IoT Using LoRaWAN

CSE3PR Industry Project

Team F – User Manual

James Curnow

Joel Morran

Michael Owczarek

Remi Petit

Luis Shigetomi

Table of Contents

Table of Diagrams

1.0 – Introduction

1.1 – Audience Description

This document is for the two audience groups of this project:

Primary: The IoT using LoRaWAN project is intended for use by staff at the LaTrobe University Centre for Technology Infusion.

Secondary: The system has also been developed with the hope that it can be marketed to and used by various people with specific needs. For example, farmers could use the system in order to track their cattle's behavior and movement patterns

For both user types, basic computer skills are required. Knowledge of how to filter data within Power BI is beneficial but not required.

Furthermore, there are two types of roles:

- **User:** Concerned with viewing and filtering the collected data in Power BI.
- **Admin:** Concerned with setting up and maintaining the system.

1.2 – Applicability Statement

In terms of hardware, the system runs on Raspberry Pi and Arduino sensors.

In terms of software, the system is managed entirely through Microsoft's Azure.

The system is accessed by the user using any web browser that supports Microsoft's Power BI. The user can also access the Azure IoT Hub online should they ever want to manage their devices.

1.3 – Purpose Statement

The purpose of the IoT using LoRaWAN project is to:

- Create a visualization environment for sensor activity
- Provide an environment capable of handling additional sensors
- Provide separate restricted views of data for users based on their role in the system
- Display sensor specific data in an understandable format

This project's aim is to implement a LoRaWAN sensor network with a few sensors to generate data and present/visualise it for a user.

1.4 – Document Usage Description

1 – Introduction:

Provides a short description outlining an overview of the system, it also discusses the audience and provides purpose of the usage of the document. The conventions used on the document and changelog are also included.

2 – Introductory Kit:

Provides a summary of the system's features as well as a guide for getting started. This includes a list of all required hardware and software.

3 – User Manual:

Provides user with how to access the Power BI interface as well as a guide for basic data filtering.

4 – Installation Guide:

Provides Admins with a complete guide on how to setup everything required to run the IoT Using LoRaWAN system.

The guide covers the setup of the following:

- Azure
- Gateway
- Sensor(s)
- Power BI

5 – System Administrator Guide:

Provides Admins with a guide on how to monitor the system, add & remove devices, clear the gateway's cache. Information on further customisation is also provided.

6 – Appendix:

Provides a glossary, bibliography, statement of authorship and statement of effort

1.5 – Conventions

The following conventions are used throughout the document:

- A consistent styling theme is kept throughout the document to maintain readability.
- Numbered headings are included to allow easy navigation.
- Each image in this document is numbered and titled

2.0 – Introductory Kit

2.1 – Summary of System Features

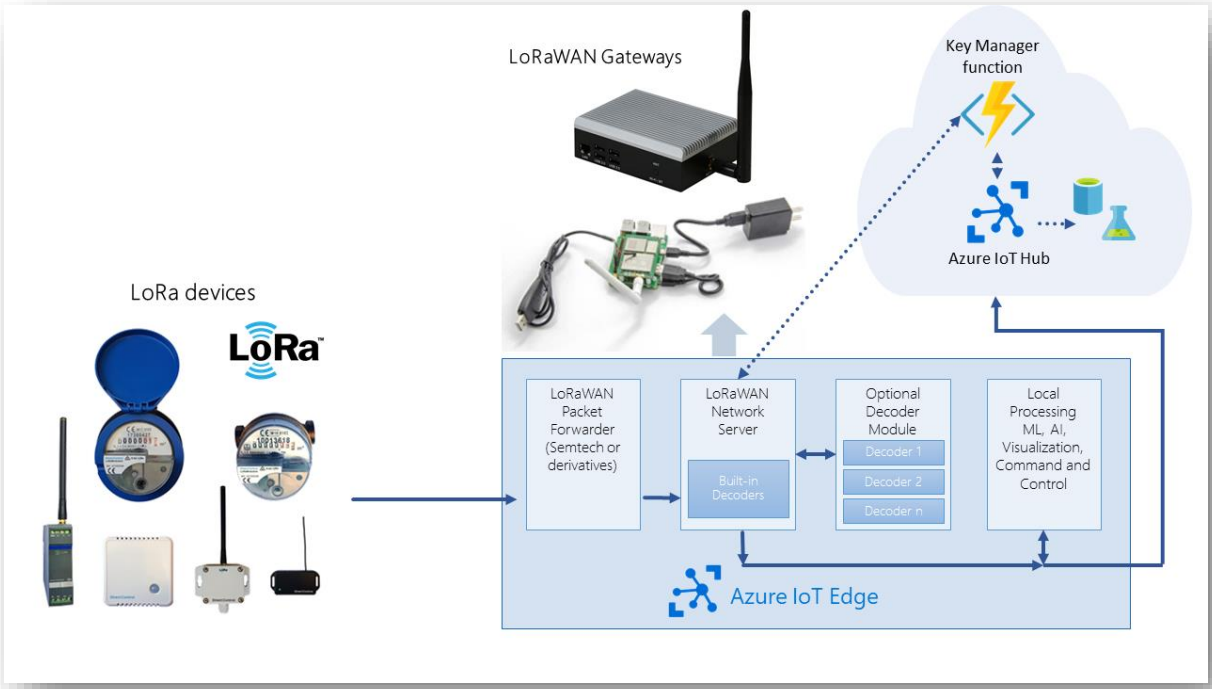

Figure 1 - System Architecture

- A group of sensors connected through LoRaWAN to a gateway (using LoRaWAN 1.0.2 implementation).
- A connection through the gateway directly to the IoT platform for sending and receiving the collected data.
- An IoT platform set up for managing the data to and from the sensors.
- A data processing algorithm to extract features and detect certain patterns in the logged data.
- A visualisation environment to illustrate the sensors activities.
- Device and Gateway management done completely through Azure IoT Hub.
- Bi-directional communication between LoRa end devices and Azure cloud.
- Custom packet decoding framework.
- Easy deployment and setup using Azure ARM template.

