information such as device name, UUID, or manufacturer data. This is because BLE devices use only one advertisement packet, not all of which broadcast full information in each packet. Some may broadcast only the device address or partial information, while others may broadcast additional information such as name or service UUID. So, for different BLE devices, the output at our receiver will look different, since each BLE device might use a different advertisement structure. This means selective data reception, where we will only show the information overtly given out by each device. Hence, the records of data will be varied, and sometimes incomplete; this however ensures the efficiency of our system in adapting to the nature of BLE advertising packets, which is highly diverse.

For testing the longer-range function, I put the sender at home for transmitting data, while the receiver was at the Tram 28 stop at Masza Utca, receiving data.

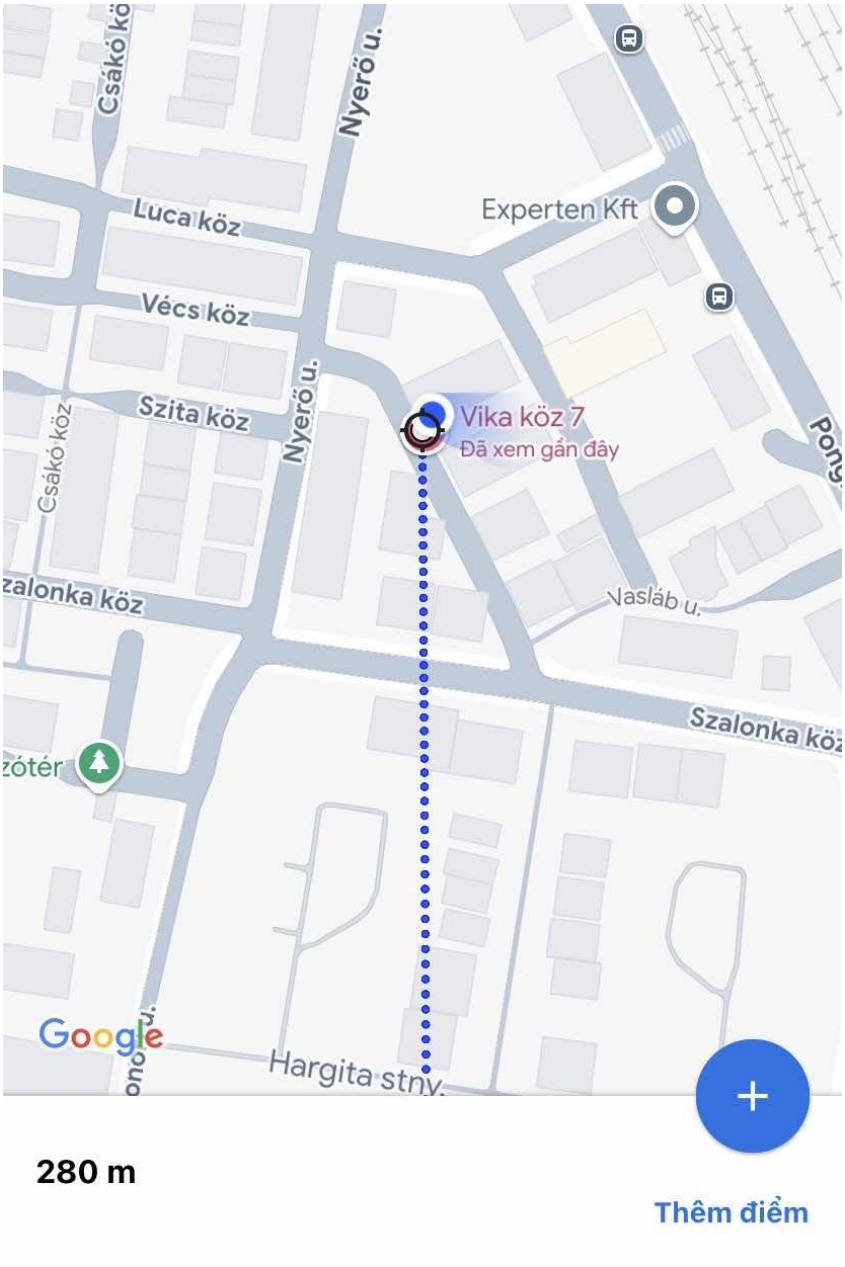

Figure 20 LoRa Distance Test (distance measured by Google Map)

