25-4-2016

Michiel Zyde & Tiziano Colpaert

STUDENTEN 2NMCT3 HOWEST

VERSLAG

OpenAquarium – Arduino – Raspberry Pi



Inhoud

[1. OMSCHRIJVING 1](#_Toc449992713)

[2. GEBRUIKTE TECHNOLOGIEËN 2](#_Toc449992714)

[ARDUINO 2](#_Toc449992715)

[RASPBERRY PI 3](#_Toc449992716)

[OPEN AQUARIUM 4](#_Toc449992717)

[3. OPENAQUARIUM LIBRARY 5](#_Toc449992718)

[4. HANDLEIDINGEN 6](#_Toc449992719)

[ARDUINO 6](#_Toc449992720)

[Temperatuur uitlezen: 6](#_Toc449992721)

[Vissen voederen: 6](#_Toc449992722)

[Huidige code 7](#_Toc449992723)

[RASPBERRY PI 8](#_Toc449992724)

[Seriële communicatie : 8](#_Toc449992725)

[5. PROBLEMEN 11](#_Toc449992726)

[Serieel uitlezen 11](#_Toc449992727)

[CGI Scripts 11](#_Toc449992728)

[6. TAAKVERDELING 12](#_Toc449992729)

[7. SAMENVATTING 12](#_Toc449992730)

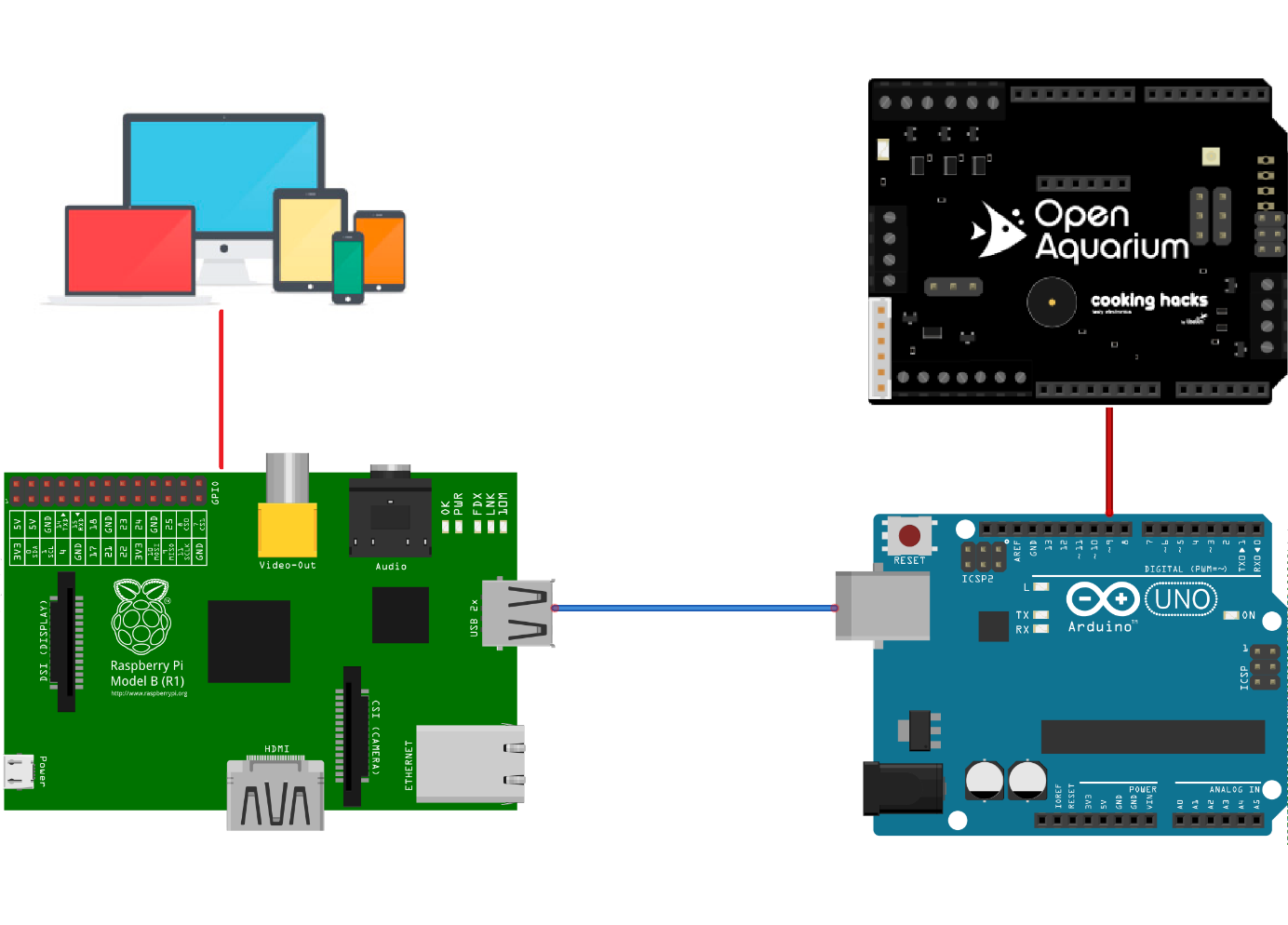
[POSITIEVE PUNTEN 12](#_Toc449992731)