2.2 – Getting Started Guide

Please ensure you have the following hardware and software before attempting to setup the IoT Using LoRaWAN system:

2.2.3 – Required Software

The following software is required to setup and use the IoT Using LoRaWAN system:

User:

- Power BI desktop app (https://powerbi.microsoft.com/en-us/desktop/)

Admin:

- Visual Studio Code (https://code.visualstudio.com/)
- Docker (https://www.docker.com/)
- SSMS (https://docs.microsoft.com/en-us/sql/ssms/download-sql-server-management-studio-ssms)
- balenaEtcher (https://www.balena.io/etcher/)

2.2.1 – Required Hardware

The following hardware is required to setup and use the IoT Using LoRaWAN system:

- Seeed Studio LoRaWAN Gateway Kit
- Grove – IMU 9DOF v2.0
- A PC capable of running Power BI
- A PC capable of interfacing with a Raspberry Pi via SSH (alternatively, you can interface with the Pi via HDMI)

2.2.2 – Hardware Setup

2.2.2.1 – Gateway Setup

Please follow the setup guide provided by Seeed Studio:
http://wiki.seeedstudio.com/LoRa_LoRaWan_Gateway_Kit/#hardwareconnection

When complete, the gateway (Raspberry) Pi should look as follows:

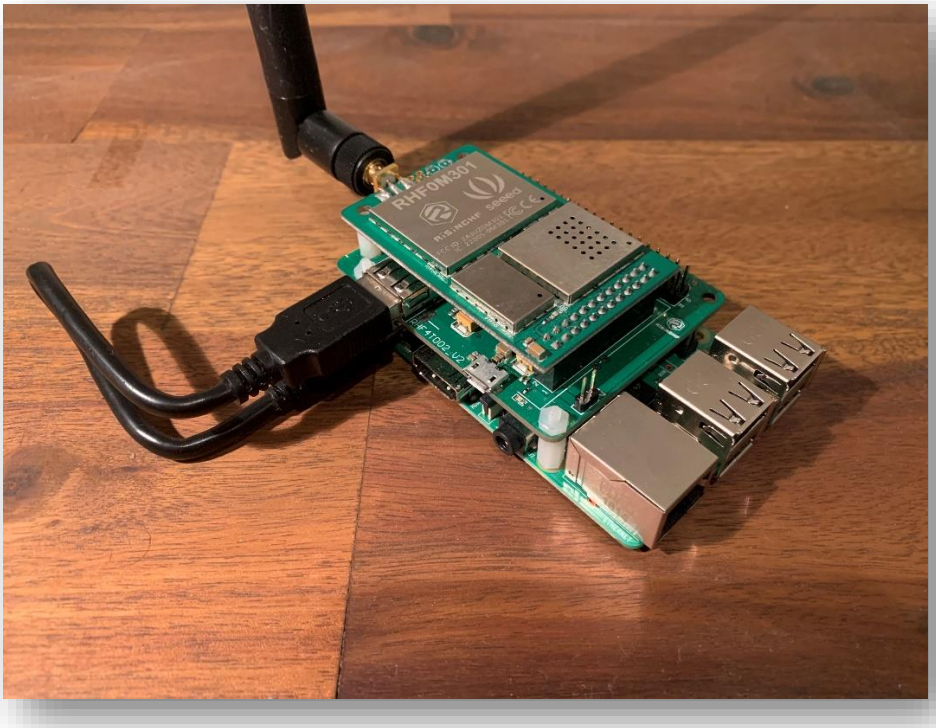

Figure 2 - Gateway Configuration

2.2.2.2 – Sensor Setup

Simply connect the Grove – IMU 9DOF v2.0 sensor to the Seeeduino LoRaWAN W/ GPS via I2C.

When complete, the sensor should look as follows:

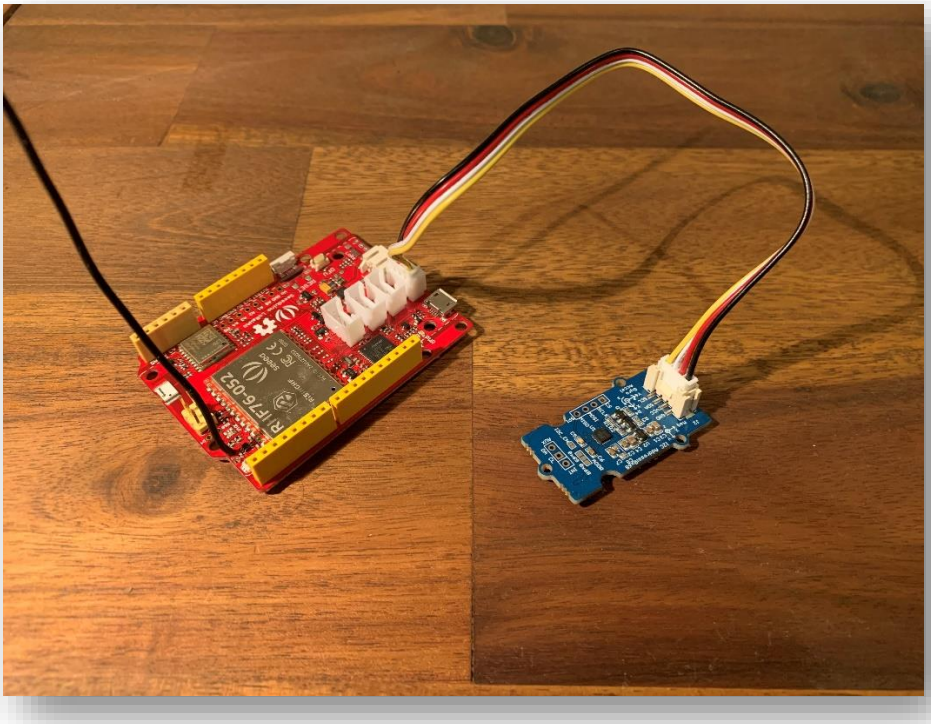

Figure 3 - Sensor Configuration

3.0 – User Manual

3.1 – Starting the Gateway

To start the IoT gateway, simply plug the Raspberry Pi into s power supply. Please allow up to 10min for the IoT Edge Daemon to start up.

NOTE: The best position for the gateway is outside or inside near a window. This is to allow the gateway to easily pickup any incoming sensor payloads.

