



# **ASHESI UNIVERSITY COLLEGE**

## **DESIGN AND IMPLEMENTATION OF AN ANTI-THEFT VEHICLE TRACKING SYSTEM USING LORA TECHNOLOGY**

**THESIS**

B.Sc. Computer Engineering

**Betty Blankson**

**2022**

**ASHESI UNIVERSITY COLLEGE**

**DESIGN AND IMPLEMENTATION OF AN ANTI-THEFT VEHICLE  
TRACKING SYSTEM USING LORA TECHNOLOGY**

**THESIS**

**CAPSTONE PROJECT**

Capstone Project submitted to the Department of Engineering, Ashesi  
University College in partial fulfilment of the requirements for the award of  
Bachelor of Science degree in Computer Engineering.

**Betty Blankson**

**2022**

## DECLARATION

I hereby declare that this capstone is the result of my own original work and that no part of it has been presented for another degree in this university or elsewhere.

Candidate's Signature:

.....

Candidate's Name:

...BETTY BLANKSON.....

Date:

.....27<sup>TH</sup> APRIL 2022.....

I hereby declare that preparation and presentation of this capstone were supervised in accordance with the guidelines on supervision of capstone laid down by Ashesi University College.

Supervisor's Signature:

.....

Supervisor's Name:

.....

Date:

.....

## **Acknowledgments**

I thank God Almighty for coming this far with my tertiary education. I also thank Ashesi for the platform to explore the different aspects of life.

I also acknowledge and appreciate my supervisor, Mr. Kofi Adu Labi, for the support and guidance during this project. I am also grateful for every lecture that taught me at Ashesi for all the encouragement and support.

## **Abstract**

In recent times, vehicle tracking systems have been introduced to reduce the number of stolen items such as phones, cars, or other devices. As years go by, these tracking devices are improved with additions such as vehicle monitoring components to ensure the safety of drivers. Such components include temperature sensors, fuel sensors, motion sensors, and many others. This paper encompasses designing and implementing an anti-theft vehicle tracking system using Arduino nano 33 BLE sense, GPS, and LoRa technology. The main components of the device include an Arduino nano 33 BLE sense, a GPS module, a LoRa transceiver, and a gateway. The Arduino nano BLE sense is embedded with a proximity sensor; the temperature sensor detects temperatures in the vehicle for the owner to be alerted when the temperature in the car rises above normal. The reading from the proximity sensor also enables the device to alert the owner when motion is detected around the car. The Arduino is interfaced with the other main components and placed inside the vehicle to be tracked. When the car is accessed, the user gets a notification and can turn off the car's fuel tank which is represented by a DC motor in this project.

## Table of Content

<b>DECLARATION .....</b>	<b>1</b>
<b>Acknowledgments .....</b>	<b>2</b>
<b>Abstract.....</b>	<b>3</b>
<b>Chapter 1: Introduction .....</b>	<b>8</b>
1.1 Background .....	8
1.3 Project Motivation .....	9
1.2 Problem Definition .....	9
1.4 Project Organization.....	10
<b>Chapter 2: Literature Review .....</b>	<b>11</b>
2.1 Vehicle Tracking Systems Related Works .....	11
2.1.1 GPS and GSM/GPRS controlled by an Embedded Linux board Vehicle Tracking Systems .....	11
2.1.2 Real-time anti-theft vehicle tracking system using GSM and GPS technology. ....	12
2.1.3 A web-based GPS-GPRS vehicle tracking system .....	13
2.2 GPS and LoRa concepts .....	14
2.2.1 GPS Description and Working Principle .....	14
2.2.2 LoRa Description and Working Principle .....	16
<b>Chapter 3: System Design .....</b>	<b>20</b>
3.1 System Objective and Overview .....	20
3.1.1 Design Specifications .....	20
3.1.2 Design Requirements .....	21
3.1.3 Non-functional Requirements .....	21
3.2 Design Decisions .....	21

3.2.1 Processor boards comparison .....	22
3.2.2 Pugh Matrix for Communication Technologies .....	23
3.2.3 Pugh Matrix for GPS Module .....	24
3.3 Proposed design Overview .....	24
3.3.1 Architecture of the overall proposed system .....	25
<b>Chapter 4: Methodology .....</b>	<b>26</b>
4.1 Experimental Setup .....	26
4.2 Hardware Components for Sensor Node .....	26
<b>APDS9960 .....</b>	<b>27</b>
4.3 Overall Circuit of the system .....	27
4.4 Nano BLE sense and Sensor node setup .....	28
4.5 Gateway Setup .....	30
4.5.1 Raspberry pi 3 setup .....	30
4.5.2 Connecting the Gateway to the Things Network (TTN) and Implementation .....	32
4.6 Web Application using Ubidots .....	33
<b>Chapter 5: Results and Discussion .....</b>	<b>34</b>
5.1 Single testing of sensors. ....	34
5.2 Testing Thresholds of sensors .....	35
<b>Chapter 6: Conclusion .....</b>	<b>37</b>
6.1 Summary .....	37
6.2 Limitations .....	37
6.3 Lessons Learnt .....	38
6.4 Future Works .....	38
<b>References .....</b>	<b>39</b>

## List of Figures

Figure 1: Block diagram of the raspberry pi vehicle monitoring system .....	11
Figure 2: Block diagram of related work 2 .....	13
Figure 3: Overview of design implemented in related work 3. ....	14
Figure 4: The three segments of the Global Positioning System .....	15
Figure 5: The trilateration algorithm. ....	16
Figure 6: LoRa technology stack. ....	18
Figure 7: Typical LoRaWAN implementation.....	19
Figure 8: Architecture diagram of the proposed system .....	25
Figure 9: Circuit connections for system .....	27
Figure 10: Raspberry pi connected to PuTTY via SSH .....	31
Figure 11: Configuring LoRa on raspberry pi .....	31
<i>Figure 12: Connecting LoRa to Nano BLE sense on Arduino .....</i>	<i>32</i>
Figure 13: Adding Raspberry pi gateway to TTN stack .....	32
Figure 14: Ubidots connected to TTN .....	33
Figure 15: Sensor testing .....	34
Figure 16: GPS and sensors testing .....	35
Figure 17: Testing thresholds .....	36
<b>List of Tables</b> .....	<b>6</b>
Table 1: Key Features of Lora technology.....	16



Table 2: Processor board Comparisons.....	22
Table 3: Pugh Matrix of Communication Technologies.....	23
Table 4: Pugh Matrix of possible GPS Modules.....	24
Table 6: Hardware components and specs.....	26
Table 7: Threshold and User Actions .....	29
Table 8: Thresholds and Notifications expected.....	29

# Chapter 1: Introduction

## 1.1 Background

The introduction of Internet of Things (IoT) in recent years to establish networks between devices is one of the greatest evolutions in technology. IoT has brought more enhancement in automation, data collection, and analysis. Individuals can now monitor different activities for different purposes efficiently. A vehicle tracking system is one example of the tremendous applications of IoT. A vehicle tracking system is a network of devices that uses the Global Positioning System (GPS) to track and sometimes control vehicles without direct contact with the car. Notwithstanding the introduction of IoT to monitor activities, vehicle theft rates are still very high worldwide. According to the Insurance Information Institute, in the US alone, the bureau released a preliminary analysis that indicates that the number of auto thefts in 2020 totaled 873,080, a 9.2 percent increase compared with 2019 when its data showed that 799,644 thefts were recorded [2].

Vehicle tracking has been around for more than four decades since the development of the first GPS module. However, most of the research work and implementation have been done with devices that need intense maintenance, hence making the tracking systems expensive. Communication methods mostly require data subscriptions to send or receive messages on the systems. Overall, most of the end products are costly and require a lot of power to run. Most of the tracking devices also do not have the means to prevent theft. Drivers hence must purchase separate theft prevention systems to prevent their cars from being stolen.

Also, GSM modules are the most used communication technology in the present tracking system research. LoRa is an emerging technology, and little research has been done on using it compared to the GSM module as a communication means for vehicle tracking systems. Presently there are 156 LoRa network operators in 171 countries. Apart from being

expensive, most communication technologies require high power to run as opposed to LoRa, which requires very low power to carry out tasks.

