

Portfolio

IWSN

Jesse van Gool

30 Maart 2021

Inhoudsopgave

Inleiding.....	3
Week 1	3
Week 2	4
Week 3	5
Week 4	6
EMON MINI PROJECT	7
Week 1	7
Week 2	7
Week 3	7
Week 4	8
Week 5	8
EMON Opdrachten	9
Stap 1.....	9
Opdracht 1	9
Opdracht 2	9
Stap 2.....	9
Opdracht 1	9
Opdracht 2	10
Opdracht 3	11
Stap 3.....	12
Opdracht 1	12
Stap 4.....	13
Opdracht 1	13

Stap 5.....	14
Opdracht 1	14
Demonstratie Video.....	14
Feedback	15
Leerervaringen	15
Het printplaat probleem	15
Het software probleem.....	16
Onder tijdsdruk werken.....	16

Inleiding

In dit document zullen de leerervaringen en opdrachten van IWSN te vinden zijn. Het is mogelijk dat niet alle opdrachten compleet zijn ivm tijdsdruk en hardware afwezigheid.

Alle code is te vinden op mijn git repo: <https://github.com/DutchLeopardNL/MiniProject-EMON>

Week 1

In week 1 zijn we bezig geweest met het opzetten van een stoplicht, door middel van een Arduino. De code die hiervoor is geschreven is als volgt:

```
1 #include <Arduino.h>
2 #include "trafficLight.h"
3 trafficLight* lightOne;
4 trafficLight* lightTwo;
5
6
7 void setup() {
8   // put your setup code here, to run once:
9   lightOne = new trafficLight(D1,D2,D3);
10  lightTwo = new trafficLight(D0,D5,D6);
11  pinMode(D1, OUTPUT); // RED
12  pinMode(D2, OUTPUT); // YELLOW
13  pinMode(D3, OUTPUT); // GREEN
14
15  pinMode(D0,OUTPUT); //RED
16  pinMode(D5,OUTPUT); //YELLOW
17  pinMode(D6,OUTPUT); //GREEN
18
19  pinMode(D7, INPUT_PULLUP); // Button
20  pinMode(D4, INPUT_PULLUP); // Button
21
22  // Switch off the leds
23  lightOne->turnOff();
24  lightTwo->turnOff();
25  lightOne->changeState(RED);
26  lightTwo->changeState(RED);
27
28  // Initialize the serial communication
29  Serial.begin(9600);
30
31 }
32
33 void loop() {
34   // put your main code here, to run repeatedly:
35   if ( digitalRead(D7) == LOW ) {
36     lightOne->goToGreen(lightTwo);
37     break;
38   }
39   if(digitalRead(D4) == LOW) {
40     lightTwo->goToGreen(lightOne);
41   }
42 }
```

```
#ifndef TRAFFICLIGHT_H
#define TRAFFICLIGHT_H

#include "Arduino.h"
#include <chrono>
#include <thread>

enum lightState {
  RED = 0,
  YELLOW = 1,
  GREEN = 2
};

class trafficLight {
  uint8_t _RED_LIGHT;
  uint8_t _YELLOW_LIGHT;
  uint8_t _GREEN_LIGHT;
  lightState currentState = RED;

public:
  trafficLight(uint8_t _RED_LIGHT, uint8_t _YELLOW_LIGHT, uint8_t _GREEN_LIGHT) {
    _RED_LIGHT = _RED_LIGHT;
    _YELLOW_LIGHT = _YELLOW_LIGHT;
    _GREEN_LIGHT = _GREEN_LIGHT;
    turnOff();
  }

  void changeState(lightState lightState) {
    switch (lightState) {
      case RED:
        digitalWrite(_RED_LIGHT, LOW);
        digitalWrite(_YELLOW_LIGHT, HIGH);
        digitalWrite(_GREEN_LIGHT, HIGH);
        currentState = RED;
        break;
      case YELLOW:
        digitalWrite(_YELLOW_LIGHT, LOW);
        digitalWrite(_RED_LIGHT, HIGH);
        digitalWrite(_GREEN_LIGHT, HIGH);
        currentState = YELLOW;
        break;
      case GREEN:
        digitalWrite(_YELLOW_LIGHT, HIGH);
        digitalWrite(_RED_LIGHT, HIGH);
        digitalWrite(_GREEN_LIGHT, LOW);
        currentState = GREEN;
        break;
    }
  }
};

#endif TRAFFICLIGHT_H
```

```
void goToGreen(trafficLight* otherLight) {
  Serial.println("THIS LIGHT: " + currentState);
  Serial.println("OTHER LIGHT: " + otherLight->currentState);
  if(currentState == RED) {
    if(otherLight->currentState == GREEN) {
      otherLight->goToRed();
    }
    changeState(GREEN);
  }
}

void goToRed() {
  if(currentState == GREEN) {
    Serial.println("GOING TO RED: " + currentState);
    delay(1000);
    changeState(YELLOW);
    Serial.println("GOING TO RED: " + currentState);
    delay(5000);
    changeState(RED);
    Serial.println("GOING TO RED: " + currentState);
  }
}

void turnOn() {
  digitalWrite(_RED_LIGHT, LOW);
  digitalWrite(_YELLOW_LIGHT, LOW);
  digitalWrite(_GREEN_LIGHT, LOW);
}

void turnOff() {
  digitalWrite(_RED_LIGHT, HIGH);
  digitalWrite(_YELLOW_LIGHT, HIGH);
  digitalWrite(_GREEN_LIGHT, HIGH);
}
```