NOTE: The gateway needs a strong stable internet connection as it will constantly be interacting with Azure and the database

3.2 – Starting a Sensor

Simply plug the desired sensor into a power supply. It will automatically start sending LoRa payloads once it receives a valid GPS signal.

NOTE: The GPS works best when outside and away from any obstructions. Using the GPS inside can result in data inconsistencies.

3.3 – Using Power BI

This is the view you are presented with when opening the Power BI template file:

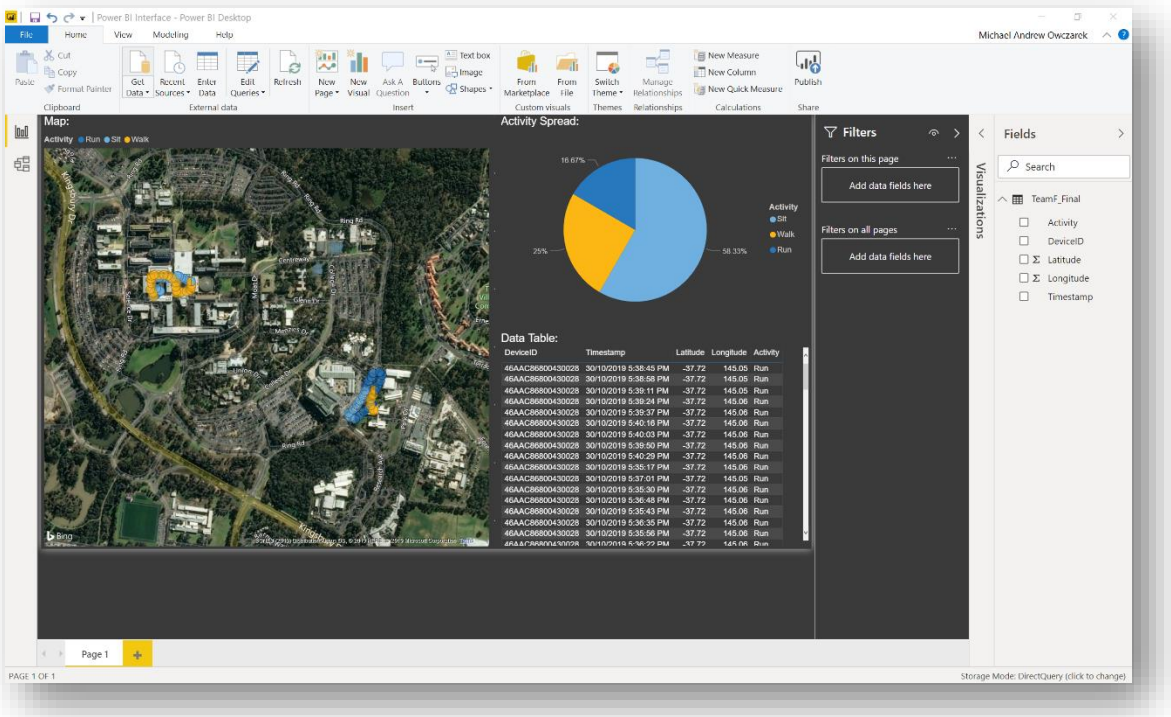

Figure 4 - Power BI Initial View

3.3.1 – Map

The map shows all data collected as pinpoints. You can mouse over any tooltip to see it's information. This includes latitude, longitude, activity, timestamp and the device ID:

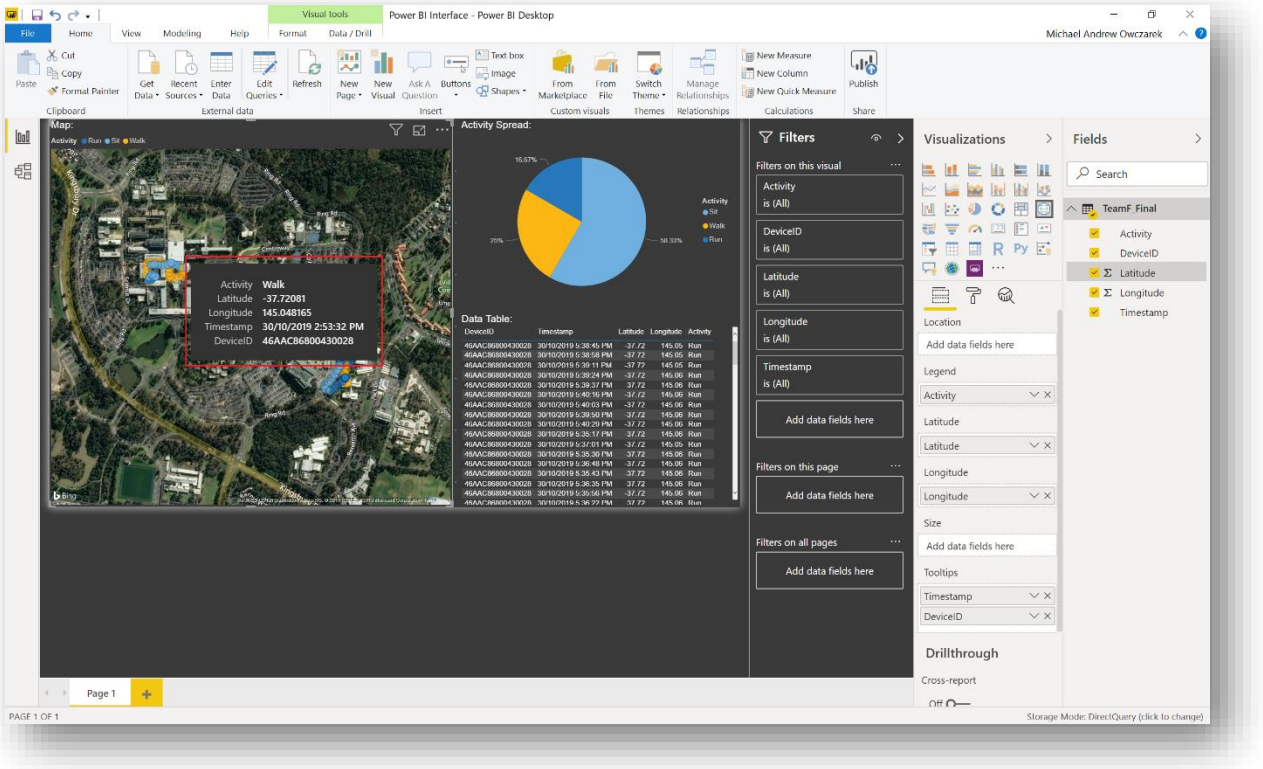

Figure 5 - Power BI Map Functionality

3.3.2 – Activity Spread Pie Chart

The pie chart shows the distribution of each of the activities. Mousing over a section will show the total count of a given activity:

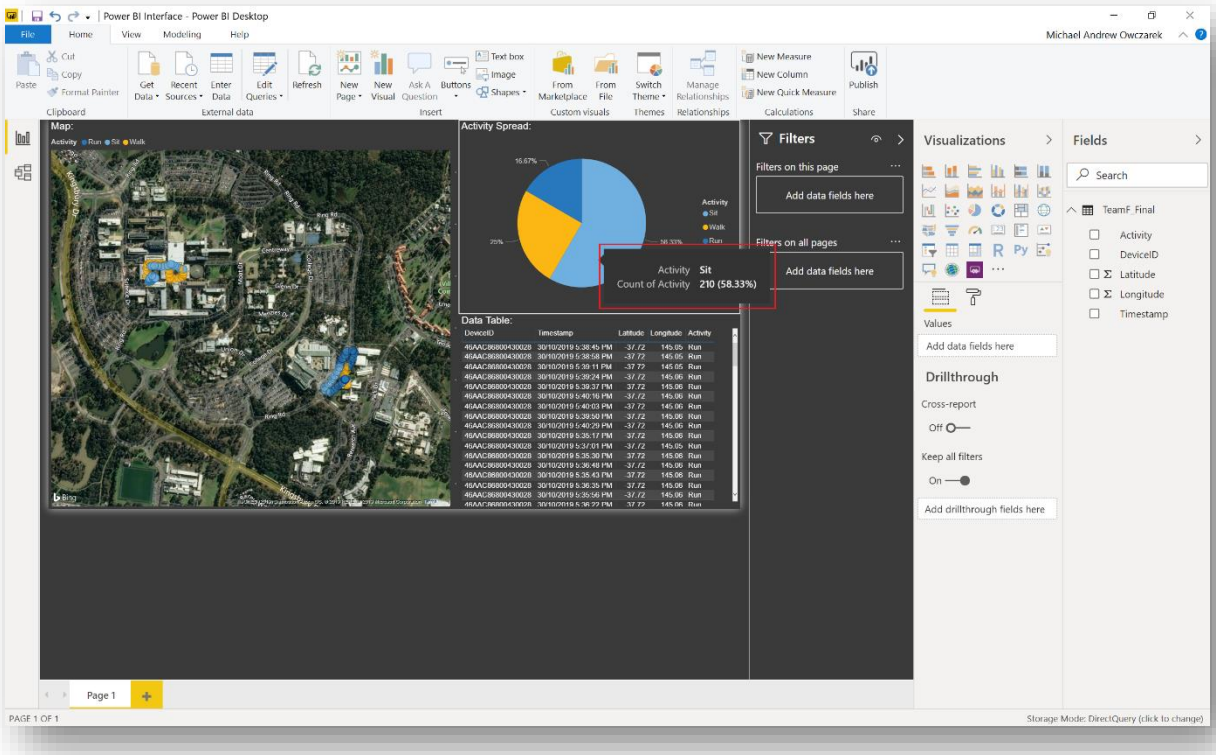

Figure 6 - Power BI Pie Chart Functionality

3.3.3 – Data Table

The table shows all data in a simple table view. Selecting a value will highlight that value in the map view

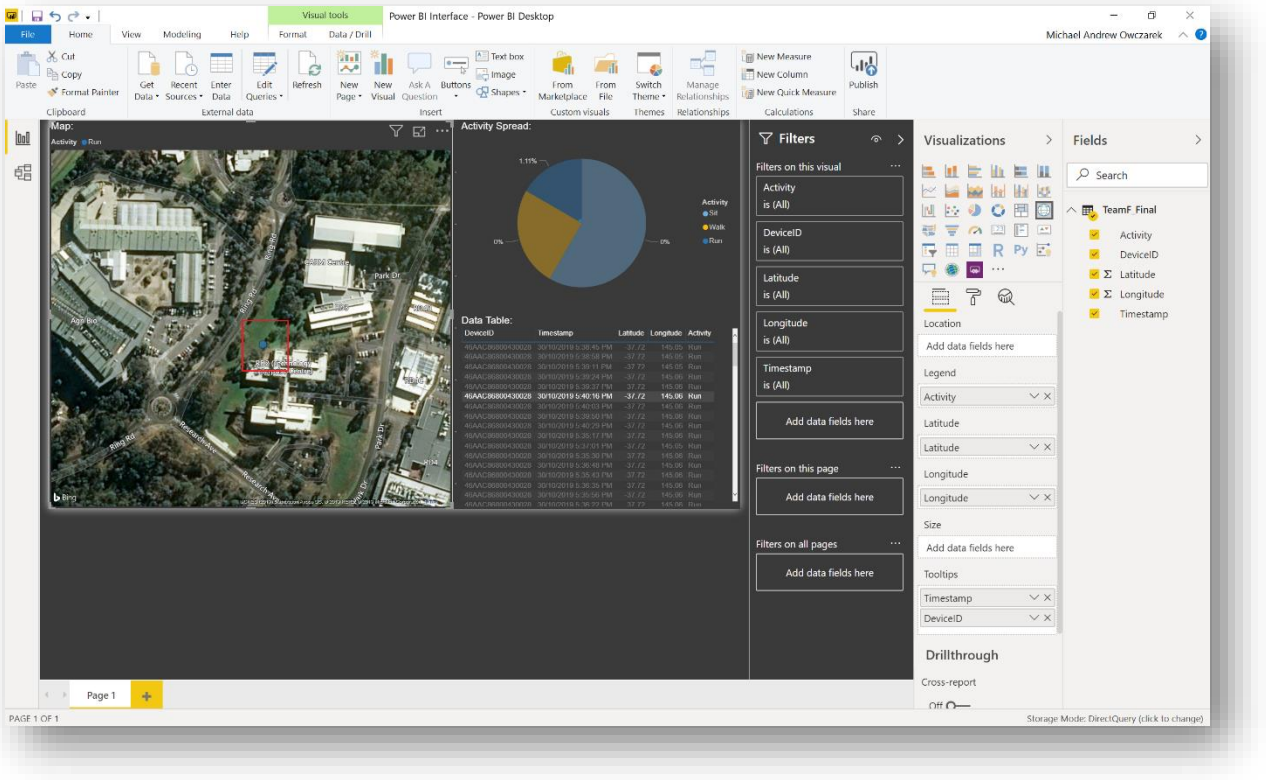

Figure 7 - Power BI Data Table Functionality

3.3.4 – Filtering Data

To filter the report by a field, drag the desired field into the **'Filters on this page section'**:

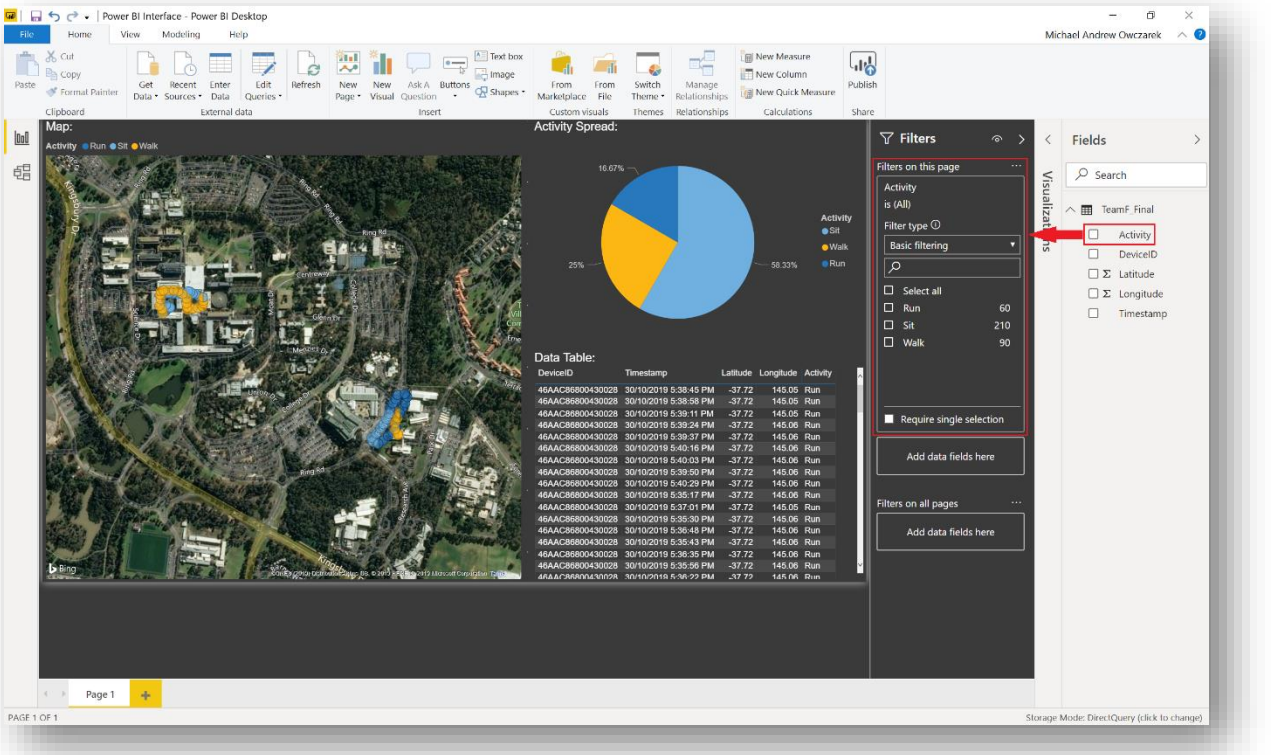

Figure 8 - Power BI Filtering 1

From there you can select which value you want to filter by:

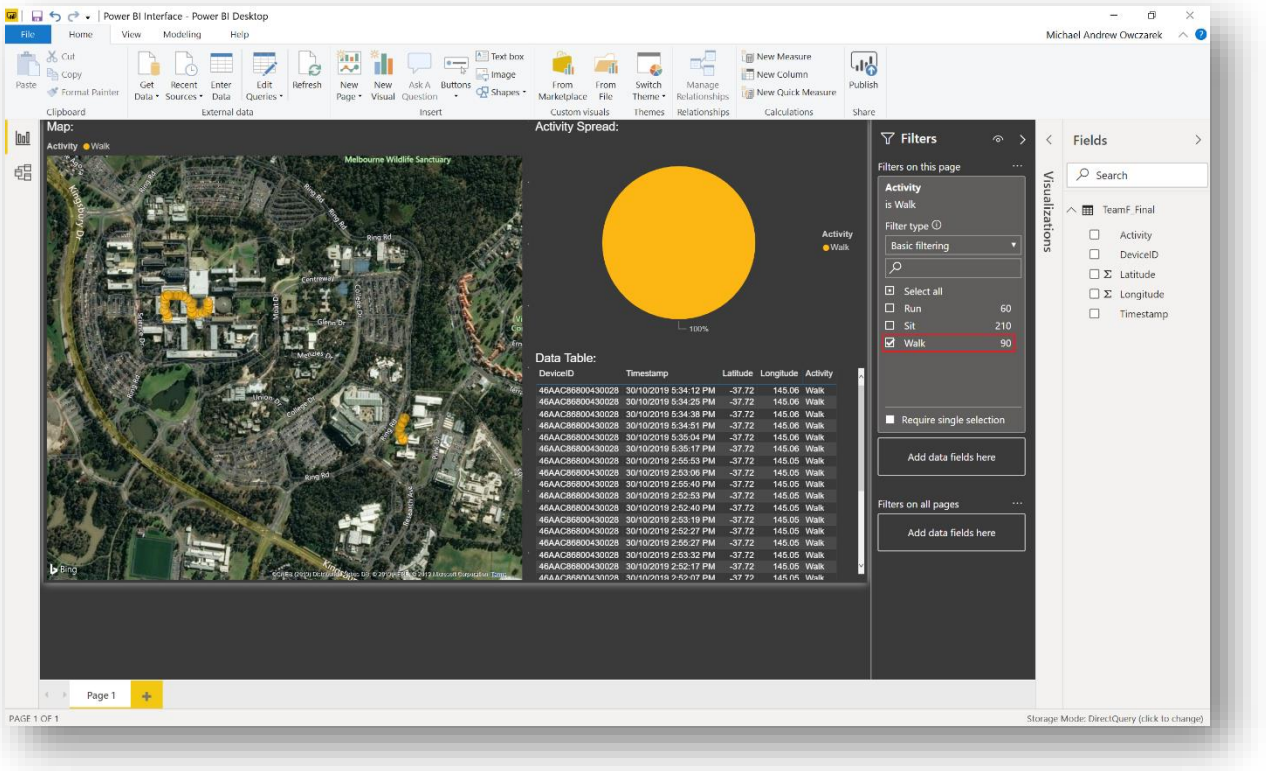

Figure 9 - Power BI Filtering 2

3.3.5 – Additional Information

For a more detailed guide on using Power BI, please refer to the following webpage: https://docs.microsoft.com/en-us/power-bi/

4.0 – Setup/Installation Guide

4.1 – Azure Setup

4.1.1 – Resource Group Setup

1. Navigate to: https://portal.azure.com/#create/Microsoft.ResourceGroup
2. Enter in any name for **'Resource group'**
3. Set **'Region'** to a region of your choice. (**'(Asia Pacific) Australia East'** is recommended)
4. Select **'Review + Create'**
5. After verifying the details, select **'Create'** and wait for Azure to confirm setup of the resource group

4.1.2 – Template Deployment

1. Navigate to: https://portal.azure.com/#create/Microsoft.Template/uri/https%3A%2F%2Fraw.githubusercontent.com%2FAzure%2Fiotedge-lorawan-starterkit%2Fmaster%2FTemplate%2Fazuredeploy.json
2. Set **'Resource group'** to the same resource group that was created in
3. Set **'Location'** to a location of your choice. (**'(Asia Pacific) Australia East'** is recommended)
4. Set **'Unique Solution Prefix'** to a prefix of your choice
5. Set **'Edge Gateway Name'** to a name of your choice
6. Set **'Deploy Device'** to **'true'**
7. Set **'Reset Pin'** to **'7'**
8. Set **'Region'** to **'US'**
9. Set **'Spi Speed'** to **'2'**
10. Set **'Spi Dev'** to **'2'**
11. Select **'Purchase'** and wait for Azure to finish the deployment
    NOTE: This process can take up to 10min to complete

4.1.3 – Decoder Deployment

NOTE: Ensure you have the Docker extension for Visual Studio Code and you can access your Azure registry. Use **'docker login < your container registry address >.azurecr.io'** to login

1. Open the Decoder folder located in the source code folder with Visual Studio Code
2. Edit the **'module.json'** file to include **'<your container registry address>/<Image Name>'** as the repository
3. Right click **'module.json'** in the explorer and select **'Build and Push IoT Edge Module Image'**
4. Select **'arm32v7'**
5. Once deployed, navigate to your IoT Hub in your Azure Portal

6. If required, add credentials to access your container registry to the IoT Edge device by adding them to your hub's module container registry settings
7. Select **'IoT Edge'**
8. Select your gateway device
9. Select **'Set Modules'**
10. Select **'Add'** under **'Deployment Modules'**
11. Select **'IoT Edge Module'**
12. Set **'Name'** to any lowercase name
13. Set **'Image URI'** to **'<your container registry address>/<Image Name>**
14. Navigate back to your IoT Hub
15. Select **'IoT Devices'**
16. Select **'Device Details'**
17. Select **'Device Twin'**
18. Set **'SensorDecoder'** to **'http://<decoder module name>/api/DecoderTest'**

4.2 – Gateway (Raspberry Pi) Setup

4.2.1 – Raspbian Installation

1. Download the following Raspbian image: https://downloads.raspberrypi.org/raspbian/images/raspbian-2018-11-15/
2. Run balenaEtcher and select the Raspbian image that you extracted in step 1
3. Select the microSD card drive. Etcher may have already selected the correct drive
4. Click **'Flash'** to install Raspbian to the microSD card
5. Remove the microSD card from your computer when installation is complete and insert the microSD card into the Pi

4.2.2 – Installing the Container Runtime

The following commands install both the Moby-based engine and command-line interface (CLI). The CLI is useful for development but optional for production deployments.

Paste and run the follow commands (without the single quotes) one at a time into your Raspberry Pi terminal:

1. **'curl -L https://aka.ms/moby-engine-armhf-latest -o moby_engine.deb && sudo dpkg -i ./moby_engine.deb'**
2. **'curl -L https://aka.ms/moby-cli-armhf-latest -o moby_cli.deb && sudo dpkg -i ./moby_cli.deb'**
3. **'sudo apt-get install –f'**

4.2.3 – Installing the IoT Edge Security Daemon

Paste and run the follow commands (without the single quotes) one at a time into your Raspberry Pi terminal:

1. **'curl -L https://aka.ms/libiothsm-std-linux-armhf-latest -o libiothsm-std.deb && sudo dpkg -i ./libiothsm-std.deb'**
2. **'curl -L https://aka.ms/iotedged-linux-armhf-latest -o iotedge.deb && sudo dpkg -i ./iotedge.deb'**
3. **'sudo apt-get install -f'**

4.2.4 – Connecting to your IoT Hub

To manually provision a device, you need to provide it with a device connection string that you can create by registering a new IoT Edge device in your IoT hub.