[NEGATIEVE PUNTEN 12](#_Toc449992732)

# OMSCHRIJVING

Met behulp van een Arduino bordje en een Raspberry Pi moeten we via een interface kunnen communiceren met een aquarium. Dit gebeurd via een Arduino module genaamd ‘OpenAquarium’. Zo moet het mogelijk zijn om de lampen van het aquarium aan of uit te doen, temperatuur uit te lezen, feeder van de vissen aan te sturen etc.

Het zal dus de bedoeling zijn om de Arduino te laten communiceren met de Pi en op die Pi een webserver te laten draaien met een interface. Natuurlijk zullen we ook in de andere richting moeten kunnen werken. (bv: feeder bedienen vanaf webinterface).



# GEBRUIKTE TECHNOLOGIEËN

## ARDUINO

Arduino is een opensource-computerplatform dat is opgebouwd rond de ATmega168 -microcontroller van Atmel en het softwareontwikkelplatform Processing. Dit platform is gemaakt voor mensen die geïnteresseerd zijn in het maken en ontwerpen van slimme en creatieve objecten die kunnen reageren op hun omgeving. Met Arduino is het mogelijk apparaten en objecten te creëren die reageren op hun omgeving door middel van digitale en analoge inputsignalen. Op basis van deze input kan een Arduinoschakeling autonome acties initiëren door het afgeven van digitale en analoge outputsignalen.

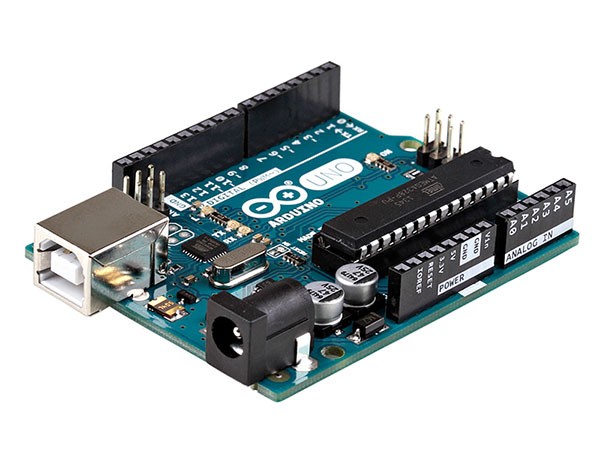
Input kan bijvoorbeeld worden gegenereerd door schakelaars, lichtsensoren, bewegingsmeters, afstandsmeters, temperatuursensoren, of op basis van commando's afkomstig van internet, een radiomodule of een ander apparaat met een seriële interface. Outputsignalen kunnen bijvoorbeeld motoren, lampjes, pompjes en beeldschermen aansturen, maar ook input genereren voor een andere Arduinomodule. In ons geval maken we gebruik van de Arduino UNO, dit is de nieuwste versie van het ontwikkelbord!