### **1.3 Project Motivation**

IoT is now a popular field of work due to how it has made operating machines easier and more accessible. Even though there is so much technological advancement through IoT, Africa seems to be lagging with these new implementations that are making lives much more manageable. Conducting research works and building designs that can be improved on is hence a great effort to push Africa forward when it comes to technological advancement.

The purpose of this project is to design and implement an anti-theft vehicle tracking system with safety features as a contribution to research and a step towards creating a device that will be useful to the continent.

### **1.2 Problem Definition**

On many occasions tracking systems are only used to find the location of the vehicles after they have been stolen or sent to a different location. For some tracking devices, the owner only gets the location of the vehicle upon request after realizing it is missing. For others, the vehicle can be controlled by the owner when it is found to make it inaccessible to the thief. However, these theft cases can be reduced by a good margin if the owner gets alerted when unauthorized persons try to access the vehicle even before taking it away. Only a few cars have a hazard detection system and hence pose a threat to riders and drivers. The combination of sensors, GPS, and communication technology can make it possible for owners to be alerted when cars are in the wrong hand and when unexpected temperatures or gases are detected in a vehicle.

## **1.4 Project Organization**

This document for the project contains six chapters. The first chapter introduces the idea of vehicle tracking and a fair idea of vehicle theft in some parts of the world. This chapter also gives a general overview of why and how the design is implemented. The first chapter also points out the motivation behind the project and what differentiates the system from already implemented devices. The second chapter elaborates on other works done to contribute to vehicle tracking using IoT as a field of study. The chapter also discusses how unique those already implemented projects are to the context within which they were designed. The following chapter details the design of the tracking system. Block diagrams, schematic diagrams, and other circuit diagrams are also explained in this chapter. The different ways that the design components are interfaced are also discussed. The methodology used in the implementation of the system is described in the fourth chapter. This includes the different processes and technologies and how the components communicate with each other to achieve desired results. In the fifth chapter, the outcome or results from the system is documented. Finally, the sixth chapter discusses problems faced and lessons learned from the project's onset. Future additions or subtractions from the design are also included in the final part of the last chapter.

## Chapter 2: Literature Review

### 2.1 Vehicle Tracking Systems Related Works

The increase in technological advancement over the years has had a good impact on society. However, as these technologies emerge, the urge to own such devices and machines also increases. This, in turn, creates vices as people who cannot afford them are tempted to use illegal means to obtain them. The theft of vehicles is a very common example of some of these vices. The emergence of IoT is hence crucial at this stage since IoT helps individuals to monitor various activities even when not physically present. Below are three examples of contributions people have made to vehicle tracking using IoT.

#### 2.1.1 GPS and GSM/GPRS controlled by an Embedded Linux board Vehicle Tracking Systems

Shinde and Mane [5] developed a vehicle monitoring and tracking system with the aid of GPS and GSM/GPRS controlled by an Embedded Linux board called Raspberry pi B+ via USB. In this system, there is automatic monitoring of school vehicles with extra safety features which include temperature sensing for an unusual rise in temperature in the vehicle and gas sensing during gas leakage. Tracking information such as location, vehicle ID, speed, date, and time are obtained and stored in the database of the raspberry pi.

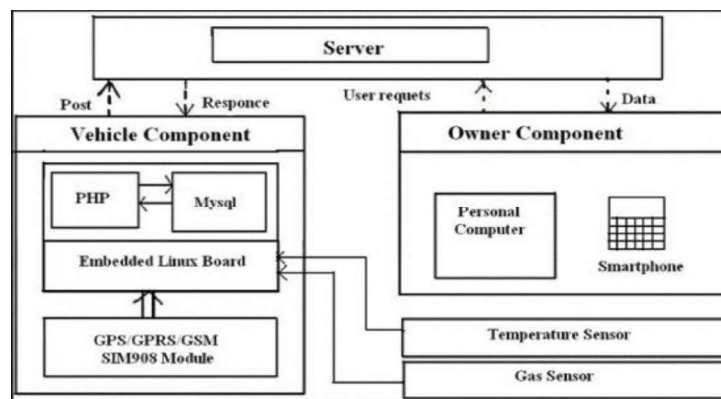


Figure 1: Block diagram of the raspberry pi vehicle monitoring system

The block diagram in figure1 [5] shows the various components of the system and how they interact with each other. With regards to tracking the vehicle, different longitudes and latitudes for a particular path obtained by GPS and GSM module are also stored in the database of raspberry pi. Hence, the longitude and latitudes of its current path are compared to what is stored. If the comparisons do not match, a wrong path detection message is sent to the user's mobile phone.

In essence, this system is very useful for school buses since information such as date and time would reduce the time students have to wait by roadsides for rides to school. Safety measures are also important when it comes to conveying students.

### **2.1.2 Real-time anti-theft vehicle tracking system using GSM and GPS technology.**

Maurya et al. [3] also proposed a real-time anti-theft vehicle tracking system using GSM and GPS technology. In their design, hardware peripherals are interfaced with the AT89C51 microcontroller. The GPS receiver and GSM modem are also connected serially to this microcontroller. The GPS modem continuously sends the latitude and longitude coordinates representing the location of the vehicle. This is picked up by the GSM modem and transmitted to the owner on request in the form of an SMS.

From the block in figure2 [3] below, in the proposed design, HOLUX GR-67 is the GPS modem used with a baud rate of 9600. SIM 900 with a baud rate of 57600 is also the GSM modem used. The mobile for receiving SMS from the GSM modem is a Nokia 5800. 5V and 12V are the voltages required by the microcontroller and the GSM, respectively. A 12V/3.2A battery supplies power and is used with regulators to provide the different voltages required. The authors of the research consider integrating other devices such as sensors in the future to improve their system.

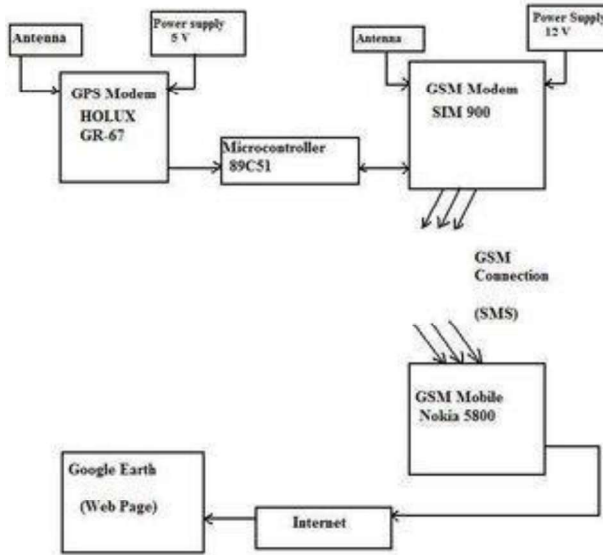


Figure 2: Block diagram of related work 2

### 2.1.3 A web-based GPS-GPRS vehicle tracking system

Another related work is a proposed design by Salim and Idrees [4]. A web-based GPS-GPRS vehicle tracking system was implemented in their design. Through a purposebuilt website, users can view the past and current location of vehicles with the tracking device. Unlike other tracking devices that receive the coordinates representing the location of vehicles via SMS using GMS devices, this tracker provides the details of location online to be accessed by users on a web page.

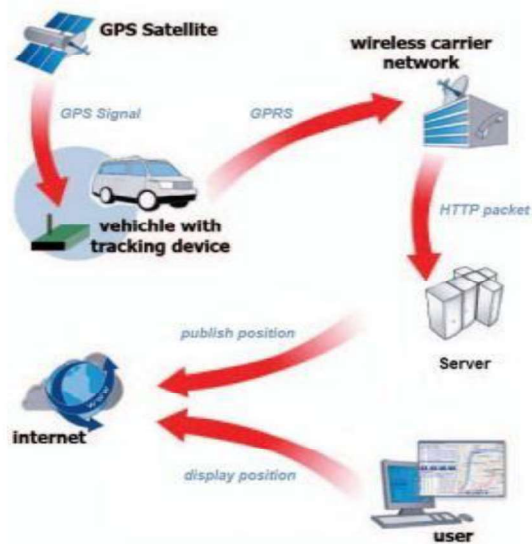


Figure 3: Overview of design implemented in related work 3.