1. Open the configuration file in terminal using the following command: **'sudo nano /etc/iotedge/config.yaml'**
2. The text document should look as follows**:**

```
yaml

provisioning:

source: "manual"

device_connection_string: "<ADD DEVICE CONNECTION STRING HERE>"

# provisioning:

#   source: "dps"

#   global_endpoint: "https://global.azure-devices-provisioning.net"

#   scope_id: "{scope_id}"

          #   registration_id: "{registration_id}"
```

3. Save and close the file using the following command: **'CTRL + X', 'Y', 'Enter'**
4. After entering the provisioning information in the configuration file, restart the daemon using the following command: **'sudo systemctl restart iotedge'**

4.2.5 – Verifying Successful Installation (Optional)

You can check the status of the IoT Edge Daemon by entering the following commands in terminal:

1. '**systemctl status iotedge'**
2. **'journalctl -u iotedge --no-pager --no-full'**
3. **'sudo iotedge list'**

4.2.6 – Enabling Interfacing Options

Enable VNC, SPI, Serial, SSH and I2C:

1. Click the '**Raspberry icon'** > '**Preferences'** > '**Raspberry Pi Configuration**'
2. On the Interfaces tab, set **'VNC'**, **'SPI'**, **'Serial'**, **'SSH'** and **'I2C'** to Enabled, then click **'OK'**
3. Reboot the Raspberry Pi

4.2.7 – Python Machine Learning Deployment

1. Run the following commands to install Python 3 and all required packages:

```
sudo pip install python3

sudo apt-get install python3-pandas
sudo apt-get install python3-seaborn
sudo apt-get install python3-matplotlib
sudo apt-get install python3-sklearn

sudo apt-get install python3-numpy
sudo apt-get install libblas-dev
sudo apt-get install liblapack-dev
sudo apt-get install python3-dev
sudo apt-get install libatlas-base-dev
sudo apt-get install gfortran
sudo apt-get install python3-setuptools
sudo apt-get install python3-scipy
sudo apt-get update
sudo apt-get install python3-h5py

sudo pip3 install keras
sudo pip3 install tensorflow
```

2. Copy the **'Python ML'** folder from the source code folder to the 'pi' directory

NOTE: It is best to restart the Pi to ensure everything is installed correctly

4.3 – Sensor (Arduino) Setup

1. Open Arduino IDE, select **'File' > 'Preferences'**
2. Paste **'https://raw.githubusercontent.com/Seeed-Studio/Seeed_Platform/master/package_seeeduino_boards_index.json'** into **'Additional Boards Manger URLs'**
3. Select **'Tools' > 'Board' > 'Board Manager'**
4. Search for and install **'Seeed SAMD Boards'**
5. Select **'Tools' > 'Board' > 'Seeeduino LoRaWAN'**
6. If not done already, connect the Arduino board to the PC

NOTE: The driver for the Seeeduino should automatically install. If it doesn't use the driver provided here:
https://github.com/SeeedDocument/Seeeduino_LoRa/raw/master/res/driver.zip

7. Select **'Tools' > 'Port'** and select the Seeeduino COM port
8. Select **'Open'** and select **'Arduino.ino'** from the source code folder
9. Select **'Upload'** and wait for the code to be uploaded onto the Seeeduino
   NOTE: Uploading to the Seeeduino is known to be temperamental, so if it doesn't work on the first try, just try again
10. The Seeeduino is now setup and will function whenever powered via battery or USB

4.4 – Power BI Setup

1. Open the Power BI desktop app
2. Select 'Get data' in the bottom left corner
3. Load the **'Power BI Interface.pbix'** file from the source code folder

5.0 – System Administration Guide

5.1 – Monitoring the System

5.1.1 – Monitoring Gateway Modules

To view the status of all IoT Edge modules, perform the following:

1. Either connect to the gateway via SSH or HDMI/mouse and keyboard
2. Enter the following command: **'sudo iotedge list'**

Alternatively, you can view the logs of an individual module:

1. Either connect to the gateway via SSH or HDMI/mouse and keyboard
2. Enter the following command: **'sudo iotedge logs <ModuleName>'** where <ModuleName> is the name of the module you'd like to view

5.1.2 – Monitoring Gateway Endpoint

1. Open Visual Studio Code
2. Connect to your Azure IoT Hub using your gateway's connection string
3. Right click the Raspberry Pi device and select **'Start Monitoring Built-in Event Endpoint'**
4. The output section will now display messages whenever a payload is received and processed by the decoder

5.1.3 – Monitoring Sensor Output

To monitor a sensors output, simply connect to the sensor via its serial port.
The sensor outputs to the serial port with the following format:

'**<DeviceID>, <DateTime>,<X>,<Y>,<Z>,<Lat>,<Long>'**
==========

Where:

- <DeviceID> = ID of sensor
- <DateTime> = Date pulled from GPS
- <X> = X axis value pulled from Accelerometer
- <Y> = Y axis value pulled from Accelerometer
- <Z> = Z axis value pulled from Accelerometer
- <Lat> = Latitude pulled from GPS
- <Long> = Longitude pulled from GPS

5.2 – Adding a Sensor

1. On your Azure Portal, navigate to your provisioned IoT Hub
2. Select **'IoT Devices'**
3. Select **'Add'**
4. Use the Device EUI as DeviceID
5. Select **'Save'**
6. Under the device you just added, select **'Device Twin'**
7. Paste the following into the text box:

```
"desired": {
   "AppSKey": "<Device AppSKey>",
   "NwkSKey": "<Device NwkSKey>",
   "DevAddr": "<Device Addr>",
   "SensorDecoder": "",
   "GatewayID": ""
 },
```

   NOTE: DevAddr must be unique for every device
8. Click **'Save'**

5.3 – Removing a Sensor

1. On your Azure Portal, navigate to your provisioned IoT Hub
2. Select **'IoT Devices'**
3. Select the checkbox of the device you'd like to remove, then click **'Delete'**

5.4 – Clearing Gateway Cache

Clearing the gateway's cache is as simple as rebooting the gateway.

Alternatively, if you wish to leave the gateway running, you can perform the following:

1. Either connect to the gateway via SSH or HDMI/mouse and keyboard
2. Enter the following command: **'sudo iotedge restart LoRaWanNetworkSrvModule'**

6.0 – Appendix

6.1 – Glossary

IoT: The Internet of things is the extension of Internet connectivity into physical devices and everyday objects. Embedded with electronics, Internet connectivity, and other forms of hardware, these devices can communicate and interact with others over the Internet, and they can be remotely monitored and controlled.

LoRaWAN: LoRaWAN is a media access control (MAC) protocol for wide area networks. It is designed to allow low-powered devices to communicate with Internet-connected applications over long-range wireless connections.