Het belangrijkste deel van de code is in de trafficLight.h gezet. Hierin zijn methodes opgezet om op een systematische manier de lampen van groen naar rood te zetten of andersom. Door middel van op een knop te drukken werden deze methodes aangeroepen waarna ze switchen van state. Deze manier van werken is niet precies een FSM maar

Page 10 of 10

```
const mqttClient = mqtt.connect(config.mqtt.broker);

var Module = module.exports;
Module.publishEvent = publishEvent;

mqttClient.on('connect', () => {
    console.log('connected to MQTT!');

    mqttClient.subscribe("iwsn-wemos-monkey", function (err) {
        if (err) {
            console.log("MQTT Error: " + err.message);
        }
    });

    mqttClient.subscribe("iwsn-wemos-event-monkey", function (err) {
        if (err) {
            console.log("MQTT Error: " + err.message);
        }
    });
});

mqttClient.on('close', () => {
    console.log("MQTT client disconnected");
});

mqttClient.on('error', () => {
    console.log("mqtt error " + config.mqtt.broker );
});

mqttClient.on('message', function (topic, message) {
    message = JSON.parse(message.toString());
    topic   = topic.toString();

    if ( topic == "iwsn-wemos-monkey" && message.looptiming ) {
        wemos.addLog(topic, message);
    }

    if ( topic == "iwsn-wemos-monkey" && message.state ) {
        wemos.setState(topic, message);
    }
});

function publishEvent(event) {
    mqttClient.publish("iwsn-wemos-event-monkey", JSON.stringify({event: event}));
}
```

Week 3

Deze week hebben we de Xbee opgezet en de omgeving hiervan. Je hebt hiervoor 2 Xbees nodig, deze zetten dan een P2P connectie op. Hierbij is een Xbee de router en de andere de coördinator. Hier hebben we de 64-bits adressen gebruikt van elk van de nodes (source en destination). In een Zigbee netwerk wordt voor de controller ook het (destination)adres 0 gebruikt. De controller kan aan alle andere nodes bericht sturen door een broadcast met (destination)adres 0xFFFF. Hierna was het de bedoeling dat dit werd toegepast op 3 nodes, een controller en twee routers, dit is helaas niet gebeurd omdat wij deze hardware niet hadden. Verder was het de bedoeling om deze twee routers aan te sluiten op een arduino met een Xbeeshield. Deze gingen vervolgens een programma draaien waarbij steeds een letter verstuurd werd.

