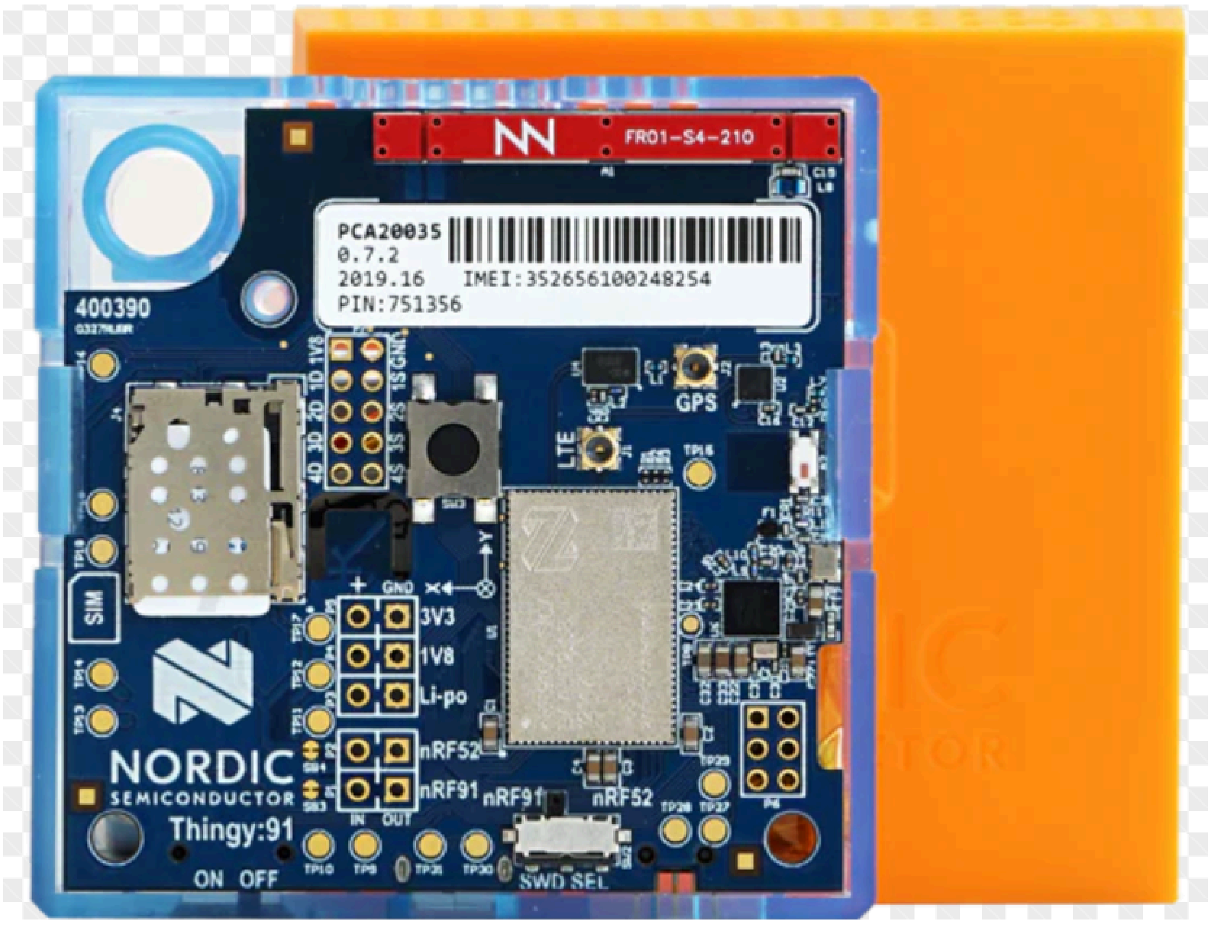
Thingy 91 Mini project

IoT Communication and Infrastructure



Date: 13-06-2023

Paras khosla - 4289382

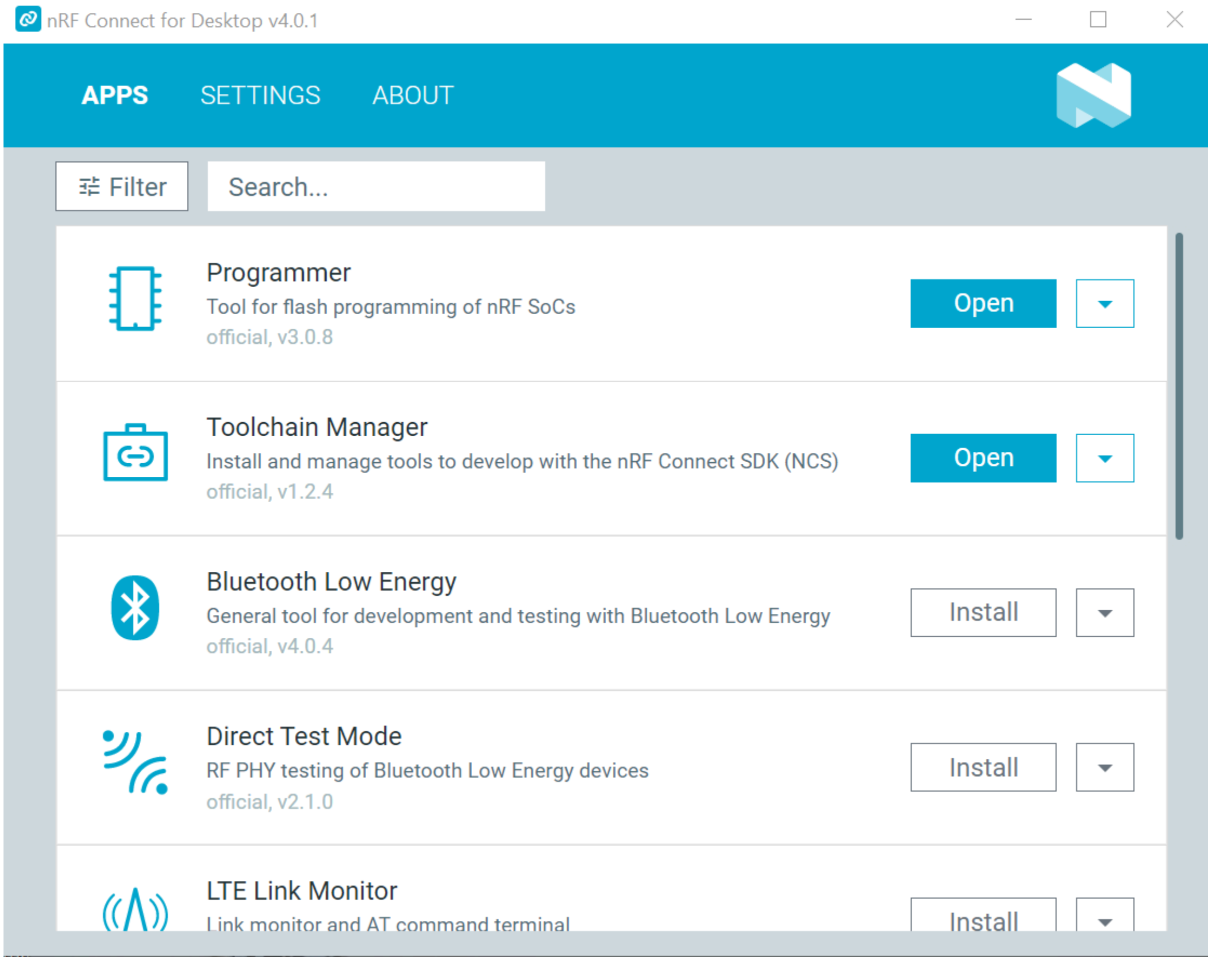
Teachers:
Renata Frenken
Mark Beks

Introduction

For the mini project we are going to work on Thingy 91 microcontroller is an advanced and versatile development platform designed to empower developers in creating innovative and connected Internet of Things (IoT) solutions. The key strengths of the Thingy 91 lies in its built-in connectivity options. It supports multiple wireless technologies, including LTE-M, NB-IoT, and GNSS, providing reliable and efficient connectivity for IoT deployments across different geographical regions.
Furthermore, the Thingy 91 boasts impressive integrated sensors, such as temperature, humidity, air quality, air pressure and motion sensors, among others. These sensors enable developers to gather valuable environmental data. All the steps taken while completing the tasks of this project will be described in the following paragraphs.

Procedure