*Voordelen:*

* Codevoorbeelden beschikbaar.
* Ready to use!
* Communicatie mogelijk met andere devices bv. Raspberry Pi.

*Nadelen:*

* Kostprijs is redelijk hoog.



## RASPBERRY PI

De Raspberry Pi is de naam van diverse singleboardcomputers gebaseerd op een ARM-processor die tegen een minimale prijs worden vervaardigd en verkocht. De Raspberry Pi werd ontwikkeld aan de Universiteit van Cambridge en was bedoeld voor educatieve doeleinden. Er zijn verschillende modellen en versies uitgebracht. De hardware van de verschillende versies van de Raspberry Pi kenmerkt zich door een open structuur die lijkt op de open structuur van de eerste IBM Personal Computers.

De aanwezigheid van de GPIO-bus maakt het mogelijk om een addon-board aan een Raspberry Pi te koppelen en de aanwezigheid van USB-poorten, een DSI- en een CSI-connector maken het mogelijk om meerdere USB-devices en een display en een camera aan het moederbord te koppelen. De addon-boards kunnen gemakkelijk verwisseld worden evenals de microSD-kaart met het live-system en de software

*Voordelen*

* Kostprijs is redelijk laag.
* Communicatie met andere devices is mogelijk bv. Arduino bordje.

*Nadelen*

* Geen ondersteuning voor een x86 OS.



## OPEN AQUARIUM

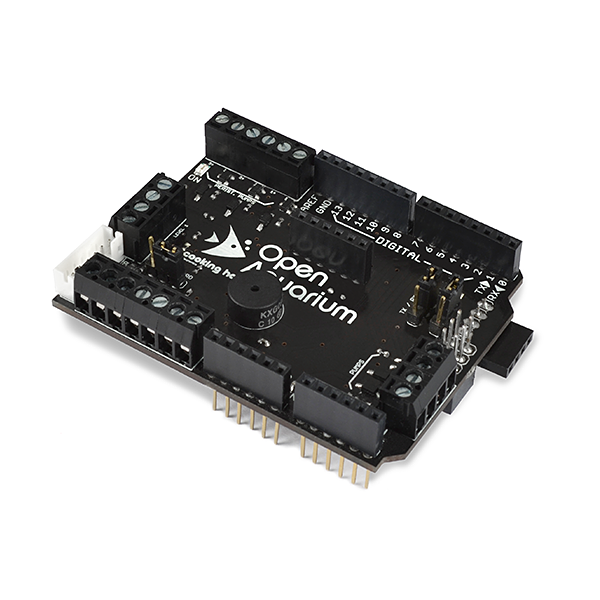
Naast het Arduino bordje en de Pi maken we ook gebruik van een uitbreidingsmodule, deze module heet ‘OpenAquarium’ en kan geplaatst worden op een standaard Arduino bordje. Met deze uitbreiding kun je bijvoorbeeld de temperatuur van een aquarium uitlezen, lampen van het aquarium aansturen etc. Ook is het nog steeds mogelijk om zaken te programmeren voor de onderliggende Arduino Uno.

*Voordelen*

* Automatische controle over aquarium.
* Gemakkelijk aan te sluiten op Arduino.

*Nadelen*

* Kostprijs is redelijk hoog.
* Zelf interface & functies programmeren.



# OPENAQUARIUM LIBRARY

Bij het OpenAquarium bordje hoort ook een library die al enkele functies ter beschikking stelt. Dit is een library die speciaal geschreven is voor de Arduino module. Hieruit hebben we enkele codelijnen kunnen gebruiken om onze basisfuncties van de module uit te testen.