SQL: SQL is a domain-specific language used in programming and designed for managing data held in a relational database management system, or for stream processing in a relational data stream management system.

Microsoft Azure: Microsoft Azure is a cloud computing service created by Microsoft for building, testing, deploying, and managing applications and services through Microsoft-managed data centers.

Raspberry Pi: The Raspberry Pi is a series of small single-board computers used to promote teaching of basic computer science in schools and in developing countries.

Arduino: Arduino is an open-source hardware and software company, project and user community that designs and manufactures single-board microcontrollers and microcontroller kits.

Seeeduino: Seeeduino V4.2 is an Arduino-compatible board, which is based on ATmga328P MCU.

Power BI: Power BI is a business analytics service by Microsoft. It aims to provide interactive visualizations and business intelligence capabilities with an interface simple enough for end users to create their own reports and dashboards.

Python: Python is an interpreted, high-level, general-purpose programming language.

6.2 – References / Bibliography

http://wiki.seeedstudio.com/LoRa_LoRaWan_Gateway_Kit

http://wiki.seeedstudio.com/Grove-IMU_9DOF_v2.0/

http://wiki.seeedstudio.com/Seeeduino_LoRAWAN/

https://docs.microsoft.com/en-us/azure/iot-edge/how-to-install-iot-edge-linux-arm

https://azure.microsoft.com/en-au/resources/samples/custom-vision-service-iot-edge-raspberry-pi/

https://thenewstack.io/tutorial-connect-and-configure-raspberry-pi-as-an-azure-iot-edge-device/

https://www.thethingsnetwork.org/docs/lorawan/

https://github.com/Azure/iotedge-lorawan-starterkit

https://downloads.raspberrypi.org/raspbian/images//raspbian-2019-04-09/

https://www.balena.io/etcher/

https://docs.microsoft.com/en-us/power-bi/

https://code.visualstudio.com/

https://www.docker.com/

https://docs.npmjs.com/cli/install

https://github.com/topics/packet-forwarder

https://docs.microsoft.com/en-us/azure/iot-hub/quickstart-send-telemetry-python

https://docs.microsoft.com/en-us/azure/iot-edge/module-composition

https://docs.microsoft.com/en-us/azure/iot-edge/module-development

https://docs.microsoft.com/en-us/azure/iot-edge/quickstart-linux

https://docs.microsoft.com/en-us/azure/iot-edge/tutorial-python-module

https://docs.microsoft.com/en-us/azure/sql-database/sql-database-connect-query-dotnet-visual-studio

6.3 – Statement of Authorship

Department of Computer Science and Computer Engineering
STATEMENT OF AUTHORSHIP FOR GROUP ASSIGNMENTS

We each certify that our own contribution in the attached material is original work and that we have no knowledge of an act of plagiarism committed by any member of this group. We declare that no other person's work has been used without due acknowledgement. Except where we have clearly
stated that we have used some of this material elsewhere, it has not been presented by us for examination in any other course or unit at this or any other institution.

We have read the Academic Integrity Statute and completed the Academic Integrity Module relating to Student Discipline and Academic Misconduct, which are available on the University's website and understand that we are bound by such Policy, Statute and Regulation and that we may be subject to student discipline processes in the event of an act of plagiarism by us.

We understand the nature of plagiarism to include the reproduction of someone else's words, ideas or findings and presenting them as our own without proper acknowledgement. Further, we understand that there are many forms of plagiarism which include direct copying or paraphrasing from someone else's published work (either electronic or hard copy) without acknowledging the source; using facts, information and ideas derived from a source without acknowledgement; producing assignments (required to be independent) in collaboration with and/or using the work of other people; and assisting another person to commit an act of plagiarism. We understand that the work submitted may be reproduced and/or communicated by the University or a third party authorised by the University for the purpose of detecting plagiarism.

- Unit Code: **CSE3PRB**
- Unit Name: **INDUSTRY PROJECT B**
- Lecturer/Tutor's Name: **Dr Amin B. Abkenar**
- Supervisor's Name: **Aidin Bervan**
- Name of Document: **User Manual**

Student No.	Family Name	Given Name	Signature	Date
18938149	Petit	Remi	Remi Petit	31/10/19
18937397	Owczarek	Michael	Michael Owczarek	31/10/19
17699381	Morran	Joel	Joel Morran	31/10/19
19331567	Curnow	James	James Curnow	31/10/19
19145764	Shigetomi	Luis	Luis Shigetomi	31/10/19

Plagiarism
One form of academic cheating is plagiarism, the reproducing of someone else's words, ideas or findings and presenting them as one's own without proper acknowledgment.

There are many forms of plagiarism, including the following:

1. Direct copying of sentences, paragraphs or other extracts from someone else's published work (including on the Internet and in software) without acknowledging the source;
2. Paraphrasing someone else's words without acknowledging the source;
3. Using facts and information derived from a source without acknowledging it;
4. Using ideas directly derived from an identifiable author without acknowledging the source;
5. Producing assignments which should be the student's own, independent work in collaboration with and/or using the work of other people (e.g. a student or tutor).

Assisting another person to plagiarise material may be punished as severely as is plagiarism itself. Assisting plagiarism may involve a student lending work (or by posting it on the Internet for sale) which is intended for submission for assessment, or which has already been submitted so that it can be copied and handed in by another student as that student's own work.3

Students may find it helpful to discuss assignments with other students and their tutors. The University encourages students to communicate with one another in constructive ways about the learning process. Students may choose to assist each other, for example in discussing the approaches that might be taken to assignment topics or helping with the availability of reading materials. They should, however, write their assignments independently, except when they are asked to work on a project as a member of a group which is to submit a joint report, as equal contributors. In this case, except where the final written work consists of sections for which particular individuals take sole responsibility, the group as a whole assumes responsibility for it and the proper acknowledgment of any use made of the words or ideas of people outside the group.

The increased vigilance of the University in regard to plagiarism is in response to the perceived growing incidence of plagiarism. The new recommended penalties reflect the value the University places on academic honesty and how seriously it regards plagiarism offenses. The University will protect its reputation for academic integrity by ensuring that serious and serial offenders, who wish to obtain an unfair advantage through cheating, are given
penalties proportionate to the offenses committed.