To begin with, first we uploaded the correct firmware to start using for our further tasks with the help of a boot loader by installing the nrf connect software to perform all other tasks with thingy 91. The supported software with thingy 91 we installed can be seen in the following image.


nRF Connect for Desktop v4.0.1

APPS SETTINGS ABOUT

Filter Search...

Programmer
Tool for flash programming of nRF SoCs
official, v3.0.8
Open

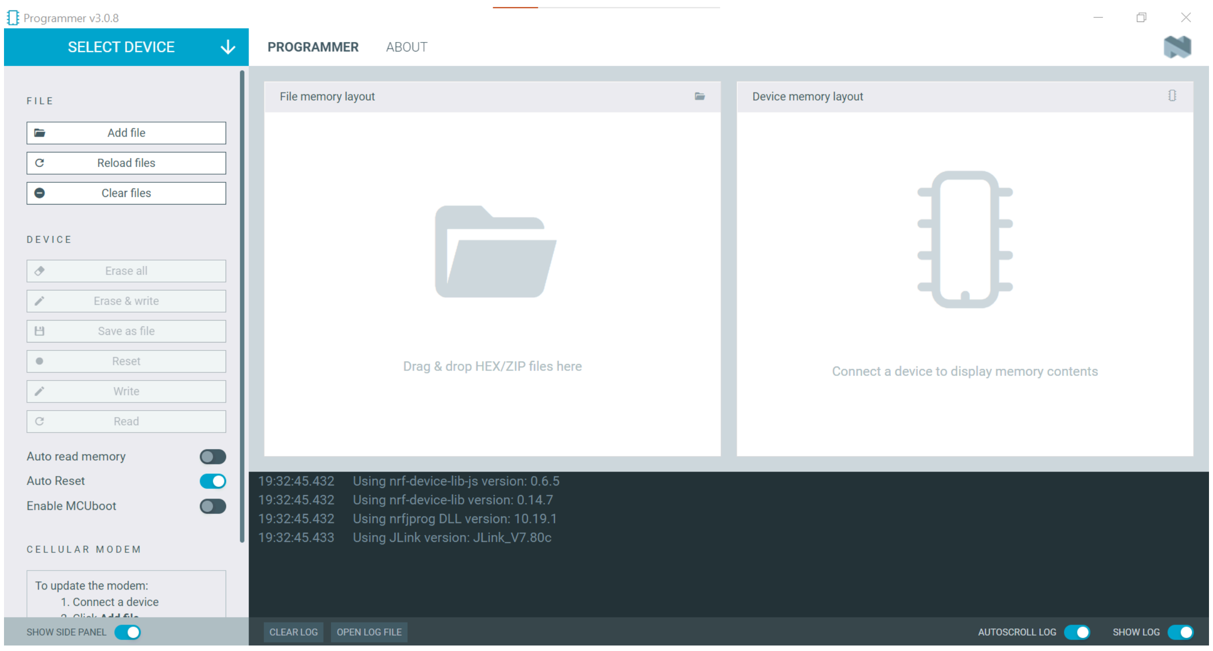
Toolchain Manager
Install and manage tools to develop with the nRF Connect SDK (NCS)
official, v1.2.4
Open

Bluetooth Low Energy
General tool for development and testing with Bluetooth Low Energy
official, v4.0.4
Install

Direct Test Mode
RF PHY testing of Bluetooth Low Energy devices
official, v2.1.0
Install

LTE Link Monitor
Link monitor and AT command terminal
Install

There are two extensions needed for right now for the thingy 91, the first one is programmer where you can update the firmware versions and the other one is toolchain manager that helps to connect VS code with nrf connect environment. The programmer extension can be seen in the following image where you can upload a firmware file to the thingy 91.