Hieronder is een voorbeeld waarbij de Xbees zijn geconfigureerd om berichten naar elkaar te sturen. Hierbij is er een controller en een router, op de Xbeeshield module runned hier code die getallen genereert. Ook hebben we gekeken wat de "send packet" functie binnen XCTU door een send sequence te maken die 10 keer "Makker" stuurde. Dit valt allemaal nog onder het AT gedeelte van de Xbee, aan het API gedeelte zijn we niet meer toegekomen ivm hardware problemen en tijdsnood.

```
2,139
2,122
2,123
2,100
2,174
2,156
2,124
2,127
MakkerMakkerMakkerMakkerMakkerMakkerMakkerMakkerMakkerMakker2,180
2,122
```

Jammer genoeg hadden we best veel problemen met de Zigbee's waardoor we geen tijd hadden voor de andere opdrachten.

Week 4

Voor week 4 is geen huiswerk te vinden op blackboard. In deze week zijn wij begonnen met het maken van het mini project. Dit proces is in het hoofdstuk hieronder beschreven.

EMON MINI PROJECT

Gedurende een periode van 5 weken heb ik samen met Noah geprobeerd een slimme meter bij mij thuis te installeren. Hieronder is een beschrijving van dit proces

Week 1

In week 1 heb ik de benodigde hardware opgehaald en thuis gesoldeerd waarna ik heb getest of dat de hardware functioneel was. Al snel kwam ik erachter dat de hardware defect was en kon hiervoor geen vervanging krijgen ivm het lage aantal setjes. Om toch te kijken wat we konden doen hebben we samen naar de opdrachten gekeken die aan het emon project verbonden zaten. We kwamen al redelijk snel tot de conclusie dat er geen lichtdiode was meegegeven en het al snel lastig werd om ook daadwerkelijk het knipperende lampje uit te lezen. We hebben daarna toch ervoor gekozen om de slimme meter via de kabel af te lezen in overleg.

Week 2

In deze week heeft Noah geprobeerd om een nieuw setje te halen en dit was succesvol. Jammer genoeg was het setje vergeten bij Breda Robotics waarna een van onze medestudenten deze mee naar huis had genomen. Hierdoor konden we pas in week 3 weer verder met het project. We hebben deze week wel al research gedaan naar andere onderdelen van het project, en zijn begonnen met de uitwerking hiervan. Met name het front-end en back-end gedeelte van het project. We hebben een begin gemaakt aan een webpagina die de resultaten van de meter uitdrukt en deze laat zien aan de bezoeker van de website. Ook hebben we wat werk gedaan aan het database gedeelte van het project.

Week 3

Deze week hebben wij samen het setje gesoldeerd. Na de aangeleverde code te hebben geupload kwamen we er snel achter dat er iets mis was met het bordje of de code. Gedurende de rest van de dag hebben we proberen uit te sluiten wat er mis ging maar we hebben niks kunnen vinden, alle hardware aansluitingen waren correct en er was geen kortsluiting te vinden bij de doorgekraste traces. We zijn daarna ook begonnen met het proberen van kleine stukjes code, en kregen na een tijdje wel sommige dingen werkend op het plaatje, wat al hoopvol was.

Week 4

In deze laatste week hebben wij weer beide gekeken wat er fout kon zijn, ook heb ik de hulp ingeschakeld van Maurice maar die kon ik niks vinden op de hardware, er is wel een aanrading geweest om zelf software te schrijven om te kijken wat er fout was. Na dit te doen heb ik niks kunnen constateren, alles lijkt goed te werken maar de data wordt niet verzonden naar mqtt. Zelf vinden wij het ook heel moeilijk om te zien wat er precies fout gaat in de code aangezien we niet kunnen debuggen en `Serial.println()` werkt ook niet meer na dat we de serial interface switchen.