During this test, the LoRa sender could transmit data over a distance of 280 meters to the receiver in an urban environment, proving that this device can enable effective long-range communication. This successful transmission proves the fact that LoRa can indeed support low-power, long-distance transmission of data in the most built-up areas. However, the strength of the signal and the quality of the transmission are susceptible to interference from a variety of factors, including physical barriers and disturbances owing to structures and other environmental factors that operate in their near vicinity. It follows that structures and dense materials along a transmission path can significantly lower the range of a signal

and its reliability. This further hints at why the technology will work best when there is a line-of-sight or at least minimal obstructions. This result shows the validity of LoRa for real-world applications, while on the other hand, it does point out some of the challenges created by environmental factors in regard to wireless communication.

After the indoor test case, it is seen that:

- Indoors, BLE and LoRa signals may be interfered with by walls, furniture, electronic devices, and metal structures. These could make BLE scanning and LoRa transmission have a lower range and reliability.
- The concrete values of radio waves' reflectivity and absorbency differ for various building materials: concrete, wood, glass, etc. For instance, the presence of metal structures can provide conditions for multipath interference when signals jump, creating delays that have a potential impact on inaccuracies in the detection of BLE devices.
- BLE and LoRa have relatively shorter ranges indoors due to obstacles, and LoRa's long-range capability might be curbed. Indoor testing would show the performance of the system when placed in enclosed areas with multiple obstructions.

With the receiver changing locations, there was also a slight reduction in speed while moving into obstructive areas. That would imply that with increased obstacles and distance, the data transfer rates are affected; hence, the position of the receiver devices is critical in order to have optimum results. These observations give insights into practical limitations of the system and its adaptiveness to different environmental circumstances.

. This also brings out the difference in signal behaviour between an indoor and outdoor test environment in view of the system's applicability to different practical engagements. In cases of high-interference environments, like crowded urban areas, the range of the system may be limited, and hence extra nodes or closer receiver placement will be required. On the other hand, open outdoor areas-like agricultural fields-offer longer-range coverage for large-scale tracking applications. All these results would help identify the suitability of the system under various real-world environments and provide suggestions on how to deploy them further. For instance, due to obstruction and interference, LoRa range is reduced in a densely populated urban environment like a city centre characterized by high-rise buildings, heavy traffic flow, and Wi-Fi congestion. In applications like smart parking,

monitoring of garbage bins, or air quality, which involve area-wide coverage, more nodes or receivers at strategic places may be required. Several gateways can also be installed on rooftops and along streets to continue the coverage despite interference. Such an intelligent parking system would further require additional nodes near parking areas in order to ensure that data collection is always reliable throughout the city.