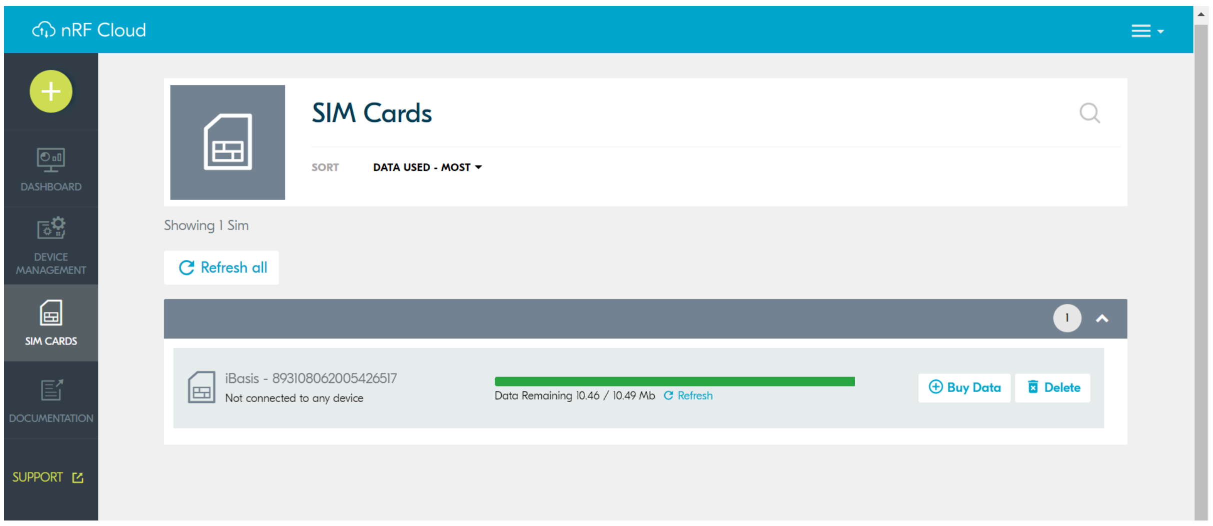


Auto read memory: off
Auto Reset: on
Enable MCUboot: off

To update the modem:
1. Connect a device

```
19:32:45.432   Using nrf-device-lib-js version: 0.6.5
19:32:45.432   Using nrf-device-lib version: 0.14.7
19:32:45.432   Using nrfjprog DLL version: 10.19.1
19:32:45.433   Using JLink version: JLink_V7.80c
```

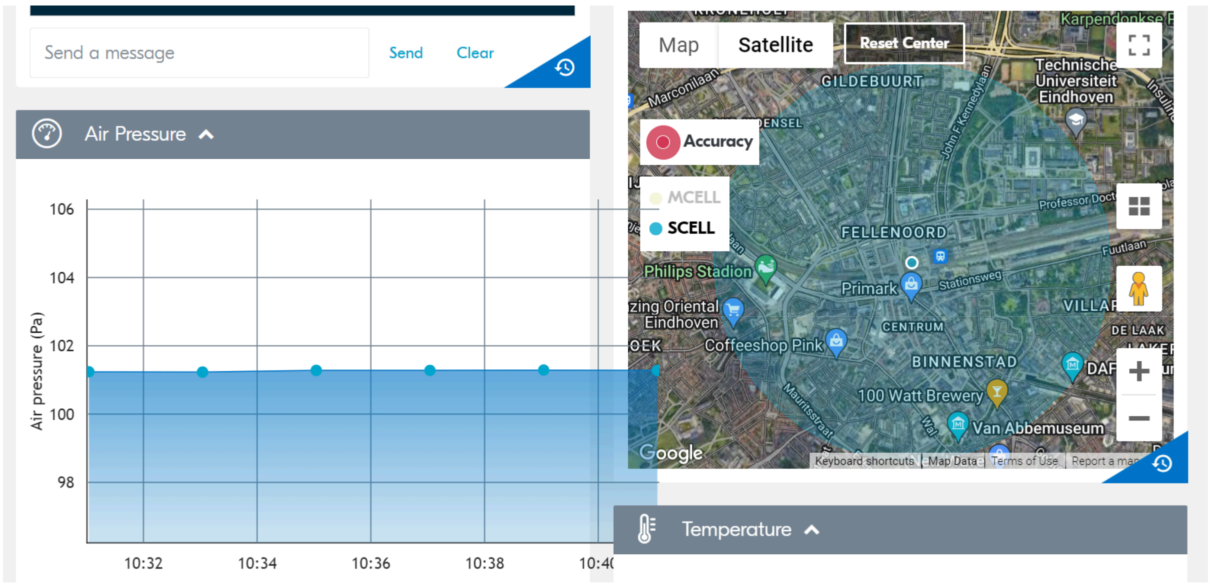
After that, we registered the sim card to nrf cloud to enable the Lte and start getting values from the thingy 91 via nrf cloud dashboard.
In the following image you can see the registered sim card on nrf cloud.



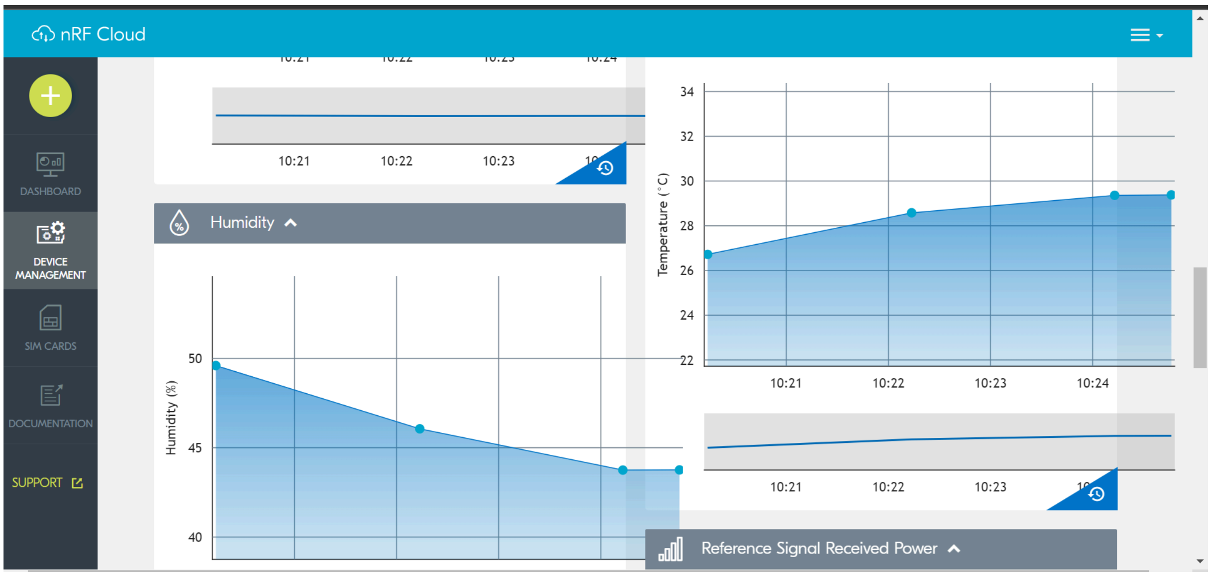
Showing 1 Sim

iBasis - 893108062005426517
Not connected to any device
Data Remaining 10.46 / 10.49 Mb