Week 5

Dit is de week van de oplevering, op maandag hebben wij samen gekeken of dat we de slimme meter nog uit konden lezen, na een samenvoeging te hebben gemaakt van de code van Diederich en eigen gemaakte code hebben we nog steeds geen oplossing gevonden, we hebben kunnen detecteren dat de uitlees methode wel bereikt wordt maar dat de data niet naar mqtt wordt verstuurd en we weten niet waarom. In de avond ben ik nog alleen doorgegaan. Rond 9 uur kreeg ik berichten vanuit de arduino. Hierna heb ik nog verder gewerkt aan het opstellen van een mqtt client in C#. De dag hierna zijn wij weer samengekomen om de frontend te maken. Jammer genoeg is er geen tijd meer geweest om een XBee opstelling te maken.

EMON Opdrachten

Stap 1

Opdracht 1

Bedenk, ontwerp en maak een elektrische interface om de door jou uitgekozen energie meter uit te lezen. Verstandig is om in een 'testomgeving' (een knipperende led kun je makkelijk simuleren) het ontwerp te testen. Bespreek de schakeling met de begeleidende docent. Beschrijf je ontwerp in maximaal één A4 met als bijlage het elektrische schema en meetresultaten. In de bijlage is een schakeling gegeven die werkt op energiemeters met een knipperende led inclusief meetresultaten.

Voor opdracht 1 is het printplaatje vanuit school gebruikt.

Opdracht 2

Interface de schakeling op een echte energiemeter (thuis) en verifieer dat de schakeling werkt. Verzamel bewijsvoering door het doen van metingen (wellicht moet de scoop een avondje mee naar huis). Wees ervan overtuigd dat de data afkomstig van de interface van kwalitatief hoog niveau is (zonder ruis, zonder valse triggers enz. Zie ook de problemen in het forum op tweakers.net).

Werking van de schakeling zal te zien zijn gedurende de demonstratie.

Stap 2

Opdracht 1

Wat is de betekenis van de signalen die je hebt gevangen? Beschrijf in een 'formule/procedure' hoe je de (tijd tussen twee) gevangen pulsjes omzet naar 'energie' met als eenheid [kWh]. Bij een smart meter beschrijf je het protocol. Beschrijf ook de 'formule/procedure' waarmee je het momentele verbruikte energie kunt berekenen met als eenheid [W].

Bij de smartmeter kwamen we er al snel achter dat alle data een eigen code heeft. Door middel van deze codes hebben we de data kunnen filteren.

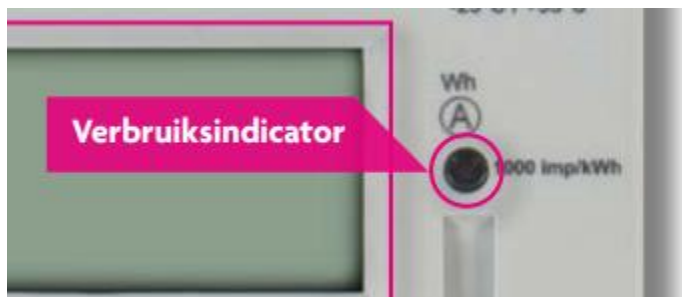
Waterkoker van 2000W die gedurende 4 minuten aanstaat	1000 imp/kWh (afgelezen op de meter)	Elke 1.8 sec een 'puls' of 'rond gedraaid'
	Momenteel verbruikt vermogen = 2000W	Opgenomen energie = 0.133 kWh. Dit kost circa 0.03€.

Beantwoord de volgende vragen:

- Wat kost een gemiddelde router aan energie per jaar in Euro's?
- Welke aannames maak je?
- Bespaar je veel geld door een NAS elke dag maar enkele uren aan te zetten? (Een NAS van het merk Synology heeft voor deze functie een speciale scheduler).