In figure 3 [4], the main components of the system include Quad-band SIM908 GSM-GPS, a web application and database, and a desktop application. On the webpage, users are required to log inn to access data on the web page. Vehicle location coordination is provided by Google; hence JavaScript embeds Google Map API on the website using key and Google maps class. PHP functions are used to implement the administration of accounts.

## 2.2 GPS and LoRa concepts

### 2.2.1 GPS Description and Working Principle

The global positioning system (GPS) is a navigation system that uses a group of satellites to provide location and time information. The GPS currently operates with 31 satellites and is managed by the United States. There are presently four other location determination systems developed by India, China, Europe, and Russia. According to the Geotab team (2020) GPS consists of three segments, namely, ground control, space segments, and the user segment. **The space segment** comprises the various satellites that transmit signals about location and time user equipment. The **ground control segment**



includes earth-based monitor stations, master control stations, and ground antennae that control the satellites and maintain them in their orbits to ensure that measurements are within tolerance ranges. The final segment, i.e., **the user equipment segment**, consists of the different GPS receivers such as smartphones, watches, GPS modules, etc., that can receive signals from the satellites to display the location and time information.

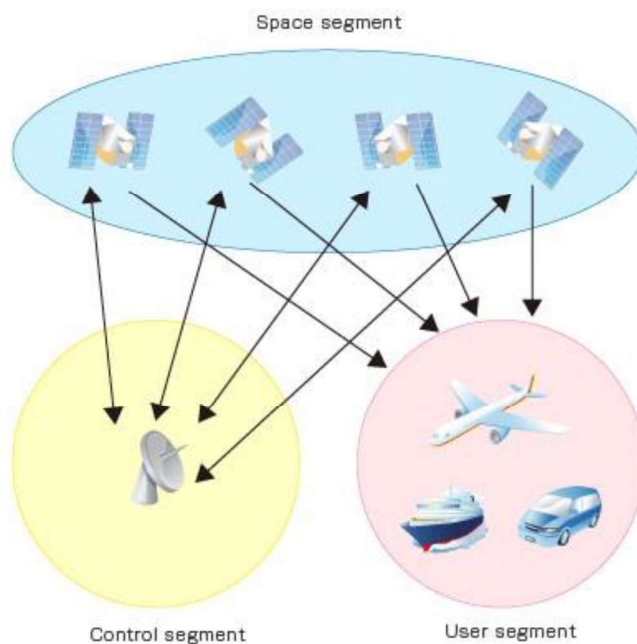


Figure 4: The three segments of the Global Positioning System

To determine location, velocity, and time a GPS uses a mathematical technique called trilateration. To properly determine locations, GPS utilizes at least three or four satellites. An initial satellite sends signals which determine the general location of the object in question. A second satellite sends signals to increase the precision of the location of the object. A third satellite is added to make the location more precise. The fourth satellite sends a signal to determine the location more precisely and more accurately.

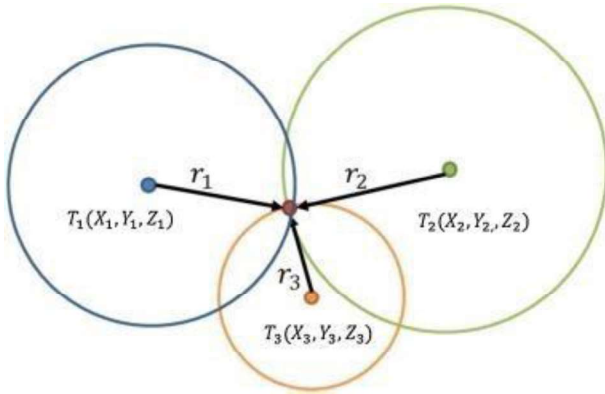


Figure 5: The trilateration algorithm.

### 2.2.2 LoRa Description and Working Principle

LoRa is a low-power wide-area technology that was developed by Cycleo in France. It is based on a modulation derived from chirp spread spectrum technology. This implies that an increase or decrease in frequency over a period is used to encode transmitted data. LoRa has low power consumption compared to other techniques and has a long transmission range. LoRa has a data rate of about 0.3 kilobits/second (kbits/s) and 5.5 kbits/s. LoRa is also known for the millions of messages it can carry during each deployment, depending on the number of gateways installed. Semtech (n.d.) stated that “An eight-channel gateway supports about a hundred thousand messages within a 24-hour period. If ten messages are sent a day, that gateway can support about 10,000 devices. Hence 10 such gateways can support roughly 100,000 devices and one million messages”. This implies that additional gateways can be added to achieve a larger capacity.

The key features are explained by Semtech (n.d.) as:

Table 1: Key Features of Lora technology

Feature	Description
---------	-------------

Long Range	Capable of connecting devices that are about 15km apart in rural areas.
Geolocation	Supports many GPS tracking applications worldwide.
Secure	Encrypted using end-to-end AES128 and mutual authentication is required.
Low Cost	Less expensive compared to other techniques due to reduced battery replacement expense, less infrastructure requirement, etc.
Standardized	LoRaWAN networks are available globally, enhancing interconnection between various devices. Hence widely used in IoT.
High Capacity	It can serve large markets and carries millions of messages in each base station.
Low Power	Utilized minimal power, therefore, prolongs battery life compared to other technologies.

According to Ray (2018) “LoRa is a lower-level physical layer technology that can be used in all sorts of applications outside of the wide area.” Operating at the physical level,

devices transform data into radio frequency signals for wireless transmission using the chirp spread spectrum communications of LoRa. In operation, radio waves are transmitted via air from a LoRa radio frequency transmitter to a radio frequency receiver in a gateway or contrariwise. More than one gateway can be in range with the transmitter; hence, the network service handles duplicate data.

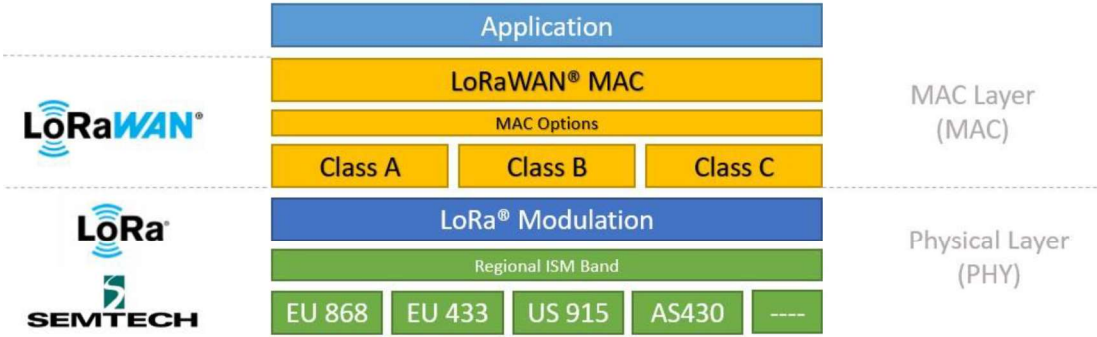


Figure 6: LoRa technology stack.

LoRaWAN is a point to a multipoint communication network that uses LoRa and has the star network topology arrangement. LoRaWAN has three classes which include the class A or ALOHA system, class B, and class C. With class A, end devices are mostly in sleep mode until a change depending on what is required to be monitored is detected. The devices send an uplink in response and wait for a downlink from the network. With class B, devices monitor to check for downlink from the network within a scheduled time given period as opposed to being in sleep mode until a downlink is received. They are hence used to monitor sensors and actuators. Class A LoRaWAN, on the other hand, is always on and ready for a downlink from the network. These are usually used in electrical meters, traffic lights, etc.

