Project Engineering

Key Tracker

Sergey Moiseenko

Bachelor of Software & Electronics Engineering

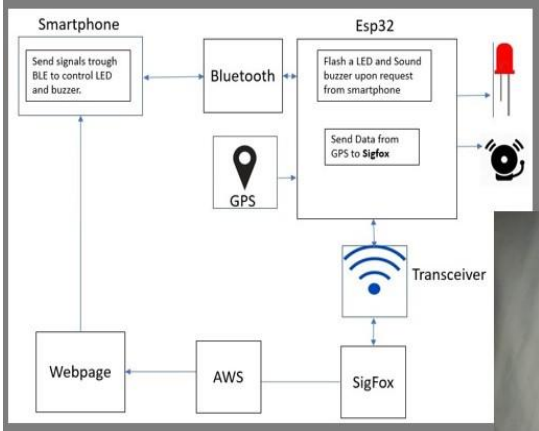
Galway-Mayo Institute of Technology

2020/2021

GMIT
INSTITIÚID TEICNEOLAÍOCHTA NA GAILLIMHE-MAIGH EO
GALWAY-MAYO INSTITUTE OF TECHNOLOGY

Sergey Moiseenko
BcEng Software and Electronic Engineering Year 4 Project

Key Tracker



Technology used
- Sigfox
- ESP 32
- Gps
- Bluetooth
- AWS

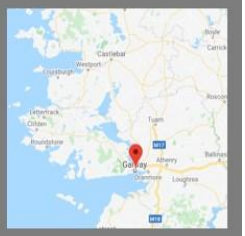
Time	Seq Num	Data / Decoding	LQI	Callbacks	Location
2021-04-07 21:55:41	9	53f735790a8f99580000 ASCII: S.5y...X..			

Project idea: A device that reads GPS co-odrdinates and sends it over Sigfox network to their backend.

From sigfox backend, gps cordinates are redirected to aws cloud where they are stored in the database.

App takes coordinates from the database and places marker on the map.

User can connect to the ESP via Bluetooth to control LED and buzzer.



Outcome:
ESP sends gps Co-ords trough transceiver. Message is redirected to aws s3 bucket (Simple Storage Service) and redirects it to the databese.

Github Project page

Table of Contents

Declaration

This project is presented in partial fulfilment of the requirements for the degree of Bachelor of

Engineering in Software & Electronic Engineering at Galway-Mayo Institute of Technology.

This project is my own work, except where otherwise accredited. Where the work of others has been used or incorporated during this project, this is acknowledged and referenced.

Summary

“Key Tracker” is a device that can track the location of a key or other important things that you do not want to lose. I chose this project because I often put my keys somewhere and then forget where they are. This makes me nervous thinking that I have lost them. Sometimes I can forget my keys in someone's home making keys impossible to find until the house owner tells me about it. I think my project can solve the problem of not knowing where you left your keys and save you from being nervous about it.

I made this project using ESP32 as a microcontroller and Arduino IDE to program it with C++ language.

I am interfacing different hardware components with it, including GPS module, special “Sigfox” transceiver, ESP32 inbuild Bluetooth, LED, and buzzer.

I am using “Sigfox” network and Amazon cloud services to output data on the internet.

Introduction

The goal of my project is to create a tracking device which will help people to keep track of the thing they do not want to lose.

Key tracker is an automatic device and do not need user to configure it or do manipulations with it other than changing battery and connecting to it via Bluetooth.