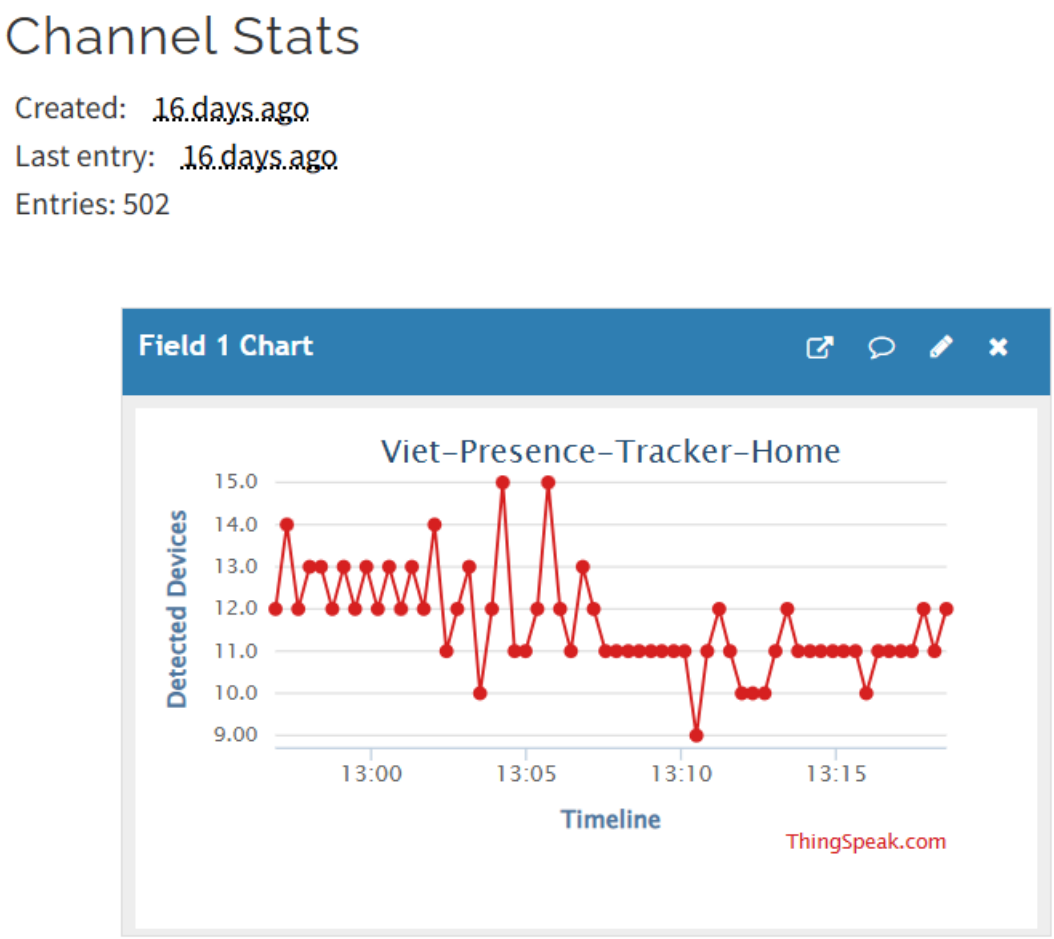

Figure 21 BLE presence tracked at home uploaded to ThingSpeak.

This will ensure that your data is well uploaded to ThingSpeak, hence us having a very important visualization of the number of BLE devices in a particular area. In this setup, the number of detected BLE devices, represented on the Y-axis, is tracked and plotted against the timeline, represented on the X-axis, showing variations in the presence of devices over time. Each point on the graph represents the number of BLE devices at an interval, hence allowing the maintenance of real-time data on activity and movement patterns in the area under observation. With the cloud service from ThingSpeak, data is not only securely stored, but it also enables remote access for real-time tracking and analysing of presence

data without storage or processing locally. This setup demonstrates the effectiveness of the "Viet-Presence-Tracker-Home" in collecting, uploading, and visualizing BLE presence data to easily monitor and analyse.

Regarding the problem is that managing data overflow when using LoRa for wireless communication is a common problem but never seen in my practical test. The solutions you can do with big datasets is fragment them into smaller packets, try sending the most important data first if you feel the user will receive data continuously , greatly cuts down the size of the data packets, implemented before transmission using data compression , use the adjusted rate of retransmission based on network conditions or device states , send information in periodic time cycles, not constant, would allow maintaining peak workloads.

When compressing the data, it is available to compress the data (e.g., using zlib, LZ4 libraries) before sending and make sure to decompress on the receiving side. This method provides the advantages of saving bandwidth since it reduces the amount of data being sent and sometimes, this may also lighten the handling of larger datasets but also, the disadvantages are adding extra processing overhead on both sending and receiving side and can introduce latency due to compression/decompression. If you have a stream of continuous data, focus on sending the most critical data first. The system shall assign a level of priority to different types of data such as real time tracking vs. historical information. This method gives high-priority data shall be transmitted, while lower priority data is buffered and transmitted later and assumes the most valuable information gets delivered first and will come in handy when available bandwidth is at a premium, but in contrast, it also must have a system that provides this functionality as well as may delay transmitting the lower level of priority data.

Table 4 Result Table

	Indoor	**Outdoor**
Location	Sender: inside the Vika köz 7 house. Receiver: Tram 28/28A/62 stop Mazsa utca	Sender: middle surrounded by 3 buildings of Obuda University (Tavaszmező street 15, Budapest, Hungary) Receiver: Backyard of Building A of Obuda University at the same location

Distance	280m	60m
Obstacles	Households walls, long distance between sender and receiver	University walls, electronic gadgets, Trees, Sunny weather
Result	Successful	Successful
Note	The data collected and sent successfully. The receiver device was brought to different area: in a tree-dense area: Noticed the longer receiving speed of the data; nearer to the sender device: Noticed a faster receiving data	The data collected and sent successfully.