Naast de basisfuncties hebben we ook flags nodig, een soort van functies die klaar staan om uitgevoerd te worden bij een bepaalde trigger. Deze flags hebben we nodig om vanaf onze webinterface functies te kunnen triggeren en zo bepaalde acties te ondernemen.

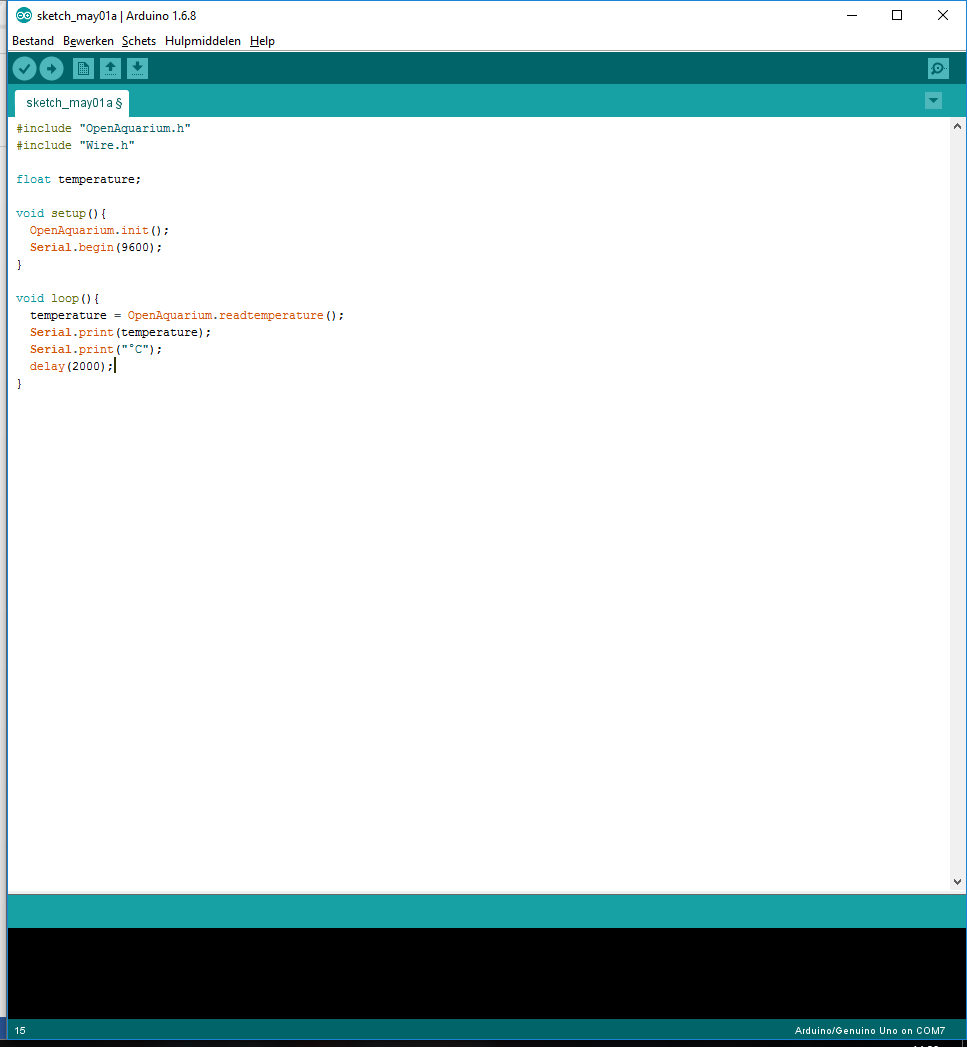
Alle documentatie vind je hier:

<https://www.cooking-hacks.com/documentation/tutorials/open-aquarium-aquaponics-fish-tank-monitoring-arduino/>

# HANDLEIDINGEN

## ARDUINO

Hieronder zullen we enkele code voorbeelden plaatsen die we uiteindelijk zullen gaan gebruiken in ons totaal pakket, als we met alles klaar geraken. Al deze code voorbeelden werden geprogrammeerd voor de OpenAquarium module en hebben telkens allemaal key-functies voor ons project.