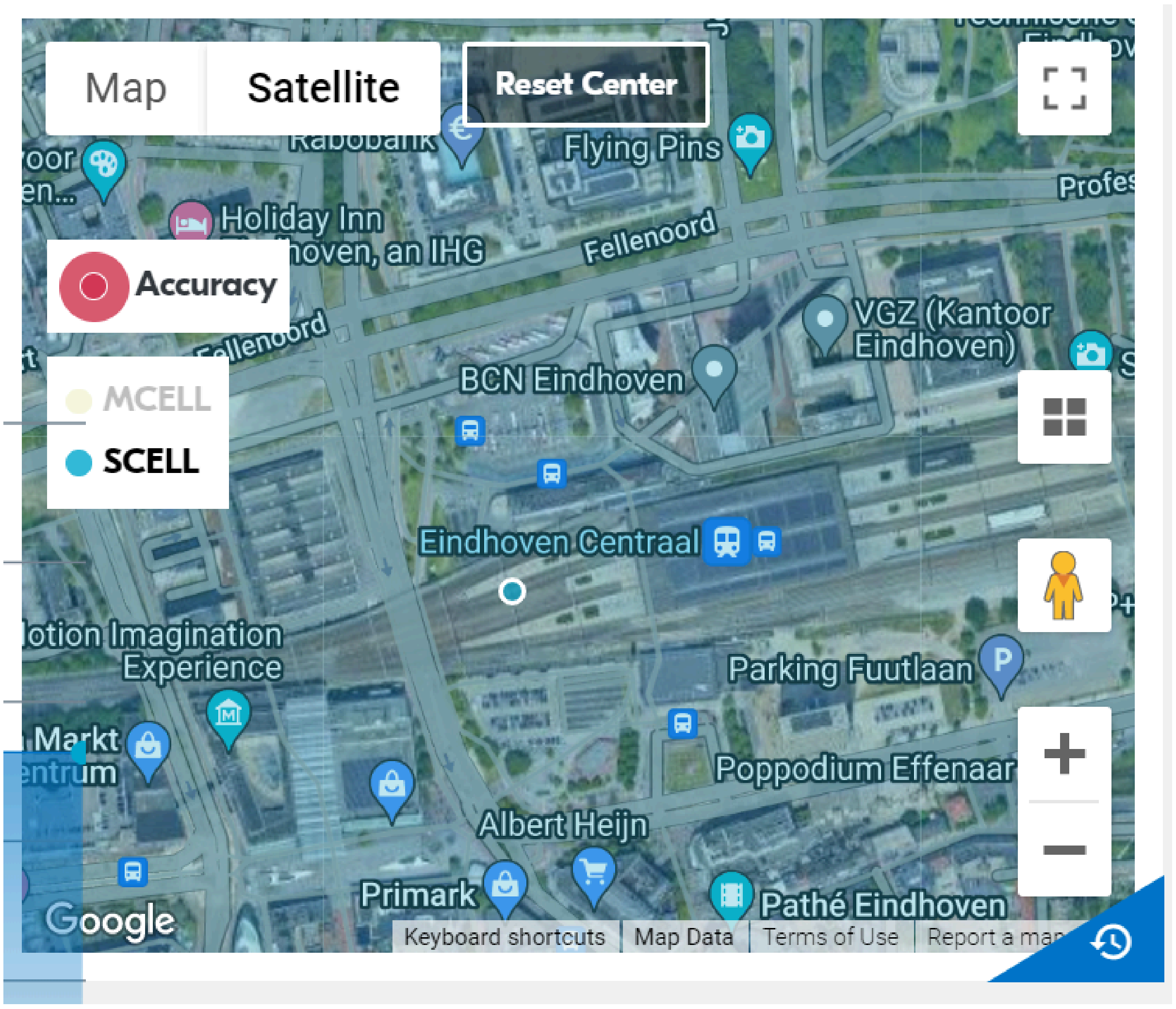
Furthermore, we started receiving values to the dashboard from all of the sensors. Moreover, we were able to track the GNSS from thingy 91 and then compared with our live location with the help of Google maps. All the results can be shown in the following images.