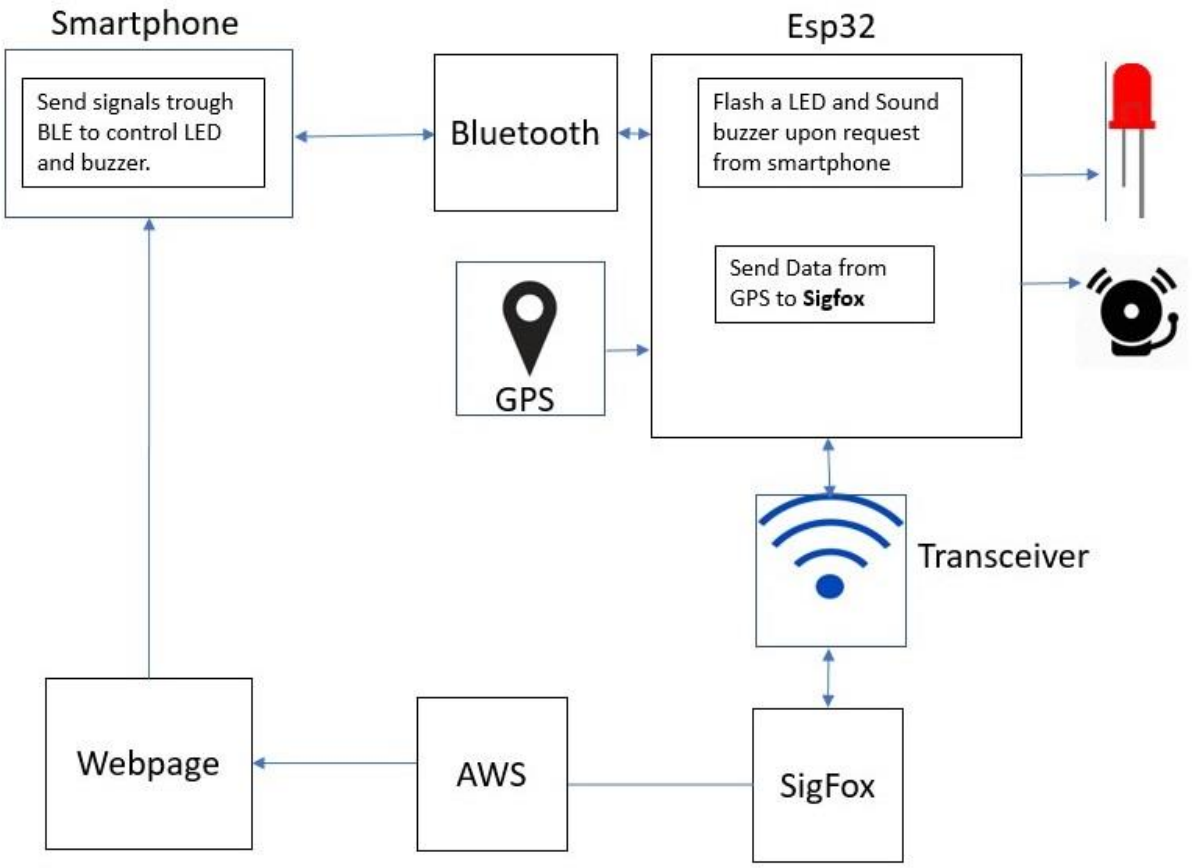
Key tracker will read GPS coordinates, ESP32 will process them and redirect them to Sigfox transceiver. Transceiver will send coordinates over LPWAN (Low-Power Wide Area Network) to Sigfox cloud.

When the message gets received in the cloud, it automatically redirected to Amazon S3 bucket (Simple Storage Service). S3 bucket redirects it to Amazon DynamoDB (database). Another S3 bucket is used to host a static website. DynamoDB sends its contents to the website where those coordinates are used to place a marker on a map.

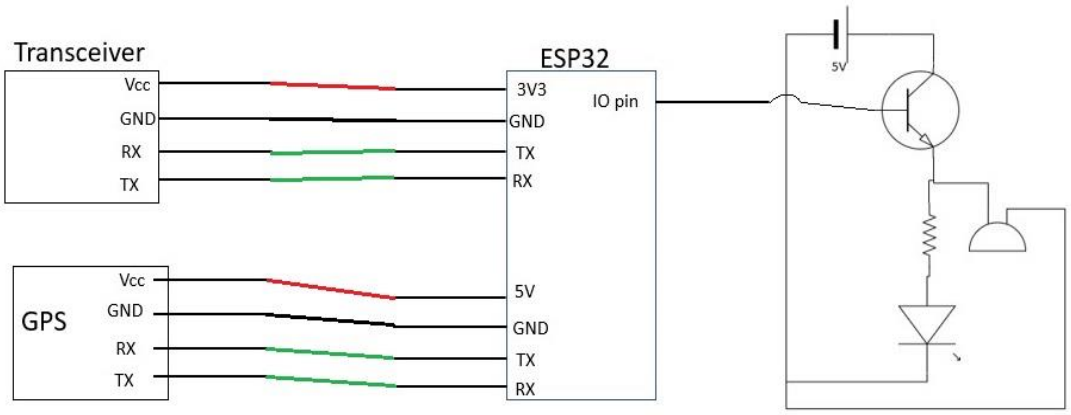
Additionally, User can connect to the device with a Bluetooth to control LED and buzzer. This will help user to locate the device when it is near to the user but cannot be seen right away.