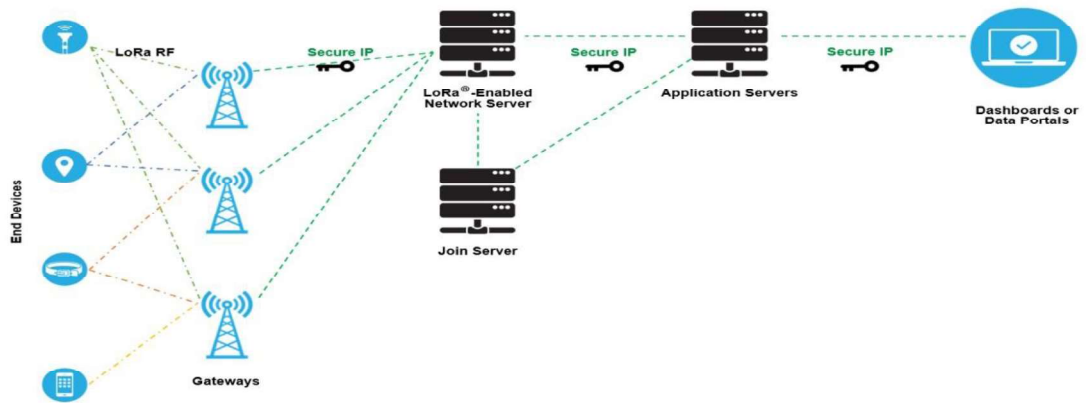


Figure 7: Typical LoRaWAN implementation

## **Chapter 3: System Design**

In this chapter, a brief overview of the whole system is given with the various requirements of the system. This chapter also elaborates on the multiple components of the tracking system design under each topic. The different comparisons made to decide on these components and to justify the design are also explained.

### **3.1 System Objective and Overview**

The objective of this project is to build an anti-theft vehicle tracking system that can detect an intruder, send a notification to the owner of the car, and shut down the car when the owner instructs it to. The device should be able to track the real-time location of the vehicle using a GPS module. As an extra feature, the system should be able to discern imminent danger by identifying when there are very high temperatures in the vehicle. To achieve this, the system needs a hardware component for sensing and other activities and a software component for data acquisition, monitoring, and controlling.

#### **3.1.1 Design Specifications**

1. Two LoRa modules to send and receive data to and from the LoRa gateway and the sensor node.
2. GPS model for receiving location data of vehicle and time from satellites
3. LoRa Gateway for sending data to a web server and for receiving data from the LoRa transceiver.
4. Database for storing the various data in the system
5. DC motor to serve as a model fuel injector
6. The device requires a battery to power up the system.

#### **3.1.2 Design Requirements**

##### **3.1.2.1. Hardware components requirement**

1. The tracking device should recognize whenever the car is being accessed.

2. The tracking device should be able to send notifications to the owner when the car is accessed.
3. The device should be able to determine the location of the vehicle using the GPS module.
4. The tracking device should get regulated power from a capable battery source.
5. The device should be capable of turning off the engine of the vehicle (DC motor) remotely when the owner instructs it to.
6. The GPS module in the system is required to communicate the LoRa gateway and send data packets to the clouds

### **3.1.3 Non-functional Requirements**

1. The device should be low-cost.
2. The device should have low power consumption.
3. System should be robust.
4. System should be sensitive to inputs.
5. Must be reliable.
6. Device should be portable.
7. Web application should be user-friendly

## **3.2 Design Decisions**

Comparisons were made for the hardware components of the system to decide on which component designs fit the system based on its requirements.

### **3.2.1 Processor boards comparison**

This table shows the possible processor boards for the project. From the comparison, the one most suited for the project is chosen.

Table 2: Processor board Comparisons

Criteria	Raspberry pi 4	ESP 32	ATmega 328p	Beaglebone	Nano BLE 33 Sense
Cost	High	Low	Low	High	High
Operating and GPIO voltage	3.3v	3.3v	5v	5v	3.3v
Analog pins	No	16	6(on Arduino)	6	8
RAM	1-4GB	520KB	32KB	4GB	1MB
Wi-Fi feature	No	Yes	No	Yes	No
Availability	Easily available	Easily available	Easily available	Uncommon	Easily available

From the comparison above, Raspberry pi was eliminated because of its high cost and unavailable analog pins since our system consists of sensors that will require analog pins. ATmega 328p was also not chosen due to its high operating power. Beaglebone was not an option either due to high power consumption or high cost. Esp32 was the appropriate processor for this project because its RAM size and analog pins were adequate. ESP32 also has a low cost and consumes less power. It also has different sleep modes that can be enabled to manage power consumption better. During the design decision state, ESP32 was chosen for the project. However, due to constraints and early availability of resources, the Nano BLE 33 Sense was used instead.



### 3.2.2 Pugh Matrix for Communication Technologies

These are dominant communication Technologies that can be considered for the project.

LoRa was chosen in the end since it suits the project well.

Table 3: Pugh Matrix of Communication Technologies

		Option-	WIFI	Bluetooth	LoRa	GSM/GPRS	ZigBee
Criteria	Baseline	Concept					
1	0	Cost	-1	0	+1	-1	0
2	0	Power saving	-1	+1	+1	-1	-1
3	0	Range	-1	-1	+1	+1	-1
4	0	Data rate	+1	+1	-1	-1	+1
Total			-1	2	3	-1	0

The LoRa communication was selected because its features suit the project when it comes to the different requirements of the system. Two major requirements of the system are prolonged battery usage and reduced cost. Since LoRa is the cheapest and has the lowest power consumption among these communication techniques, it is more suitable. Also, given that the purpose of the device is to track vehicles, it is required that the components of the device can communicate within wide ranges since the vehicle can be taken anywhere after the theft. With regards to data rate, LoRa has a low data rate due to the trade between the bits of data and the range of transmission. However, the main data for transmission in this project is data from sensors which also have a very low data rate. Therefore, the low data rate of LoRa does not greatly affect the project. The closest communication technique was Bluetooth. However, its range of communication is too short for the purpose of this project.

### 3.2.3 Pugh Matrix for GPS Module

To track the location of the vehicle, a GPS module is required to receive GPS data from available satellites. Four possible GPS modules were identified and compared to choose the most suitable module for the purpose of the project.

Table 4: Pugh Matrix of possible GPS Modules

		Option A	Option B	Option C	Option D
Criteria	Weight	Ublox NEO6 GPS	Ublox NEO7 GPS	GPS module Copernicus II DIP	SparkFun Venus GPS
Cost	3	0	-1	-1	
Availability	4	0	-1	-1	0
Power Consumption	2	0	0	0	-1
Versatility	1	0	0	0	0

### 3.3 Proposed design Overview

The system is divided into four parts which include a sensor node, a LoRa gateway, A Lora network stack (The Things Network), and a dashboard on Ubidots. To get sensor readings from the devices, an Arduino code is uploaded to the Nano BLE sense. The serial monitor of the Arduino IDE also prints out expected results for debugging purposes.

#### 3.3.1 Architecture of the overall proposed system

The diagram below shows the overall architecture of the system with the various functioning components of the device. In this project, the Things Network Stack serves as a cloud to make the sensor data available to the user for view on the web.