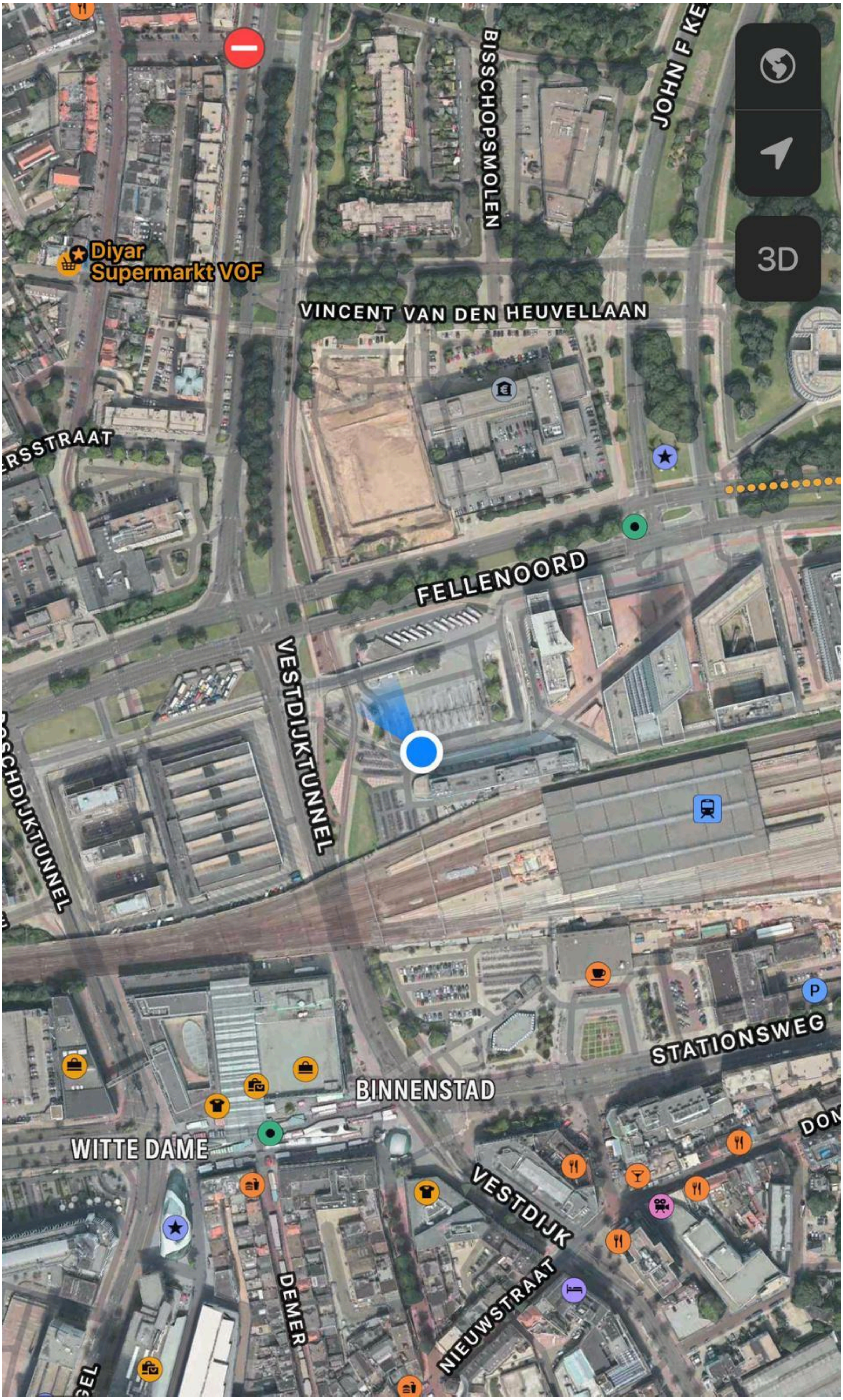
The above image shows the air pressure sensor values and the thingy 91 location in a real time basis.



In the above image we can see the dashboard on the nrf cloud and the values of humidity and temperature.

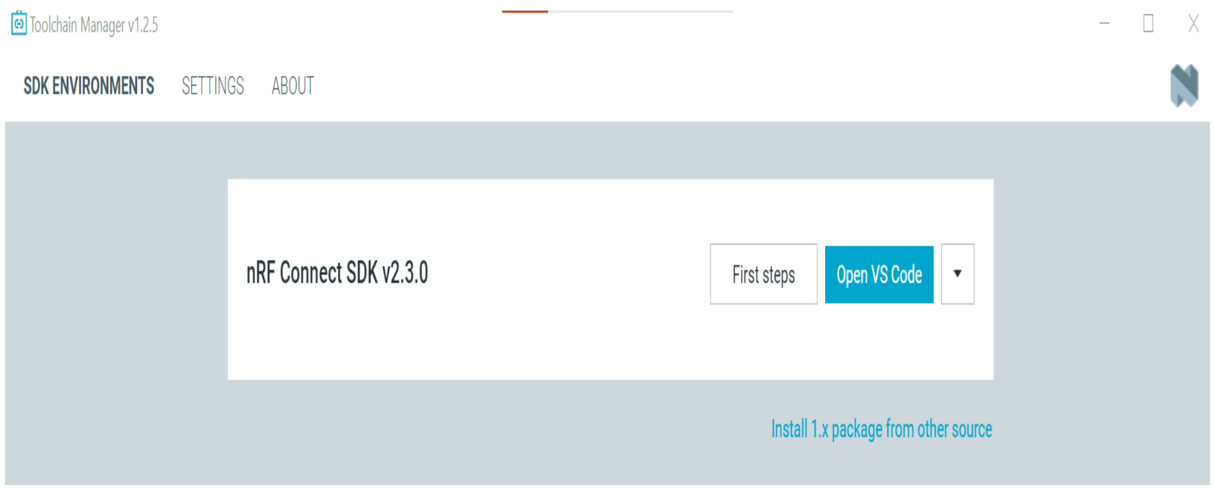


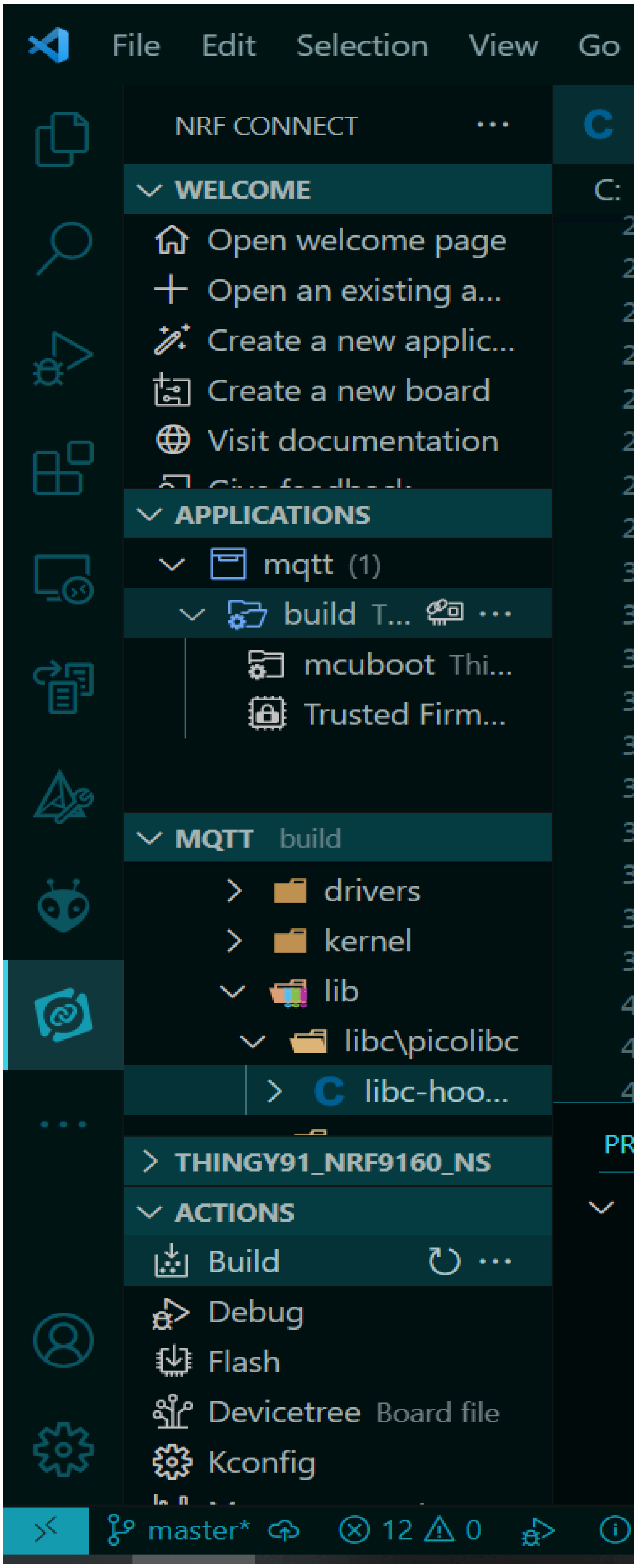
In the above image we can see the real time location of thingy 91 and in the picture below we can see the location of our team mate who is holding thingy 91 with google maps on an ios device.