Wij hebben beide een Ziggo Connectbox. Hierdoor kunnen we zeggen dat wij beide 30W per uur verbruiken, dit getal is berekend door $12V \cdot 2.5A$ te doen. Als we uit willen rekenen wat wij op jaarbasis verbruiken kunnen we de volgende formule gebruiken: $\frac{(30 \cdot 24 \cdot 365)}{1000} \cdot 0.20$. Hierbij nemen we 20 cent als kosten per kWh, hierdoor komen we tot een bedrag van 52.56 euro per jaar. Bij een van ons staat thuis een NAS, deze NAS heeft een stroomverbruik rondt de 10W. Als we dit voor een jaar willen uitrekenen kunnen we dezelfde formule als hierboven gebruiken. Door dit te doen komen we op een bedrag van 17.52 euro, als het systeem de hele dag aan staat. Indien we een scheduler instellen en hiervan acht uur wegnemen neemt dit bedrag af tot 11.68 euro, indien het systeem dan 0W verbruikt.

Gelukkig hebben we de documentatie van de slimme meter bij de hand, hiermee kunnen we het gebruik per knipper gewoon opzoeken hierin. Op de slimme meter zelf staat dat per 1000 knippersignalen er een kWh is verbruikt.



Opdracht 2

Maak een keuze voor een database (op basis van welke criteria?) en implementeer de database. Geef antwoord op de volgende vragen:

- Is het mogelijk de database te bevragen (query's) voor het maken van grafieken (historische data). Bijvoorbeeld voor 'het verbruik van afgelopen 3 dagen in een 10 minuten interval' of 'de gemaakte kosten gedurende afgelopen maand per dag'. In een later stadium gaan we met een web applicatie en/of een mobile applicatie de database bevragen. Houd er rekening mee dat sommige query's heel lang kunnen duren! Hoe zou je dit kunnen oplossen?
- Is de databases geschikt om resultaten van andere locaties te vergelijken met lokale gegevens? Voor besparing van energie is het wenselijk het eigen verbruik te spiegelen aan dat van anderen. Beschrijf het ontwerp en beantwoord de vragen in maximaal één A4. Geef ook enkele query's met bijbehorende resultaten. Plot resultaten in Excel en verifieer de gemeten data.

Om een database op te stellen zullen we C# gebruiken samen met verschillende frameworks. Deze frameworks zijn Mediatr, Entity Framework en Swagger. Door middel van deze frameworks kunnen wij een code-first database opstellen. Indien wij hierna onze modellen aanpassen kunnen we de database ook gemakkelijk updaten.

Opdracht 3

Voor het transport van de data naar de server is een IoT geschikt protocol noodzakelijk. Let op: in eerste instantie moet de worden gepubliceerd op de broker van het SendLab. Pas daarna is de keuze voor een geschikt IoT protocol vrij. Beantwoord de volgende vragen:

- Wordt de data bewerkt voordat deze wordt verzonden? Is er sprake van 'data naar informatie' transformatie?
- Kun je een uitspraak doen over de integriteit van de data/informatie die uiteindelijk wordt opgeslagen.

Op het moment stuur ik de data vanuit de Arduino naar de mosquitto server, tegelijkertijd vang ik deze op via een C# client applicatie, deze applicatie verwerkt het naar een bruikbaar model. Deze data wordt hierna opgeslagen in de database

Stap 3

Opdracht 1

Ontwerp op basis van Zigbee sensoren voor bijvoorbeeld temperatuur, luchtdruk, luchtvochtigheid, CO2 en/of daglicht. Combineer deze met de informatie wat betreft energieverbruik. Wat is hierbij het gestelde doel? Beschrijf de ontwerpen en motiveer de gemaakte keuzes. Geef ook aan waar de gemeten grootheden worden opgeslagen. Maak van het systeem een architectuurdiagram.

Deze opdracht is in verband met tijd limitaties en hardware problemen niet gedaan.