Project Architecture

Block Diagram



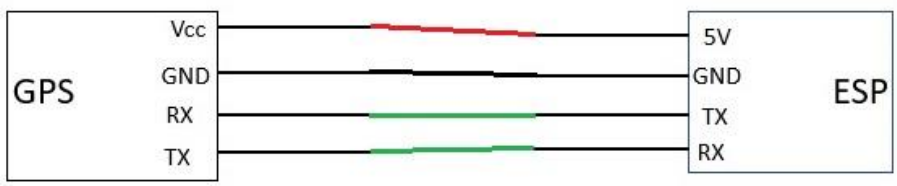
Wiring diagram



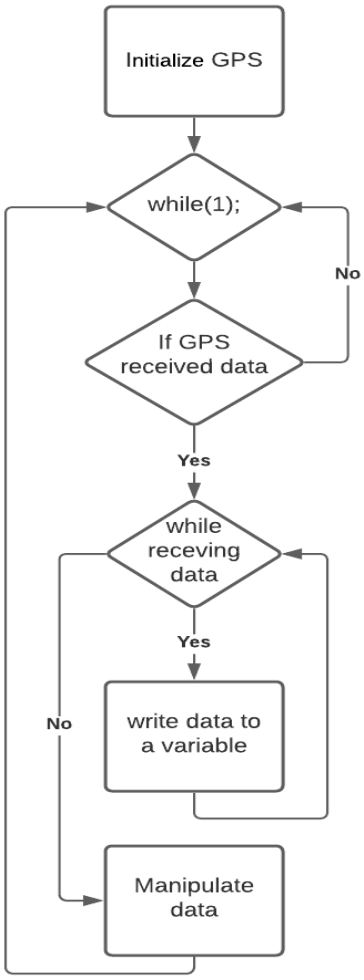
GPS

GY-NEO6MV2 is the GPS module I used. It is compatible with 3-5V power supply, and the default baud rate is 9600. [GPS Datasheet]

Wiring diagram



Code Flowchart



Initialize GPS mean set up RX/TX pins on ESP32, Setup the Baud rate.

If the GPS will get a message coming in it would go to a while loop.

Else it will wait for the message to be received or do other code (not specified here).

Second while loop is needed because GPS is receiving one character at a time. Every time it receives a character it must be stored otherwise it will be lost.

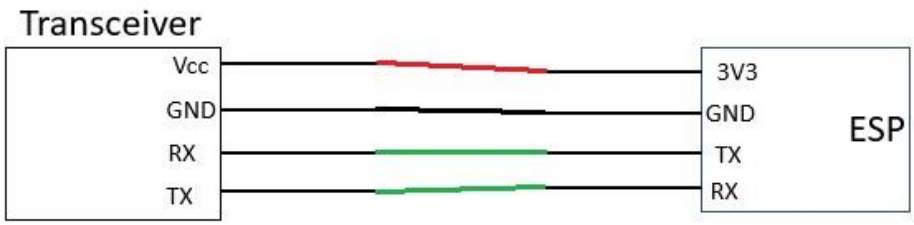
A string or a character array will be suitable for this purpose. Each character is appended to the variable. Then we can use this variable to manipulate data inside of it.

GPS messages contains a lot of information we don't need so we need to extract the latitude and longitude data from the message so that we can process it further