3D
BISSCHOPSMOLEN
JOHN F KE
Diyar Supermarkt VOF
VINCENT VAN DEN HEUVELLAAN
ERSSTRAAT
FELLENOORD
VESTDIJKTUNNEL
BOSCHDIJKTUNNEL
STATIONSWEG
BINNENSTAD
WITTE DAME
VESTDIJK
NIEUWSTRAAT
DEMER
DOM
GEL

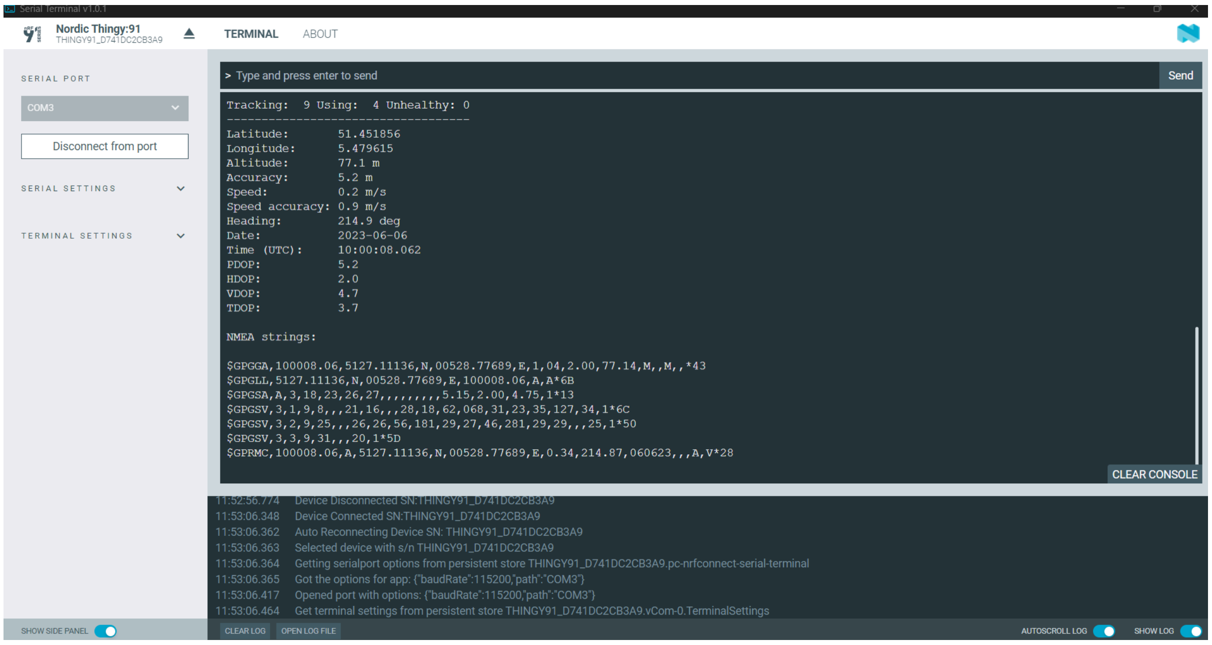
Sprint 2

During the second sprint, our main focus was to establish a seamless connection between VS Code and the nRF Connect environment. This enabled us to start coding and explore the functionality of the accelerometer and GNSS features on the Thingy91. We successfully accessed and retrieved data from these sensors and displayed it on the serial monitor for analysis. To provide a visual representation of our progress, I have included images showcasing the process of enabling nRF Connect in VS Code and the associated extension within the nRF Connect environment.




File
Edit
Selection
View
Go
NRF CONNECT
WELCOME
Open welcome page
Open an existing a...
Create a new applic...
Create a new board
Visit documentation
APPLICATIONS
mqtt (1)
build T...
mcuboot Thi...
Trusted Firm...
MQTT build
drivers
kernel
lib
libc\picolibc
libc-hoo...
THINGY91_NRF9160_NS
ACTIONS
Build
Debug
Flash
Devicetree Board file
Kconfig
master*
12
0

During this phase, I successfully implemented GNSS functionality by leveraging a sample code and making necessary modifications based on the microcontroller version. I utilised the VS Code environment, which was already connected with the nRF Connect platform. In the initial testing, I encountered a challenge where the signals required for calculating longitude and latitude were not being received. However, I was able to receive tracking values. To troubleshoot this issue, I decided to step outside the R10 building and remained there for approximately 5 minutes. As a result, I started receiving accurate location values. To monitor the data sent by the Thingy 91 to the serial interface, I used the Nordic Serial Monitor. The successful results of the location tracking can be observed in the attached image.