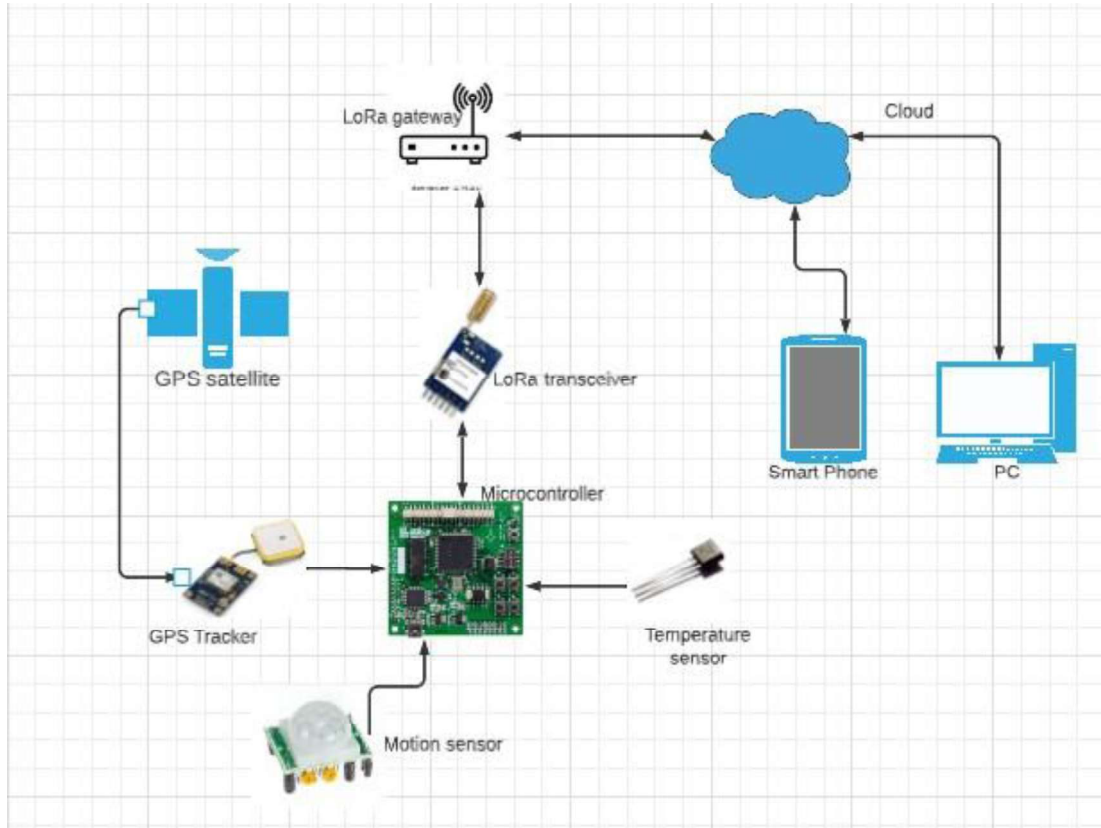


Figure 8: Architecture diagram of the proposed system

## Chapter 4: Methodology

### 4.1 Experimental Setup

The design of the system prototype for the vehicle tracking system consists of a gateway, A sensor node, and a dashboard for monitoring and controlling sensors on the sensor node.

### 4.2 Hardware Components for Sensor Node

This table includes the various components of the system and their specifications with respect to the purpose of the project.

Table 5: Hardware components and specs

HARDWARE COMPONENT	MODEL	SPECIFICATIONS
Development Board	Nano BLE sense	3.3v operating voltage, 15mA DC current per I/O pin, 64MHz clock speed, 1MB flash memory, 256KB SRAM.
GPS	NEO-6M GPS module	50 Channels, GPS L1 frequency, 2m position accuracy, 5 Hz update rate, -161 dBm tracking sensitivity, 8×6×4cm dimensions.
LoRa model	Reyax RYLR890	208 number of channels, 15km maximum range, 3.3v maximum voltage, 868915MHz frequency range, 43mA transmit current, 16.5mA receive current.
Temperature sensor (On Nano BLE sense)	HTS221	1.7 to 3.6v supply voltage, 2μA power consumption, ±0.5°C accuracy.
Battery	1200mAh Li-Ion Battery	9V normal voltage, 0-60°C operating temperature,

		17.5mm×48.5mm×26.5mm dimension.
Relay	REES52 5V	1 x 5V relay module, 10 grams weight.
DC motor	TT gearbox motor	150mA no-load current, 36v rated voltage, 90+/-10% RPM min operating speed (3v), 200+/-10% RPM operating speed(6v), 70×22×18mm.
Proximity Sensor (On Nano BLE sense)	APDS9960	2.4-3.8v input voltage, 0.2mA operating current, 400kHz data rate, 7m sensing range, 32×24mm dimension, -40 to 85°C operation temperature.

### 4.3 Overall Circuit of the system

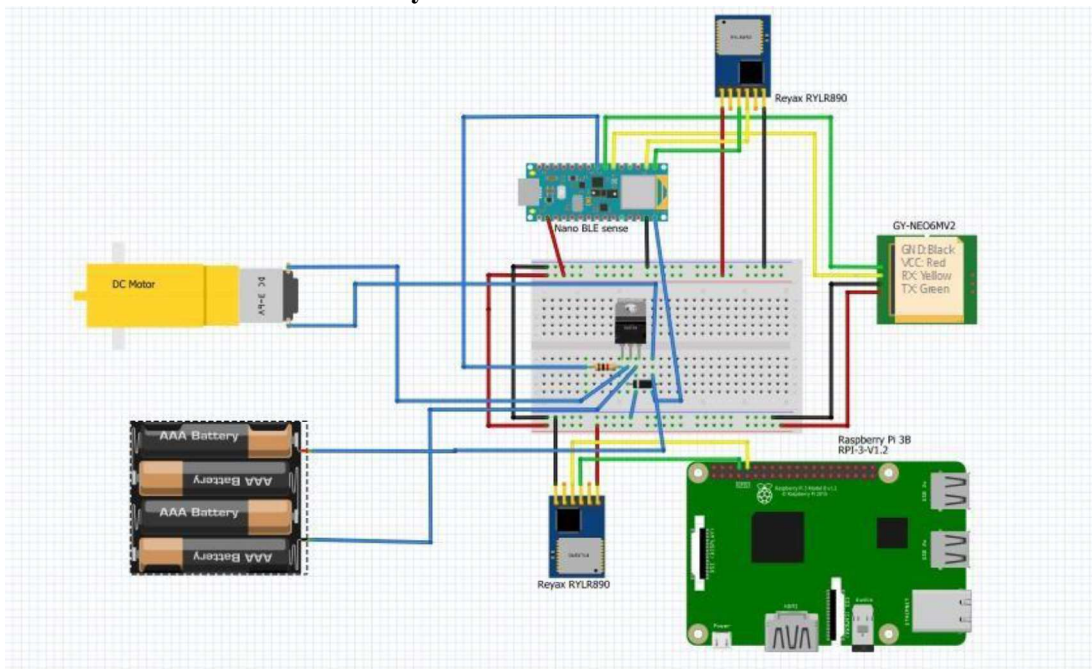


Figure 9: Circuit connections for system

To implement the anti-theft measures of the system, a dc motor is used to serve as a model fuel injector which will be turned on or off depending on the input of the system. To control

the DC motor, a diode and transistor are connected as an alternative to using a motor controller since the motor only needs to turn on and off, notwithstanding its speed for the purpose of this project. The efficiency of using a transistor to control the motor was tested by uploading an Arduino code.

#### 4.4 Nano BLE sense and Sensor node setup

The end node of the prototype was implemented using the Nano BLE sense development board and a REYAX RYLR896 Lora module. The Nano BLE development has an embedded proximity sensor and temperature sensor to detect gestures and high temperatures respectively. A GPS module is also connected serially to the Nano BLE sense to receive the GPS located data of the vehicle. Given that the Nano BLE sense has only one hardware serial pin, to connect both the lora module and the GPS module serially, two digital pins (4 and 3) on the development board were converted to serial pins using the following code.

```
#define GPS_RX_PIN 4
#define GPS_TX_PIN 3

UART gpsSerial(digitalPinToPinName(GPS_TX_PIN), digitalPinToPinName(GPS_RX_PIN), NC, NC);
TinyGPSPlus gps;
```

The reading from the sensors determines the response from the Nano BLE sense board and hence the output of the device. The device is programmed to identify threshold readings from the sensors, prompt the user and respond according to the user's input from the dashboard. The sensor node is programmed to send data from the various sensors to the Raspberry Pi gateway through radio frequency communication between the lora module of the gateway and the lora module of the sensor node. With regards to this two-way communication, the gateway first broadcasts query commands via the REYAX RYLR896 Lora module. The sensor node receives the requests and starts gathering readings from the

sensors and sends them to the lora model of the sensor node. After verifying the completeness of the received data, the gateway sends the data to the Things Network stack for access by the web application.

Table 6: Threshold and User Actions

<b>Devices</b>	<b>Possible user Actions</b>	<b>Thresholds</b>
GPS model	Turn ON/OFF	No threshold
Motion Sensor	Turn ON/OFF	Distance<30cm
Temperature Sensor	Turn ON/OFF	Temperature>70°C
Gas Sensor	Turn ON/OFF	Sensor value >300
DC motor	Turn ON/OFF	No threshold

Table 7: Thresholds and Notifications expected

<b>Thresholds</b>	<b>Emergency Notifications</b>	<b>User Action on web app</b>	<b>End-node response</b>
Distance<30cm	Motion Detected	1. Turn ON GPS if off 2. Turn OFF DC motor	1. Turn ON GPS 2. Turn OFF DC motor.
Temperature>70°C	Temperature too high	Turn OFF DC motor	Turn OFF DC motor.
Gas sensor value >300	Smoke Detected	Turn OFF DC motor	Turn OFF DC motor.