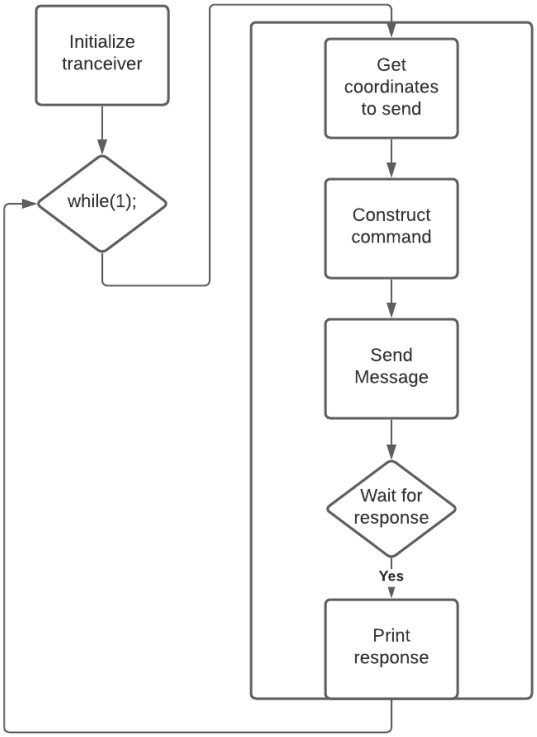
Transceiver

I used SNOC SFM10R1 Sigfox Breakout Board. This module requires 3V3 power supply. Its baud rate defaults to 9600. The module is controlled via serial AT commands. [Transeiver Datasheet]

Wiring diagram



Code Flowchart



Initializing mean set up RX/TX pins on ESP32, Setup the Baud rate.

When time comes to send message. Code takes the variable with coordinates. Then it is needed to construct a string that will contain AT command

”AT$SF<co-ords>\n”.

\n is required to notify that the command is finished, and the chip can proceed executing it. When module finish sending it will wait to receive a response ”200 Ok”.

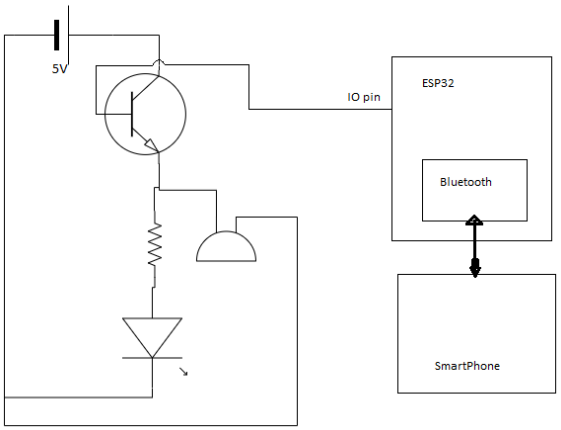
After receiving a successful response, the code will procced to execute other code (not specified here).

Reference code for sending messages [1]

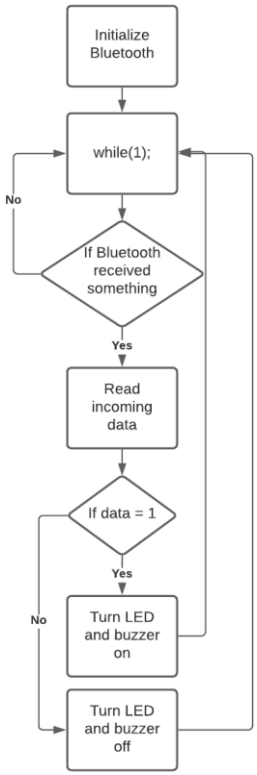
Bluetooth, LED and Buzzer

I am using a Bluetooth that is inbuild into ESP32 board. ESP32 Datasheet. LED and buzzer are connected to the ESP32 trough a transistor (Explained in second problem in problem solving section). Bluetooth is paired with a smartphone that has premade app **[2]**. LED and buzzer start to signal when button is pressed it the app.

Wiring Diagram



Code flowchart



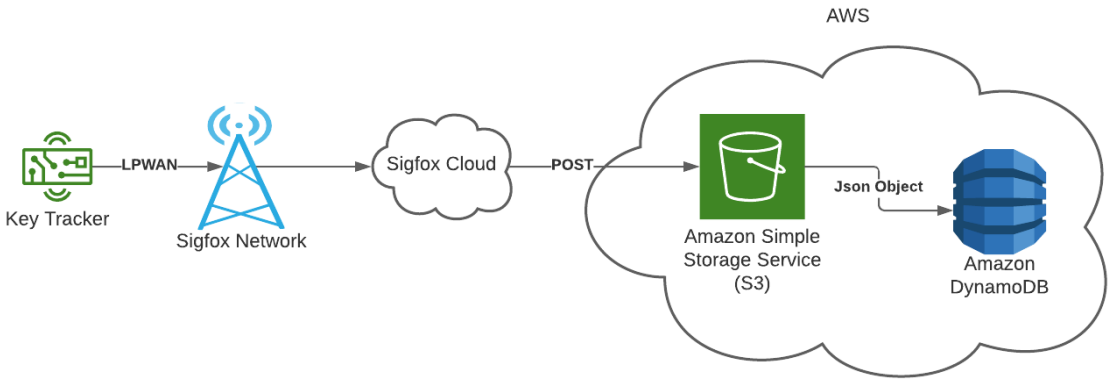
Using ”BluetoothSerial.h” library **[3]** to setup Bluetooth.