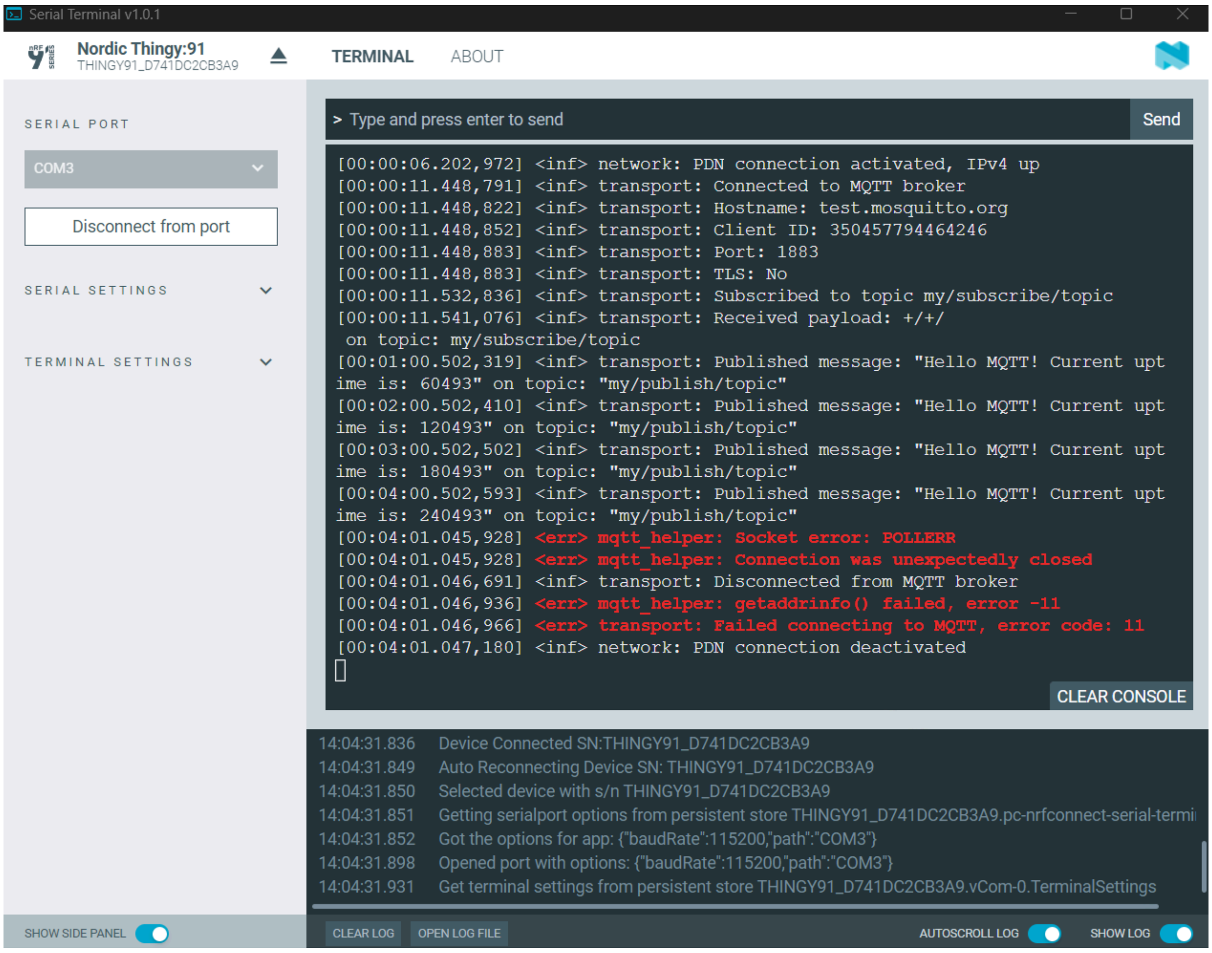


```
Tracking:  9 Using:  4 Unhealthy: 0
-----------------------------------
Latitude:       51.451856
Longitude:      5.479615
Altitude:       77.1 m
Accuracy:       5.2 m
Speed:          0.2 m/s
Speed accuracy: 0.9 m/s
Heading:        214.9 deg
Date:           2023-06-06
Time (UTC):     10:00:08.062
PDOP:           5.2
HDOP:           2.0
VDOP:           4.7
TDOP:           3.7

NMEA strings:

$GPGGA,100008.06,5127.11136,N,00528.77689,E,1,04,2.00,77.14,M,,M,,*43
$GPGLL,5127.11136,N,00528.77689,E,100008.06,A,A*6B
$GPGSA,A,3,18,23,26,27,,,,,,,,,5.15,2.00,4.75,1*13
$GPGSV,3,1,9,8,,,21,16,,,28,18,62,068,31,23,35,127,34,1*6C
$GPGSV,3,2,9,25,,,26,26,56,181,29,27,46,281,29,29,,,25,1*50
$GPGSV,3,3,9,31,,,20,1*5D
$GPRMC,100008.06,A,5127.11136,N,00528.77689,E,0.34,214.87,060623,,,A,V*28
```

```
11:52:56.774   Device Disconnected SN:THINGY91_D741DC2CB3A9
11:53:06.348   Device Connected SN:THINGY91_D741DC2CB3A9
11:53:06.362   Auto Reconnecting Device SN: THINGY91_D741DC2CB3A9
11:53:06.363   Selected device with s/n THINGY91_D741DC2CB3A9
11:53:06.364   Getting serialport options from persistent store THINGY91_D741DC2CB3A9.pc-nrfconnect-serial-terminal
11:53:06.365   Got the options for app: {"baudRate":115200,"path":"COM3"}
11:53:06.417   Opened port with options: {"baudRate":115200,"path":"COM3"}
11:53:06.464   Get terminal settings from persistent store THINGY91_D741DC2CB3A9.vCom-0.TerminalSettings
```

After completing the initial tasks, our focus shifted towards the crucial IoT aspect of the project, which involved connecting the Thingy91 device to an MQTT server or a dashboard for data transmission. Through thorough research and exploration, I successfully established the connection between Thingy91 and MQTT. Initially, I encountered some challenges with the libraries used in the sample code. However, after dedicating an entire day to troubleshooting, I was able to resolve the issues by replacing the outdated libraries with newer, built-in or package libraries.