Stap 4

Opdracht 1

Ontwerp een service om de informatie uit de database te ontsluiten. De service moet een zinvolle interface hebben. Ook moet de service toegangscontrole hebben met behulp van web-tokens. Documenteer ontworpen service met bijvoorbeeld Swagger. Geef aan wat de gebruiker kan verwachten en wat het resultaat is. Geef hierbij telkens een voorbeeld. Is de gemaakte service flexibel om straks een frontend en/of mobile app te bedienen?

Deze service zal gerealiseerd worden door middel van EntityFramework en Mediatr zoals eerder is beschreven. Swagger is gebruikt om de API te visualiseren. Er is gekozen voor deze implementatie omdat ik deze al op mijn stage had gebruikt voor een ander project

EMONAPI 1.0 OAS3
https://localhost:44371/swagger/v1/swagger.json

Datagram

GET /api/Datagram/lastDatagram

GET /api/Datagram/datagrams/{amount}

Parameters Try it out

Name	Description
amount * required integer(int32) (path)	amount

Responses

Code	Description	Links
200	Success	No links

Media type: text/plain

Controls: Accept header.

Example Value | Schema

```
{
  "datagrams": [
    {
      "id": "string",
      "timestamp": "string",
      "currentUsage": 0,
      "totalLow": 0,
      "totalHigh": 0,
      "returnLow": 0,
      "returnHigh": 0,
      "gasUsage": 0,
      "signature": "string"
    }
  ]
}
```

Stap 5

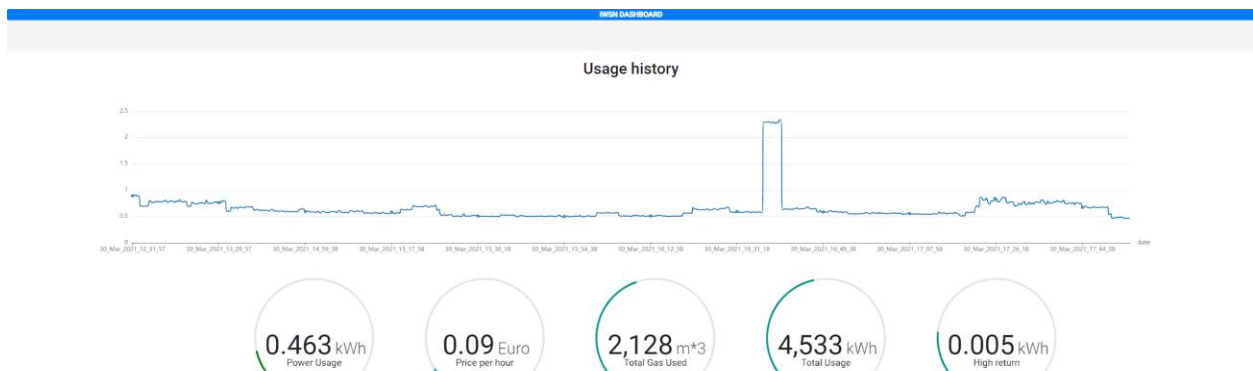
Opdracht 1

Ontwerp en implementeer een cliënt site (webapplicatie) waarmee je het systeem visualiseert. Tenminste de volgende informatie moet worden getoond:

- Momentele Energieverbruik (Watt)
- Verbruikte energie afgelopen N uren per uur (kWh)
- Trend
- Gemaakte kosten
- Gemeten grootheden van andere sensoren
- Status
- Verbruik bij anderen

Tip: gebruik een UI-framework zoals bijvoorbeeld Angular Material of Bootstrap.

Een foto van ons dashboard is hieronder te zien.