#### 4.5 Gateway Setup

The gateway consists of a Raspberry pi model 3 and one of the REYAX RYLR896 Lora modules. After flashing, the Raspbian OS is flashed onto the raspberry pi and configured it

to work headless, i.e., without a monitor, keyboard, or mouse for data transmission. The 3.3v and ground pins of the two devices were connected accordingly. The transmit (TX) pin of the raspberry pi was also connected to the receive (RX) pin of the Reyax module and vice versa for the Reyax module.

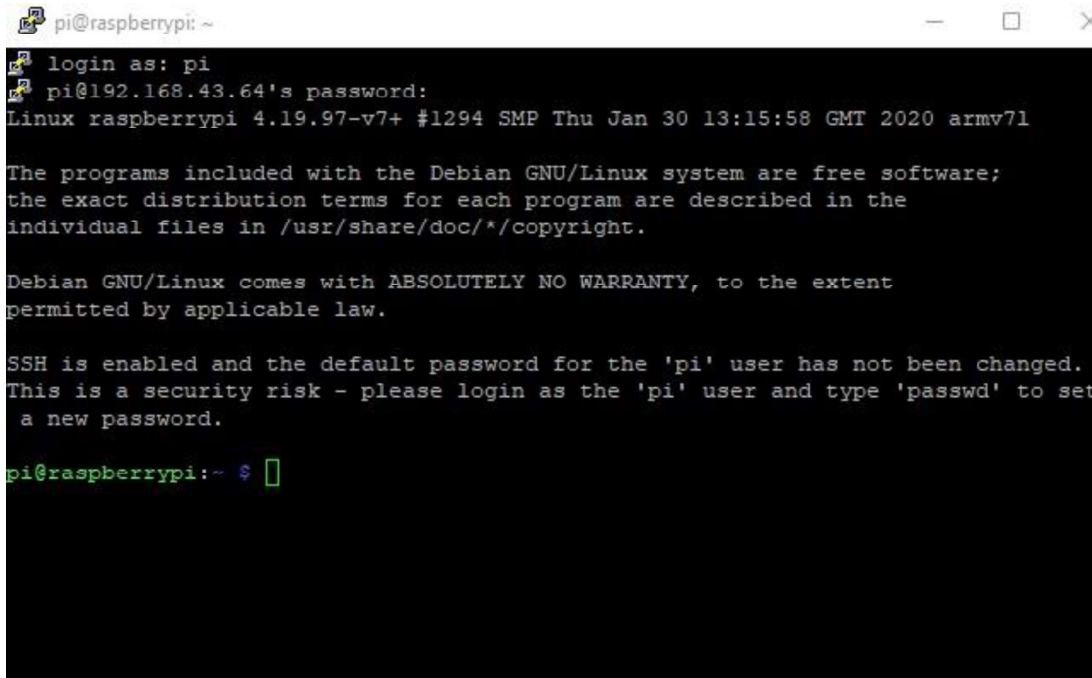
#### 4.5.1 Raspberry pi 3 setup

To use the raspberry pi as a gateway, the SD card of the raspberry was flashed with the raspberry pi OS using the raspberry pi imager. To configure the raspberry pi as headless, wireless networking and SSH are enabled when creating the pi image. This is done by creating an SSH file and a wpa\_supplicant.conf file in the boot folder of the SD card. In this project, the wpa\_supplicant.conf data used is shown below.

```
country=GH
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
network={
    ssid="Nana"
    psk=
    key_mgmt=WPA-PSK
}
```

Also, the AP band for this network was set to 2.4GHz, given that the raspberry pi module used does not support the dual AP band access. After this setup, a network is created between the laptop and the raspberry pi using a mobile hotspot with the SSID “Nana.” The “**ipconfig**” command is then issued on the command line to get the IP address of the raspberry pi, which is obtained as “**192.168.43.64**”. To program the raspberry pi gateway, The PuTTY software was used to control the raspberry pi terminal remotely using the IP address of the pi. Logging into the raspberry pi on the PuTTY is shown below.





```
pi@raspberrypi: ~
login as: pi
pi@192.168.43.64's password:
Linux raspberrypi 4.19.97-v7+ #1294 SMP Thu Jan 30 13:15:58 GMT 2020 armv7l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

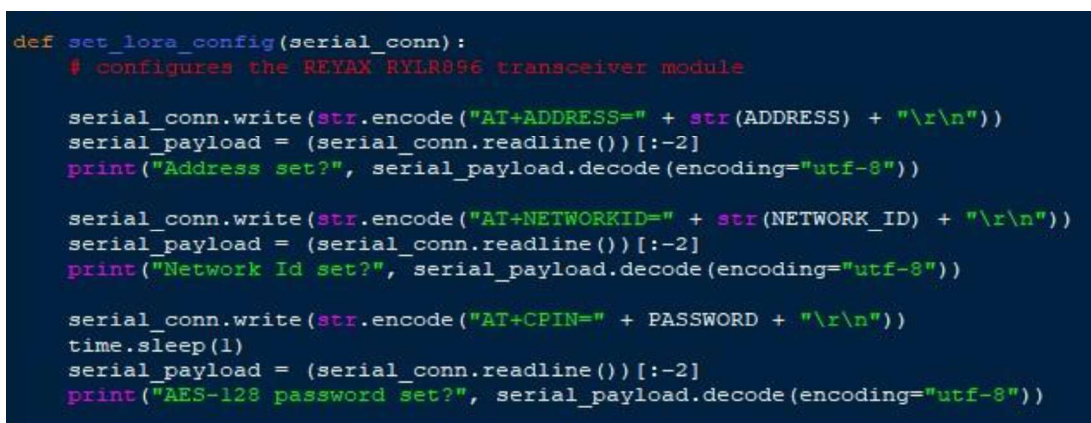
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.

SSH is enabled and the default password for the 'pi' user has not been changed.
This is a security risk - please login as the 'pi' user and type 'passwd' to set
a new password.

pi@raspberrypi:~ $
```

Figure 10:Raspberry pi connected to PuTTY via SSH

To configure the raspberry pi as a LoRa gateway, a python script is run on the raspberry pi using pycserial since the LoRa module will need access to the serial port of the raspberry pi. The python script receives and decrypts the data payload, parses the various sensor readings from the nano BLE sense, and displays the reading on the raspberry pi terminal. The function in the python script for configuring the Reyax module of the gateway to receive LoRa packets is shown below.



```
def set_lora_config(serial_conn):
    # configures the REYAX RYL886 transceiver module

    serial_conn.write(str.encode("AT+ADDRESS=" + str(ADDRESS) + "\r\n"))
    serial_payload = (serial_conn.readline())[:-2]
    print("Address set?", serial_payload.decode(encoding="utf-8"))

    serial_conn.write(str.encode("AT+NETWORKID=" + str(NETWORK_ID) + "\r\n"))
    serial_payload = (serial_conn.readline())[:-2]
    print("Network Id set?", serial_payload.decode(encoding="utf-8"))

    serial_conn.write(str.encode("AT+CPIN=" + PASSWORD + "\r\n"))
    time.sleep(1)
    serial_payload = (serial_conn.readline())[:-2]
    print("AES-128 password set?", serial_payload.decode(encoding="utf-8"))
```

Figure 11:Configuring LoRa on raspberry pi

The same configuration is implemented in the Arduino sketch of the sensor node, as shown below.