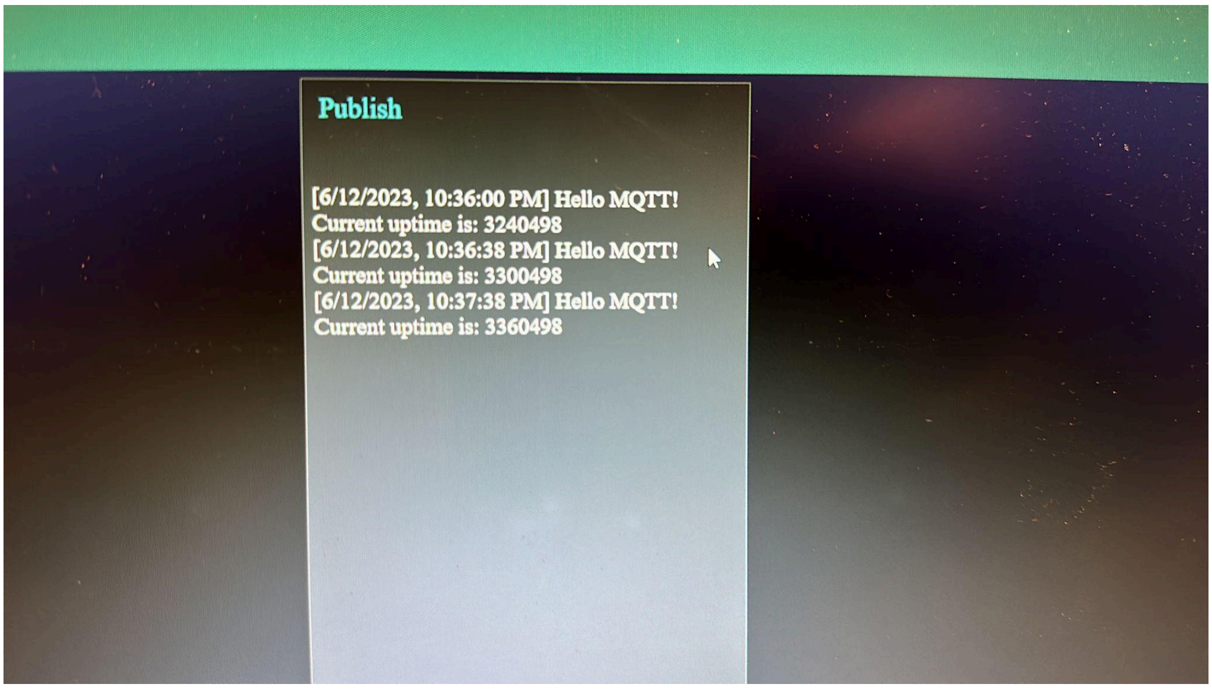
Once the library issues were resolved, I configured the necessary parameters such as MQTT topic and publishing delay and verified the successful transmission of data to the MQTT server. This progress was evident through the data being printed on the serial monitor. Furthermore, I linked the MQTT topic with the Thingy91 device and successfully received messages from the device, indicating the successful integration. Detailed visuals showcasing these achievements are included in the images below.



```
[00:00:06.202,972] <inf> network: PDN connection activated, IPv4 up
[00:00:11.448,791] <inf> transport: Connected to MQTT broker
[00:00:11.448,822] <inf> transport: Hostname: test.mosquitto.org
[00:00:11.448,852] <inf> transport: Client ID: 350457794464246
[00:00:11.448,883] <inf> transport: Port: 1883
[00:00:11.448,883] <inf> transport: TLS: No
[00:00:11.532,836] <inf> transport: Subscribed to topic my/subscribe/topic
[00:00:11.541,076] <inf> transport: Received payload: +/+/
 on topic: my/subscribe/topic
[00:01:00.502,319] <inf> transport: Published message: "Hello MQTT! Current upt
ime is: 60493" on topic: "my/publish/topic"
[00:02:00.502,410] <inf> transport: Published message: "Hello MQTT! Current upt
ime is: 120493" on topic: "my/publish/topic"
[00:03:00.502,502] <inf> transport: Published message: "Hello MQTT! Current upt
ime is: 180493" on topic: "my/publish/topic"
[00:04:00.502,593] <inf> transport: Published message: "Hello MQTT! Current upt
ime is: 240493" on topic: "my/publish/topic"
[00:04:01.045,928] <err> mqtt_helper: Socket error: POLLERR
[00:04:01.045,928] <err> mqtt_helper: Connection was unexpectedly closed
[00:04:01.046,691] <inf> transport: Disconnected from MQTT broker
[00:04:01.046,936] <err> mqtt_helper: getaddrinfo() failed, error -11
[00:04:01.046,966] <err> transport: Failed connecting to MQTT, error code: 11
[00:04:01.047,180] <inf> network: PDN connection deactivated
```

```
14:04:31.836   Device Connected SN:THINGY91_D741DC2CB3A9
14:04:31.849   Auto Reconnecting Device SN: THINGY91_D741DC2CB3A9
14:04:31.850   Selected device with s/n THINGY91_D741DC2CB3A9
14:04:31.851   Getting serialport options from persistent store THINGY91_D741DC2CB3A9.pc-nrfconnect-serial-termi
14:04:31.852   Got the options for app: {"baudRate":115200,"path":"COM3"}
14:04:31.898   Opened port with options: {"baudRate":115200,"path":"COM3"}
14:04:31.931   Get terminal settings from persistent store THINGY91_D741DC2CB3A9.vCom-0.TerminalSettings
```

Mqtt Dashboard screenshot:



[6/12/2023, 10:36:00 PM] Hello MQTT! Current uptime is: 3240498
[6/12/2023, 10:36:38 PM] Hello MQTT! Current uptime is: 3300498
[6/12/2023, 10:37:38 PM] Hello MQTT! Current uptime is: 3360498

Conclusion

Throughout my research on the Thingy 91 micro-controller, I have delved into its capabilities and explored various aspects of its functionality. I successfully updated the firmware of the device, ensuring that it is equipped with the latest enhancements and features. Moreover, in my research on the Thingy 91, GNSS integration, and data visualisation through nRF Cloud, has been a fulfilling journey. The knowledge and experiences gained through this exploration have enhanced my proficiency in the IOT field with the thingy 91 mini project research.

Despite facing challenges with library compatibility specific to the Thingy91 microcontroller, I successfully implemented GNSS functionality and established message sending to an MQTT server. Through extensive research and troubleshooting, I resolved the library issues by finding alternative libraries and adapting the code. This achievement showcased my problem-solving skills and provided valuable experience in working with emerging technologies. The project taught me to adapt to new technologies, think critically, and persevere in complex situations. Overall, this experience strengthened my passion for embedded systems and my ability to overcome technical obstacles.

References

- nRF cloud dashboard
  # https://nrfcloud.com/#/devices/nrf-350457794464246
- nRF desktop application/software
  # https://nrfcloud.com/#/devices/nrf-350457794464246
- Firmware files
  #
  https://www.nordicsemi.com/Products/Development-hardware/Nordic-Thingy-91/Download