4.3 Improvement Opportunities and Research Directions

Improving efficiency in BLE scanning can significantly enhance the performance and energy economy of the presence-tracking system. Dynamically adjusting the BLE scan intervals with respect to environmental conditions may optimize not only the accuracy and coverage but also the battery life of a system with respect to device density or movement levels within the area. This approach could involve adaptive filtering that prioritizes certain device types or data patterns, effectively reducing redundant scanning and improving the quality of data gathered. Research in adaptive BLE scanning techniques, potentially incorporating machine learning, may offer valuable insights into predicting the ideal scan intervals. Adaptive models, sensitive to real-time device-density and movement-pattern data, can strike a balance between efficient scanning and minimized power consumption by the system.

Error correction techniques, retry mechanisms, and signal-strength monitoring may be used to improve data accuracy and reliability of LoRa transmissions especially at longer ranges or in difficult environments. Adding mechanisms like automatic retransmissions for failed packets or continuous monitoring of signal quality further stabilizes the data flow. Adaptive transmission protocols for LoRa especially through dynamic adjustment of parameters such as the spreading factor or bandwidth, based on environmental conditions are of great potential. This could enhance the range and resilience in a variety of environments. Furthermore, a hybrid communication model that changes data transmission

based on network conditions and the size of data by using BLE for close-range and smaller data packets, and LoRa for longer-range transmissions would help optimize network resources to generally improve the efficiency of the system.

In my humble opinion, the convergence of AI and deep learning algorithms with BLE and LoRa-based presence tracking systems points toward promising directions for advanced analysis and real-time anomaly detection. The application of machine learning models, especially deep neural networks, could enable the system to learn autonomously and detect complex patterns in large datasets that contain nuanced movement behaviours and irregularities in device activities. For instance, such algorithms could provide automated insight into device clustering patterns, highlight particular usage trends, and even predict spatial or temporal anomalies without manual rule-setting—all in an effort to bring up security and asset-tracking applications.

One of the important research directions is applying supervised learning in the classification and detection of specific types of devices or movement signatures in various environments. Similarly, unsupervised learning models will potentially enable the system to adapt itself—recognize patterns or deviations that were unknown in advance, particularly in dynamic environments. This would lead toward adaptative presence-tracking frameworks able to modify BLE scan rates or LoRa transmission parameters according to real-time context. Furthermore, the neural networks can be trained to do any number of tasks, such as predicting optimal scan intervals, identifying clusters of devices, or even predicting the future path that a device may take. By building on these areas, scientists can put the foundation for highly flexible, intelligent systems in various field applications, such as surveillance, monitoring of personnel, and resource management.

5 SUMMARY

In this paper, the design and implementation of a multi-technology presence management system are presented. This system integrates Bluetooth Low Energy, LoRa, and Wi-Fi in the detection of BLE-enabled devices and in tracking those inside the area of interest. With the low-power and efficient scanning features that BLE offers, this system would be able to monitor surrounding devices in real time—something quite vital for applications in smart environments, asset tracking, and security monitoring. The data collected by the BLE-enabled sender device are transmitted via LoRa to a receiver for local data collection. Wi-Fi allows for the upload of data in real time to ThingSpeak, a cloud-based solution. ThingSpeak provides capabilities for visualization, allowing users to inspect and analyse the number of devices detected with an easily accessible dashboard, thus being useful in gaining the best insights about the presence and movement patterns of devices within the area under surveillance.

Integration of LoRa for long-range communication, Wi-Fi for cloud-based data access, and BLE for local device detection ensures flexibility, scalability, and efficient data management within the system. The design hence offers a flexible solution applicable in several fields of industry and the public sector, such as manufacturing, healthcare, and public safety. The successful testing and visualization on ThingSpeak prove that the system is effective in capturing and displaying the data of the devices accurately for easy interpretation by the users. This project shows the possibility of combining multiple IoT communication technologies into one platform, thus creating a robust and practical presence tracking system suitable for smart environments and the Internet of Things (IoT) ecosystem.