Demonstratie Video

<https://streamable.com/0rf742>

Feedback

Deze feedback is geleverd door Vincent:

- Verslag ziet er zeer netjes en verzorgd uit, alles is zover mogelijk beschreven.
- De styling is ook erg prettig om te lezen
- Er is een duidelijk verdeling gemaakt tussen de werkzaamheden, zoals de practicum opdrachten en het eindproduct.
- Het EMON mini-project is duidelijk en stapsgewijs uitgelegd.
- De afbeeldingen zijn prettig en leesbaar geplaatst, maar zijn niet voorzien van een onderschrift. - De inleiding is iets wat kort voor het gevoel, maar verteld ook wel waarom dit het geval is.
- Algemeen: Dit verslag is prettig om door te nemen. Er zijn nette verwijzingen gemaakt, waardoor het opzoeken van informatie erg gemakkelijk gaat. Het is jammer dat deze periode erg lastig is verlopen, omdat dit ook ten koste gaat van de algehele informatie die hierdoor ontbreekt. De demonstratievideo is kort, maar geeft duidelijk weer hoe onderdelen met elkaar werken. Al met al een prima verslag dat er verzorgd uitziet.

Leerervaringen

Het printplaat probleem

Voor het EMON project van Noah en mij had ik thuis de hardware liggen. In de eerste week van het project heb ik het kitje zelf proberen te solderen, dit is niet gelukt. Het led lampje op de printplaat branden verschillende kleuren, zelf weet ik nog steeds niet waar dit aan lag aangezien alles goed was gesoldeerd en de traces waren doorgekrast. De week hierop heb ik Noah een nieuw kitje op laten halen en deze hebben wij samen gesoldeerd, dit was mijn 2de keer solderen en voor Noah de eerste keer. Na de code van Diederich geüpload te hebben kwam ik er snel achter dat er iets mis was, maar wat wist ik niet. De printplaat bleef resetten, eerst dacht ik dat het kortsluiting was maar dat was het niet. Om dit op te lossen heb ik gekeken of ik zelf code kon uploaden en deze wel te laten werken. Dit werkte, jammer genoeg was het toen al de laatste vrijdag. In totaal heb ik denk 40+ uur geprobeerd om een probleem op te lossen op 2 verschillende printplaten en ik weet nog steeds niet waar het aan lag. Gelukkig heb ik vrijdag kunnen zorgen dat ik in ieder geval code kon uploaden. Hiervan heb ik geleerd dat het altijd handig is om eigen code uit te proberen om te kijken of het echt aan de hardware ligt.



Het software probleem

De maandag hierna heb ik de gehele dag besteed om te proberen om de P1 poort uit te lezen na 9 uur aan debuggen en verschillende manieren van uitlezen kwam ik erachter dat in de PubSubClient de mqtt limit te laag stond en deze heb ik hoger gezet. Hierna werkte alles, soms is het dus ook handig om een library te bekijken.

Onder tijdsdruk werken

Dinsdag (de laatste dag) heb ik besteed met het opzetten van de frontend en de API doormiddel van C# en Angular. Hierbij was het het moeilijkst om de rauwe data te parsen. Hiervoor heb ik een klein parser algoritme geschreven die checked op de codes voor de data. Toen dit klaar was was het al 1 uur. Nu moest de API en de frontend nog worden gemaakt. Op dit moment besloot ik ook om de Zigbee niet toe te gaan voegen aangezien hier echt geen tijd meer voor was. Het opzetten van de API ging vrij soepel jammer genoeg duurde het vrij lang omdat het opzetten van een API in C# veel klassen vereist die allemaal aan elkaar gelinked zijn, aangezien er ook vrij weinig documentatie is over de manier waarop wij het hebben gedaan kostte dit ook extra tijd. Na het maken van de API met 2 simpele queries: haal de laatste datagram op en haal x aantal datagrammen op. Was het tijd om de frontend te maken. Voor de frontend hebben we voornamelijk gebruik gemaakt van ngx-echarts en ngx-gauges, dit zijn 2 libraries om data weer te geven in verschillende vormen. Uiteindelijk vindt ik dat we in verband met de gelimiteerde tijd toch een mooi product hebben opgeleverd zeker aangezien we eigenlijk maandag pas van start konden gaan, hierdoor heb ik ingezien dat ik veel doorzettingsvermogen heb.

Deze periode heb ik vaak dagen gehad waar ik 12+ uur heb gewerkt aan een vak zelfs als het niet lukte gaf ik het nooit op.