Figure 12: Connecting LoRa to Nano BLE sense on Arduino

```
// needs all need to be same for receiver and transmitter
Serial1.print((String)"AT+ADDRESS=" + ADDRESS + "\r\n");
delay(200);
Serial1.print((String)"AT+NETWORKID=" + NETWORK_ID + "\r\n");
delay(200);
Serial1.print("AT+CPIN=" + PASSWORD + "\r\n");
delay(200);
Serial1.print("AT+CPIN?\r\n"); // confirm password is set
```

#### 4.5.2 Connecting the Gateway to the Things Network (TTN) and Implementation

To make data from the sensor node available in the cloud, the data from the pi gateway received from the sensor node is sent to The Things Network stack. The TTN also makes the gateway readily available for use by other users nearby if the owner of the gateway allows it. To add to TTN, required information which includes gateway ID and gateway EUI (MAC address of the Raspberry pi), was filled in the TTN console. Finally, LNS configuration is done to link the gateway to the gateway server for traffic exchange. The gateway was added to the TTN stack, as shown below.

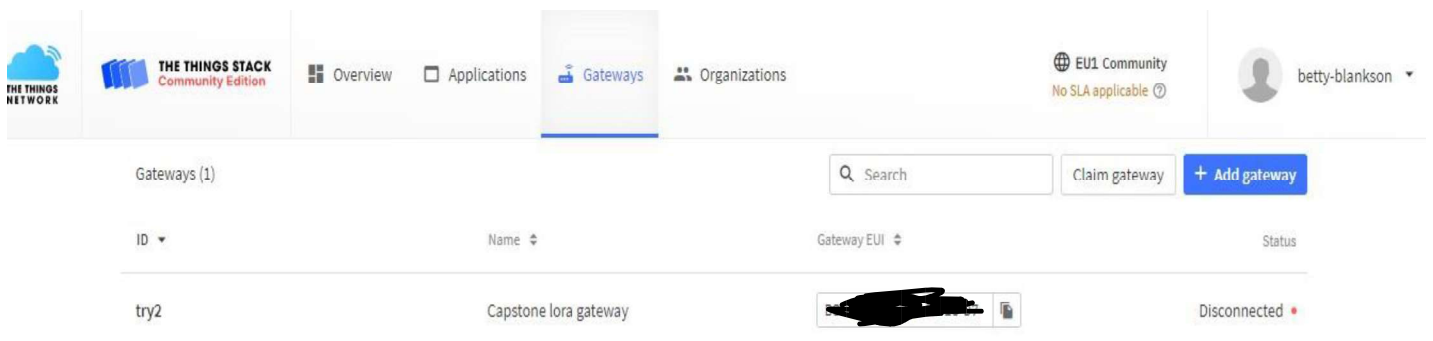


Figure 13: Adding Raspberry pi gateway to TTN stack

#### 4.6 Web Application using Ubidots

Ubidots is an IoT platforms that enable users to visualize data from IoT devices. It also enables users to trigger actions and alerts based on data from sensors. A dashboard with sensor data, switches, and alerts section is displayed on Ubidots. Users can then be allowed access to the dashboard after signing into the page. To get the data from the Things Network to Ubidots, the LoRaWAN payload on the Things Network is first decoded into Ubidots

format. The integration section of the Things Network stack is then implemented to connect Ubidots to the Things Network (TTN) stack.

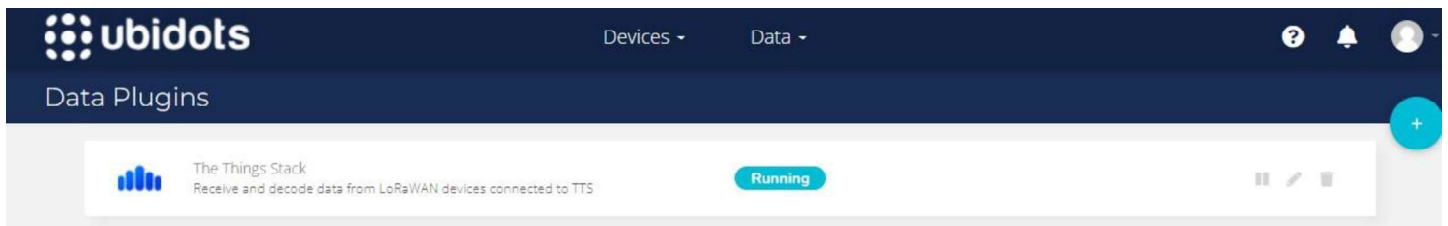
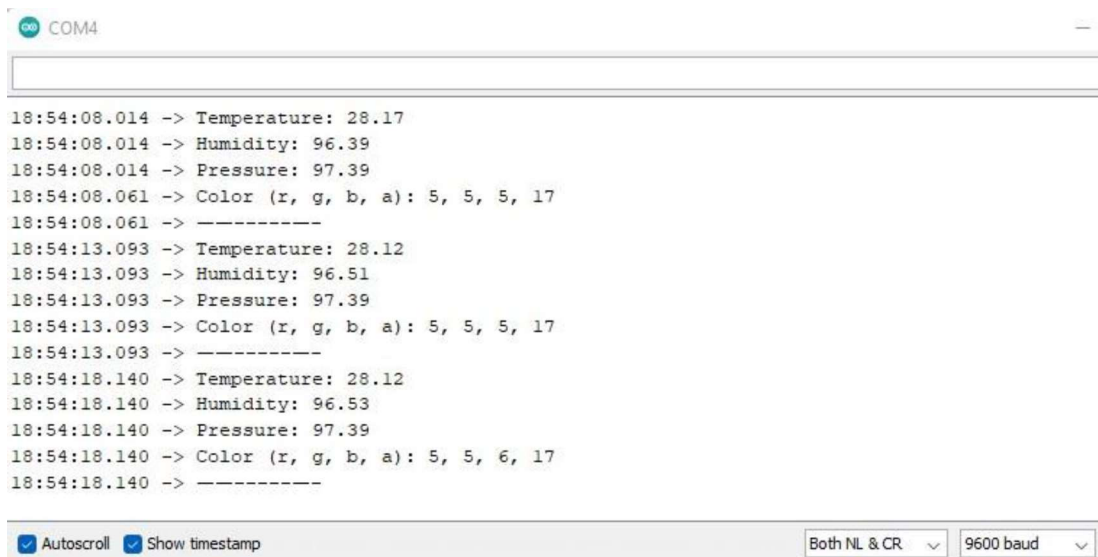


Figure 14:Ubidots connected to TTN

## Chapter 5: Results and Discussion

### 5.1 Single testing of sensors.

For debugging purposes, each sensor reading was displayed on the serial monitor of the Arduino ide. The temperature, pressure, and proximity sensor values are displayed. From the proximity sensor, when the values of (r, g, b, a) are below a given value, then the motion threshold is reached, and a “**motion detected**” message is sent.



```
COM4
18:54:08.014 -> Temperature: 28.17
18:54:08.014 -> Humidity: 96.39
18:54:08.014 -> Pressure: 97.39
18:54:08.061 -> Color (r, g, b, a): 5, 5, 5, 17
18:54:08.061 -> -----
18:54:13.093 -> Temperature: 28.12
18:54:13.093 -> Humidity: 96.51
18:54:13.093 -> Pressure: 97.39
18:54:13.093 -> Color (r, g, b, a): 5, 5, 5, 17
18:54:13.093 -> -----
18:54:18.140 -> Temperature: 28.12
18:54:18.140 -> Humidity: 96.53
18:54:18.140 -> Pressure: 97.39
18:54:18.140 -> Color (r, g, b, a): 5, 5, 6, 17
18:54:18.140 -> -----
```

Autoscroll  Show timestamp Both NL & CR 9600 baud

Figure 15:Sensor testing

Testing the gps module of the system also yielded the results shown below. The gps module was sometimes accurate and took a while before transmitting the right data.