6 REFERENCES

[1] “Bluetooth specification version 4.0,” Bluetooth SIG, [Online]. Available: https://www.bluetooth.com/specifications/specs/core-specification-4-0/. [Accessed 27 10 2024].

[2] J. Haartsen, M. Naghshineh, J. Inouye, O. J. Joeressen, and W. Allen, “Bluetooth: vision, goals, and architecture,” 1 Oct 1998.

[3] M. A. Al-Shareeda, S. Manickam and S. Karuppayah, “Bluetooth low energy for internet of things: review, challenges, and open issues Bluetooth low energy Bluetooth low energy for internet of things Internet of things,” *Indonesian Journal of Electrical Engineering and Computer Science,* vol. 31, no. 2, pp. 1182-1189, August 2023.

[4] A. Lacava, V. Zottola, A. Bonaldo, F. Cuomo and S. Basagni, “Securing Bluetooth Low Energy networking: An overview of security procedures and threats,” *Computer Networks,* vol. 211, p. 108953, 2022.

[5] N. Phuong, “Understanding Bluetooth Low Energy: The Technology Behind IoT Devices,” 9 8 2024. [Online]. Available: https://kiemtientuweb.com/bluetooth-low-energy-1724088823.html. [Accessed 3 11 2024].

[6] J. Tosi, F. Taffoni, M. Santacatterina, R. Sannino and D. Formica, “Performance Evaluation of Bluetooth Low Energy: A Systematic Review,” *Sensors,* vol. 17, no. 12, p. 2898, 2017.

[7] S. M. Marroudi and C. Gomez, "Bluetooth Low Energy Mesh Networks: A Survey," *Sensors,* vol. 17, no. 7, p. 1467, 22 June 2017.

[8] M. R. Ghori, T.-C. Wan and G. C. Sodhy, “Bluetooth Low Energy Mesh Networks: Survey of Communication and Security Protocols,” *Sensors,* vol. 20, no. 12, p. 3590, 25 June 2020.

[9] “Wearable Gesture Control - #6 BLE communication with an app,” element14, [Online]. Available: https://community.element14.com/challenges-

projects/design-challenges/low-power-iot/b/blog/posts/wearable-gesture-control---6-ble-communication-with-an-app. [Accessed 1 11 2024].

[10 M. Ryan, “Bluetooth: With Low Energy Comes Low Security,” iSEC Partners, [Online]. Available: https://www.usenix.org/conference/woot13/workshop-program/presentation/ryan. [Accessed 22 10 2024].

[11 A. Barua, M. A. Al Alamin, M. S. Hossain and E. Hossain, “Security and Privacy Threats for Bluetooth Low Energy in IoT and Wearable Devices: A Comprehensive Survey,” *IEEE Open Journal of the Communications Society,* vol. 3, pp. 251-281, 2022.

[12 “What Is LoRa®?,” Semtech Corporation, [Online]. Available: https://www.semtech.com/lora/what-is-lora. [Accessed 20 9 2024].

[13 “LoRa™ Modulation Basics,” Semtech Corporation, May May 2015. [Online]. Available: https://web.archive.org/web/20190718200516/https:/www.semtech.com/uploads/documents/an1200.22.pdf. [Accessed 11 3 2024].

[14 "What is LoRaWAN," The Things Industries 2024, [Online]. Available: https://www.thethingsnetwork.org/docs/lorawan/what-is-lorawan/. [Accessed 1 11 2024].

[15 T. Zanke, “LoRa : A Breakdown of What It Is & How It Works,” 24 Nov 2022. [Online]. Available: https://medium.com/@tusharzanke16/lora-a-breakdown-of-what-it-is-how-it-works-36d6d4e9cd59. [Accessed 2 11 2024].

[16 Wikipedia, “LoRa,” [Online]. Available: https://en.wikipedia.org/wiki/LoRa. [Accessed 29 10 2024].

[17 “LoRa- (Long Range) Network and Protocol Architecture with Its Frame Structure,” Codeplayon, 30 January 2019. [Online]. Available: https://www.codeplayon.com/lora-long-range-network-and-protocol-architecture-with-its-frame-structure/. [Accessed 1 11 2024].