Code checks if there are data coming. I am sending character “1” or “0” from then app on smartphone. Code checks which character is it.

If its 1, turn LED and buzzer on. If its 0, turns LED and buzzer off.

Amazon and Sigfox

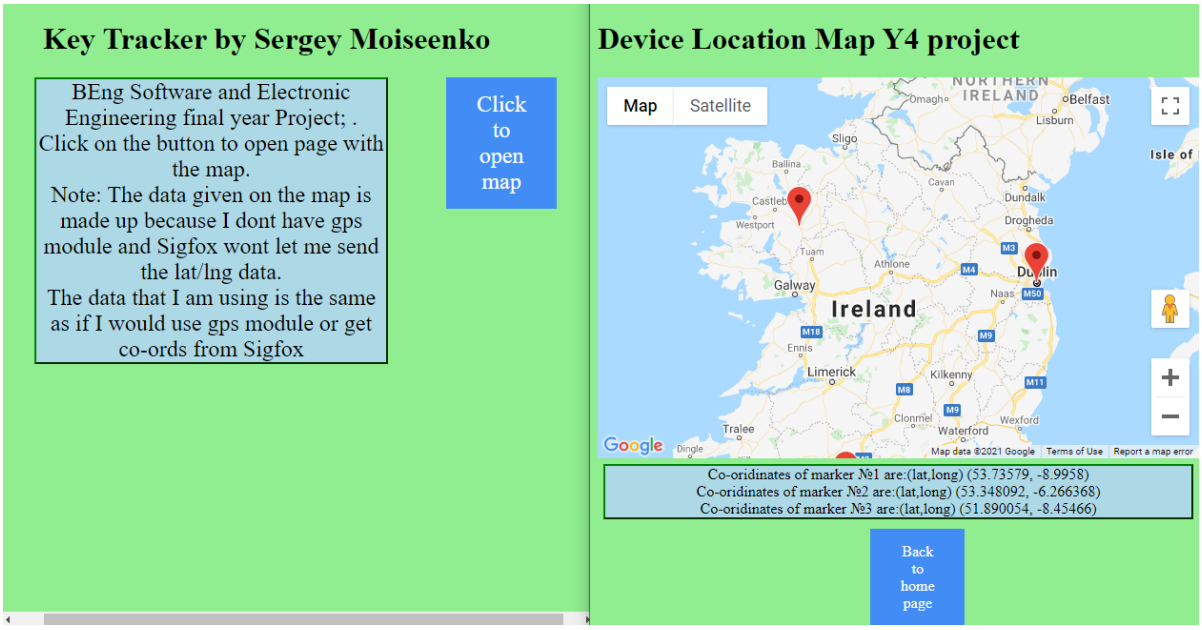
When Message from the device reaches Sigfox cloud. It triggers a callback service that sends this message to Amazon S3 bucket **[4]**. Message is sent via POST request and the message itself is a Json object that contains device information such as ID time stamp and other payload data which includes the co-ordinates that we got from the GPS. S3 buckets forwards this Json object to DynamoDB where it stored in a table.