Temperatuur uitlezen:

Deze code laat ons toe om de temperatuur van onze aquarium uit te lezen. We printen om de 2 seconden de temperatuur van het water naar de seriële monitor.

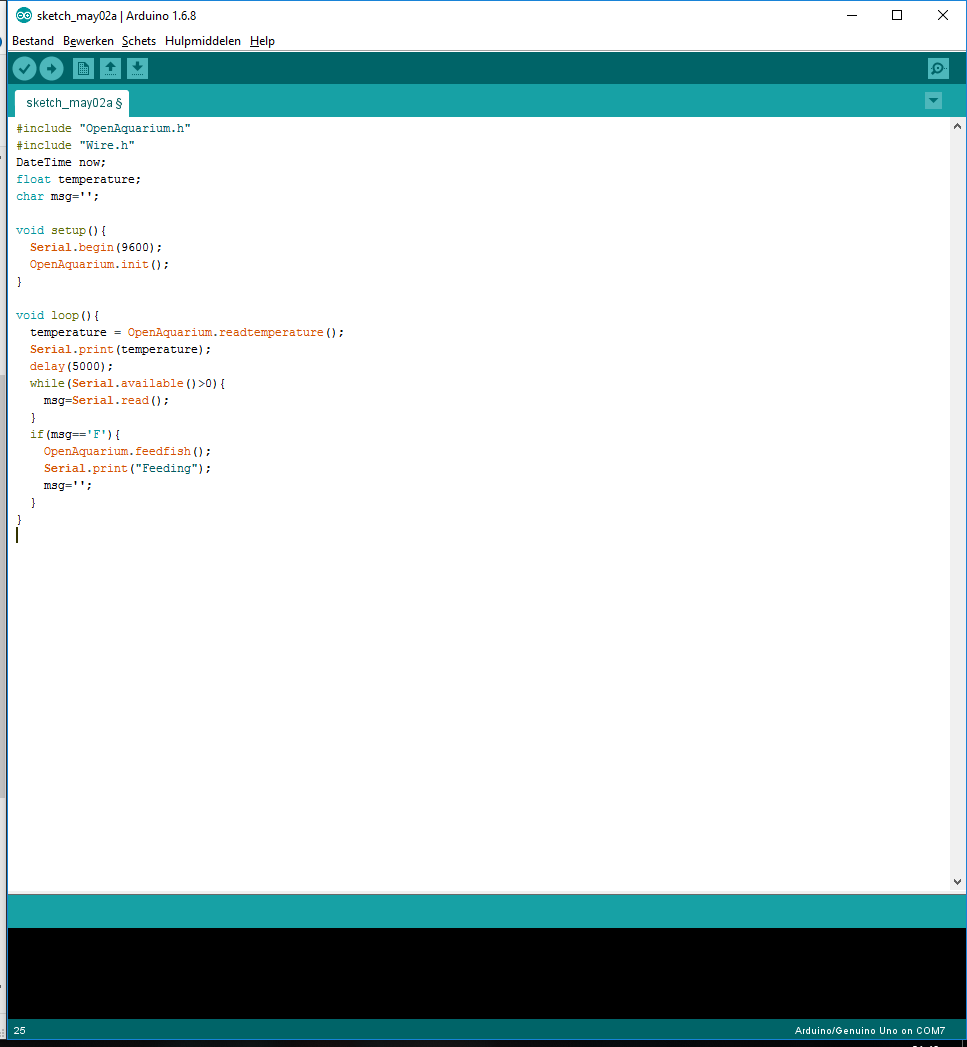
Later is het dan de bedoeling dat we deze data uit de seriële monitor zullen displayen op onze webinterface.