[18 “LoRa- (Long Range) Network and Protocol Architecture & Frame Structure,” Techplayon, 10 October 2018. [Online]. Available:

https://www.techplayon.com/lora-long-range-network-architecture-protocol-architecture-and-frame-formats/. [Accessed 1 11 2024].

[19 "Long Range Wide Area Network (LoRaWAN) Protocol," PiEmbSystech, [Online]. Available: https://piembsystech.com/long-range-wide-area-network-lorawan-protocol/. [Accessed 6 11 2024].

[20 "LoRaWAN and IoT: Revolutionizing Connectivity for a Smarter World," Emertxe, [Online]. Available: https://www.emertxe.com/news/emertxe-events/2024/10/10/smart-connectivity-with-lorawan-and-iot. [Accessed 2 11 2024].

[21 "Wi-Fi: Overview of the 802.11 Physical Layer and Transmitter Measurements," Tektronic, [Online]. Available: https://www.tek.com/en/documents/primer/wi-fi-overview-80211-physical-layer-and-transmitter-measurements. [Accessed 30 10 2024].

[22 "History | Wi-Fi Alliance," Wi-Fi Alliance, [Online]. Available: https://www.wi-fi.org/who-we-are/history. [Accessed 1 11 2024].

[23 "Global Wi-Fi Enabled Devices Shipment Forecast, 2020-2024," Market Intelligence & Consulting Institute (MIC), July 2020. [Online]. Available: https://www.researchandmarkets.com/reports/5135535/global-wi-fi-enabled-devices-shipment-forecast. [Accessed 29 10 2024].

[24 "HackerBox 0073: LAN Lord," AUTODESK Instructables, [Online]. Available: https://www.instructables.com/HackerBox-0073-LAN-Lord/. [Accessed 30 10 2024].

[25 D. Morcrii, Y. Chen and P. Musilek, "IoT-based smart homes: A review of system architecture, software, communications, privacy and security," *Internet of Things*, vol. 1, no. 2, pp. 81-98, 2018.

[26 "Wi-Fi," Wikipedia, the free encyclopedia, [Online]. Available: https://en.wikipedia.org/wiki/Wi-Fi. [Accessed 30 10 2024].

[27 “ThingSpeak Support from Desktop MATLAB,” MathWorks, [Online]. Available: https://ww2.mathworks.cn/en/hardware-support/thingspeak.html. [Accessed 4 11 2024].

[28 “ThingSpeak for IoT Projects,” MathWorks, [Online]. Available: https://ww2.mathworks.cn/en/products/thingspeak.html. [Accessed 4 11 2024].

[29 “What is Wirepas Mesh?,” haltian, [Online]. Available: https://haltian.com/resources/what-is-wirepas-mesh/. [Accessed 31 10 2024].

[30 J. Vehkalahti, “Wirepas Mesh vs. BLE Mesh,” 12 March 2021 . [Online]. Available: https://www.linkedin.com/pulse/wirepas-mesh-vs-ble-jani-vehkalahti/ .

[31 “Compare Products,” AVIA Marketplace, [Online]. Available: https://marketplace.aviahealth.com/compare/72006/71982. [Accessed 28 10 2024].

[32 “Asset Tracking for Healthcare,” BeaconTrax, [Online]. Available: https://www.beacontrax.com/asset-tracking-for-healthcare/. [Accessed 30 10 2024].

[33 “Real-Time Asset Tracking for Manufacuturing,” BeaconTrax, [Online]. Available: https://www.beacontrax.com/real-time-inventory-and-asset-tracking-for-manufacturing/. [Accessed 3 11 2024].

[34 S. Zimmermann, “Mystery Apple AirTag on car was used to track us, say former Country Club Hills couple, calling it ‘very, very creepy’,” 14 Oct 2023. [Online]. Available: https://chicago.suntimes.com/2023/10/14/23915233/apple-airtag-stalking-class-action-lawsuit-desiree-freeman-frank-country-club-hills. [Accessed 1 11 2024].

[35 “WiFi LoRa 32,” Heltec Automation, [Online]. Available: https://docs.heltec.org/en/node/esp32/wifi_lora_32/index.html. [Accessed 1 11 2024].

[36 “Products & Solutions,” BeaconTrax, [Online]. Available: https://www.beacontrax.com/products/. [Accessed 5 11 2024].