Testing both the GPS model and lora model together on the nano ble sense posed a little challenge because the nano ble sense has only one hardware serial port; hence transmitting data from the lora and GPS serially at the same time caused intermittent disruptions and on a number of occasions, the Arduino sketch had to be re-uploaded.

```

COM4
Send
00:23:05.653 -> GPS data is given as :
00:23:05.653 -> -----
00:23:05.653 ->
00:23:05.653 -> Temperature:
00:23:05.653 -> 25.96
00:23:05.653 -> Humidity: 99.17
00:23:05.653 -> Pressure: 97.43
00:23:05.653 -> Color (r, g, b, a): 12, 13, 11, 32
00:23:05.653 -> -----
00:23:05.746 -> GPS data is given as :
00:23:05.746 ->
00:23:05.746 -> $GPTXT,01,01,02,HW  UBX-G60xx  00040007 FF7FFFFFFp*53-----
00:23:05.746 ->
00:23:05.746 -> Temperature:
00:23:05.746 -> 26.00
00:23:05.746 -> Humidity: 99.07
00:23:05.746 -> Pressure: 97.43
00:23:05.746 -> Color (r, g, b, a): 12, 13, 11, 32
00:23:05.746 -> -----
00:23:05.794 -> GPS data is given as :
00:23:05.794 ->
00:23:05.794 -> $GPTXT,01,01,02,HW  UBX-G60xx  00040007 FF7FFFFFFp*53-----
00:23:05.794 ->
00:23:05.794 -> Temperature:
00:23:05.794 -> 25.98
00:23:05.794 -> Humidity: 99.09
00:23:05.794 -> Pressure: 97.42
00:23:05.794 -> Color (r, g, b, a): 12, 13, 11, 32
00:23:05.794 -> -----

```

Figure 16:GPS and sensors testing

## 5.2 Testing Thresholds of sensors

To test the messages expected when thresholds are reached, the anticipated messages are included in the Arduino code to print on the serial monitor. In this project, a temperature threshold of 30°C and (r, g, b, a) values < 5 thresholds for the colors are used for testing purposes. Hence an object placed in front of the proximity sensor reduces the (r, g, b, a) colors which trigger the message “**motion detected**” while “**Vehicle temperature is too high**” is displayed when the surrounding temperature goes above 30°C

```
00:46:02.364 -> Humidity: 98.81
00:46:02.364 -> Pressure: 97.41
00:46:02.364 -> Color (r, g, b, a): 1, 1, 1, 3
00:46:02.364 -> -----
00:46:02.364 -> Motion detected
00:46:08.390 -> Temperature: 27.54
00:46:08.390 -> Humidity: 98.03
00:46:08.390 -> Pressure: 97.41
00:46:08.390 -> Color (r, g, b, a): 1, 1, 1, 2
00:46:08.390 -> -----
00:46:08.390 -> Motion detected
00:46:14.442 -> Temperature: 28.35
00:46:14.442 -> Vehicle temperature high
00:46:15.426 -> Humidity: 96.72
00:46:15.426 -> Pressure: 97.41
00:46:15.426 -> Color (r, g, b, a): 1, 1, 1, 2
00:46:15.473 -> -----
00:46:15.473 -> Motion detected
00:46:21.482 -> Temperature: 28.75
00:46:21.482 -> Vehicle temperature high
00:46:22.516 -> Humidity: 95.27
00:46:22.516 -> Pressure: 97.41
00:46:22.516 -> Color (r, g, b, a): 12, 13, 11, 32
00:46:22.516 -> -----
00:46:27.527 -> Temperature: 27.94
00:46:27.527 -> Humidity: 94.35
00:46:27.527 -> Pressure: 97.41
00:46:27.527 -> Color (r, g, b, a): 12, 13, 11, 32
00:46:27.527 -> -----
00:46:32.576 -> Temperature: 27.65
00:46:32.576 -> Humidity: 94.32
00:46:32.576 -> Pressure: 97.41
00:46:32.576 -> Color (r, g, b, a): 12, 13, 11, 32
00:46:32.576 -> -----
00:46:37.643 -> Temperature: 27.56
```

Figure 17: Testing thresholds

## **Chapter 6: Conclusion**

### **6.1 Summary**

In this project, there is a comprehensive explanation of how a low-cost anti-theft vehicle tracking system with safety features is implemented. The device was implemented using the LoRa communication technology. Users of the device will be able to check the location of their vehicles and know some safety conditions of the vehicle, i.e., whether there is an attempt of theft or whether there is an abnormally high temperature in their vehicle. The main devices of the project are hence a GPS module and two LoRa modules for data transmission between the sensors and the cloud.

With this system implemented, more vehicle tracking systems that use the LoRa communication technology (which is a cheaper form of communication) will significantly reduce the theft of vehicles since more users will be able to purchase their cars due to affordability.

### **6.2 Limitations**

- During the start of the project, the ESP32 microcontroller was chosen as the required board for the project. However, the availability of devices caused an improvise in using the nano 33 BLE sense, which reduced the efficiency of the project, especially with respect to serial communication.
- Also, implementing a hardware gateway using raspberry pi as opposed to buying an already configured lora gateway also posed a challenge because the SD card got

corrupted a number of times during implementation hence causing delays with project progress.

### **6.3 Lessons Learnt**

Aside that I learned a lot since most of the IoT concepts and the devices in my project were new to me; while carrying out this project, I have also learned that:

- When carrying out an engineering project it is very important to have alternate devices and concepts to fall on because there could be the need to improvise when components are unavailable.
- In engineering, planning is essential when carrying out projects since good planning makes implementations much easier.

### **6.4 Future Works**

To improve the efficiency of the implementation of the project, a number of changes or additions to the project are as follows:

- Due to the challenge faced with two serial communications at the same time, a more suitable microcontroller board that has more than one hardware serial port will be used in future implementations.
- Another GPS module with better accuracy and at a low cost can also be explored since the GPS module in this project was not 100% efficient. Since a more accurate location is preferred when locating a stolen vehicle using the device.
- An already configured gateway such as the Dragino LORAWAN gateway can also be explored, even though this may raise the total cost of the device.



## References

- [1] Federal Aviation Administration (2021, August 12). *Satellite navigation-GPS how it works*. [https://www.faa.gov/about/office\\_org/headquarters\\_offices/ato/service\\_units/techo ps/navservices/gnss/gps/howitworks/](https://www.faa.gov/about/office_org/headquarters_offices/ato/service_units/techo ps/navservices/gnss/gps/howitworks/)
- [2] Insurance Information Institute (2021). *Facts + Statistics: Auto theft*. <https://www.iii.org/fact-statistic/facts-statistics-auto-theft>
- [3] Maurya, K., Singh, M., & Jain, N. (2012). Real Time Vehicle Tracking System using GSM and GPS Technology-An Anti-theft Tracking System. *International Journal of Electronics and Computer Science Engineering*, 1, (1103-1107). [https://www.researchgate.net/publication/266412980\\_Real\\_Time\\_Vehicle\\_Tracking\\_System\\_using\\_GSM\\_and\\_GPS\\_Technology-An\\_Anti-theft\\_Tracking\\_System](https://www.researchgate.net/publication/266412980_Real_Time_Vehicle_Tracking_System_using_GSM_and_GPS_Technology-An_Anti-theft_Tracking_System)
- [4] Ray, B. (2018, October 29). *What Is LoRaWAN? [Technical Breakdown]*. Link labs. <https://www.link-labs.com/blog/what-is-lorawan>
- [5] Salim, K., & Idrees, I. (2013). Design and Implementation of Web-Based GPS GPRS Vehicle Tracking System. *International Journal of Electronics and Computer Science Engineering*, 1(12), (443-448). [https://www.researchgate.net/publication/310267800\\_Design\\_and\\_Implementation\\_of\\_Web-Based\\_GPS-GPRS\\_Vehicle\\_Tracking\\_System](https://www.researchgate.net/publication/310267800_Design_and_Implementation_of_Web-Based_GPS-GPRS_Vehicle_Tracking_System).
- [6] Semtech. (n.d.). *What are LoRa® and LoRaWAN®*. LoRa developer portal. <https://lora-developers.semtech.com/documentation/tech-papers-and-guides/lora-andlorawan/>
- [7] Shinde, P., & Mane, Y. (2015). Advanced Vehicle Monitoring and Tracking System based on Raspberry Pi. *IEEE 9th International Conference on Intelligent Systems and Control (ISCO)*, 1-6. <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7282250&tag=1a>
- [8] Stafford, G. (2020, August 10). *LoRa and LoRaWAN for IoT: Getting Started with LoRa and LoRaWAN Protocols for Low Power, Wide Area Networking of IoT*. Medium. <https://programmaticponderings.com/tag/raspberry-pi/>