### Vissen voederen:

Deze code laat ons toe om op bepaalde tijdstippen de vissen te voederen. Met deze code gaan we dus de huidige tijd gaan ophalen en dan tussen bepaalde tijdstippen onze feeder gaan aansturen.

Voor onze demo zullen we de feeder direct 1 maal aansturen, zo hoeven we niet te wachten tot een bepaald tijdstip.

### Huidige code



Dit is onze huidige code, met deze code gaan we om de vijf seconden onze temperatuur gaan updaten op de interface. We zorgen er ook voor dat de Arduino de hele tijd open staat voor seriële communicatie. Als we dan uiteindelijk een commando geven vanaf de webserver is het de bedoeling dat de Arduino deze commando’s gaat uitlezen. Als in dit geval de binnenkomende string gelijk is aan “F”, dan gaan we de feeder aansturen en krijgen de vissen eten.

## RASPBERRY PI

Seriële communicatie :   
Om er voor te zorgen dat het Arduino bordje kan communiceren met de Raspberry Pi, moeten we eerst een sketch runnen op de Arduino die om de x aantal seconden serieel iets print. Daarna sluiten we de Arduino via USB aan op de Pi. Om op de Pi serieel dingen van de Arduino te kunnen uitlezen hebben we pySerial nodig.

#### pySerial installeren.

* **Instructies** voor het **installeren** van pySerial:
  + Download pySerial op je laptop: <http://sourceforge.net/projects/pyserial/files/pyserial/2.5/>
  + Unzip de tar.gz
  + Upload de pySerial folder met SFTP naar een temp folder op de Pi
  + Maak verbinding met de Pi via SSH (op je laptop)
  + Navigeer naar de temp folder en installeer pySerial (sudo python setup.py install)
* Eenmaal pySerial geïnstalleerd is moeten we nog **2 zaken** doen vooraleer we kunnen gebruik maken van de van de seriële poorten:
  + sudo usermod -a -G dialout pi (geeft serial access aan de gebruiker)
  + 2. Voeg ' KERNEL=="ttyACM0", SYMLINK+="ttyS7" ' toe aan /etc/udev/rules.d/ (om de USB connectie te kunnen gebruiken/zien)
* **Instructies** voor het **uitlezen** van seriële data op de Pi:
  + Type 'Python' + enter
  + Type 'import serial'
  + Type 'ser = serial.Serial(‘/dev/ttyACM0’, 9600)' (geef de seriële variabele een goede naam zoals ser, en stel de juiste baudrate in)
  + Type 'while 1 :' + enter
  + Type 'ser.readline()' (nu zal de Pi de hele tijd de data van de Arduino uitlezen)

Na al deze instructies te hebben uitgevoerd kun je enkel nog maar data van het Arduino bordje aflezen op de Pi, nog niet op de webinterface. Ook is het mogelijk om met pySerial al data te gaan versturen naar de Arduino. Dit kun je doen door je variabele in te typen en dan een ‘.write()’ te gaan doen. Dus als we bijvoorbeeld een string willen doorsturen doen we dit door ‘ser.write(‘test’)’, later is het dus de bedoeling dat we deze string dan gaan opvangen en een functie gaan uitvoeren.

De volgende stap is het opstellen van een webserver om de verkregen informatie te displayen op een website.