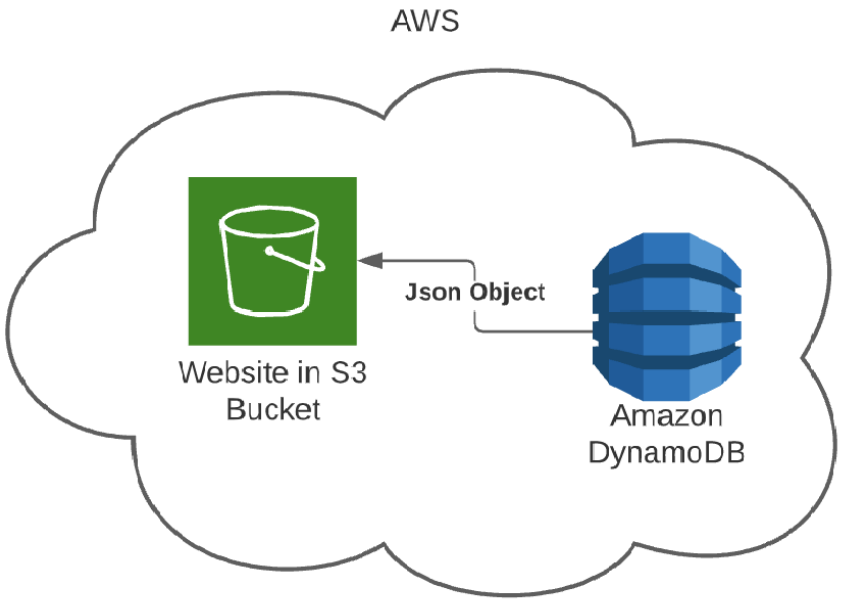
Sigfox uses LPWAN to communicate with the devices. This method allows reliable connection even in places that have no network because it is using low frequencies signal (200 kHz) that travel very far and it is harder to absorb.

Website

It is a 2-page html site that is static hosted by Amazon S3 bucket It has a home page and a page with a google map.



DynamoDB takes everything from its table with the data and sends it to S3 Bucket as one Json object file. The website should have (see sixth problem in problem solving section) read this Json file and extract co-ordinates and time data from it. Based on that data places a marker on a map and display information about it at the bottom of the map. Numeration of co-ordinates would be based on time data.



Problem Solving

First problem that I encountered during work on this project was exceptionally long components delivery. I had to adapt to the situation by adjusting the project plan to address this problem. Also, because I could not solve the issue all I could is to minimize the effects it might have on the overall result. To minimize the problem, while I was waiting for the components, I did majority of the research on every component. When they arrived, I already had a clear idea on how I should use each component.

Second problem was getting buzzer to work. When the buzzer arrived, it turned out that it was 5V buzzer while the IO pins on ESP32 are 3V3. Therefore, I could not wire buzzer directly to the ESP. To solve This problem, I found old broken electric toy car. I cut out battery compartment from it so that I could use it as an external power. To wire it up safely to the ESP32 IO pins I found a transistor which I used as a switch and wired the transistor's base pin to the IO pin, Collector pin to the power source and emitter to the buzzer. To reduce complexity, I wired-up LED same way so that one transistor could turn on LED and buzzer at the same time.

Third problem was getting GPS to work. I could not get any response from the GPS I did serval testing to check if I were doing something wrong or the module was faulty. I produced the conclusion that module is unusable either because of the module was faulty or the antenna for this module was too weak and it could not pick up any signal. I could not order new one, so I faked GPS coordinates that I am sending.

Forth problem was getting transceiver to send messages with GPS coordinates to Sigfox cloud. To solve it I had to divide this task into three subtasks. First task was to use simple AT commands that does not require transceiver to send anything. Completing this task verified that the AT commands work, and the module is responding to them correctly. Second subtask was to send a small and simple message to verify that messages getting received in the cloud. Final task was to send GPS coordinates. Result the module works as intended.

Fifth problem was connecting Sigfox cloud to external cloud. Initially I was planning to use Google cloud as an external cloud service. Google cloud interfacing with Sigfox was extremely complicated. I have spent most of the time I have planned for this task to get it to work. In the end when I almost ran out of time and still could not solve the issue, I started to look for alternatives to Google cloud and found that Amazon has better a much simpler APIs to interface with Sigfox. So, I moved to AWS instead of Google.

Sixth problem was with getting website to read Json file with coordinates from the database. I tried to solve this issue by using react-native mobile app because it is better at reading Json files, but I ran out of time that I have planned for the project, so I had to leave it as it is.

Conclusion

Key Tracker can send GPS coordinates to Sigfox cloud where user can see the location of the device. Message is redirected to Amazon cloud where it gets stored in the database. User can control LED and buzzer with a Bluetooth

In future if I get to do this project again, I will pick a better GPS module with a good antenna. I will change 5V buzzer to 3V3 and make a single mobile app from where the user would be able to see the map with the device location and will be able to control LED and Buzzer from the same app.

References

[1] Reference code to send messages to Sigfox cloud

[2] Premade Bluetooth app

[3] BluetoothSerial.h

[4] Sigfox To AWS