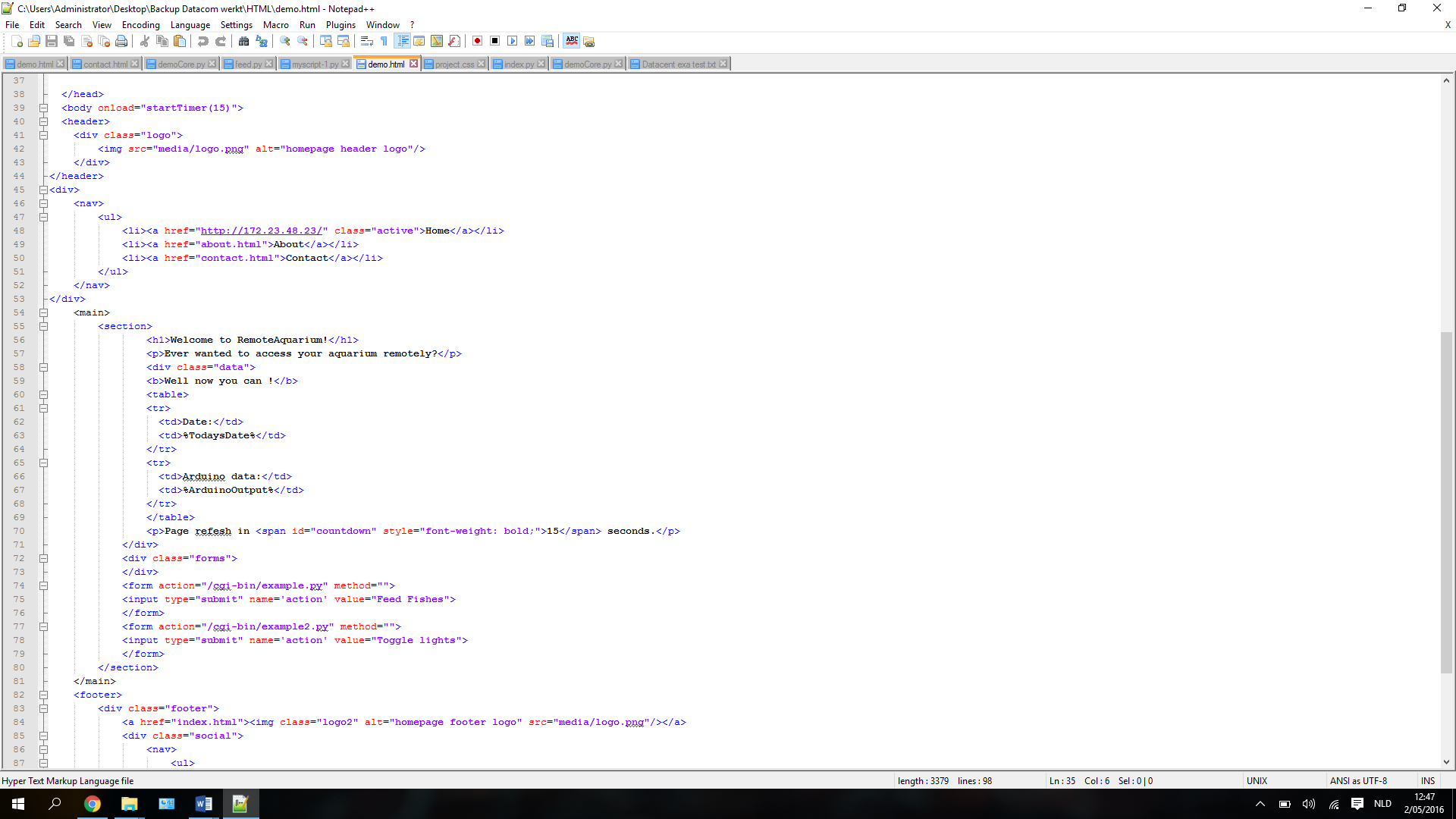
#### Apache2 opstellen

* **Instructies** voor het **opstellen** van een webserver:
  + - Type ‘apt-get install apache2’
    - Type ‘usermod –a –G dialout www-data’
    - In de sites-enabled folder zit er een config file die de apache server configureert, deze heeft enkele aanpassingen nodig en deze zijn hier te vinden:
    - http://pastebin.com/vLkikT2m
    - Type ‘a2enmod wsgi’
    - Type ‘/etc/init.d/apache2’ restart

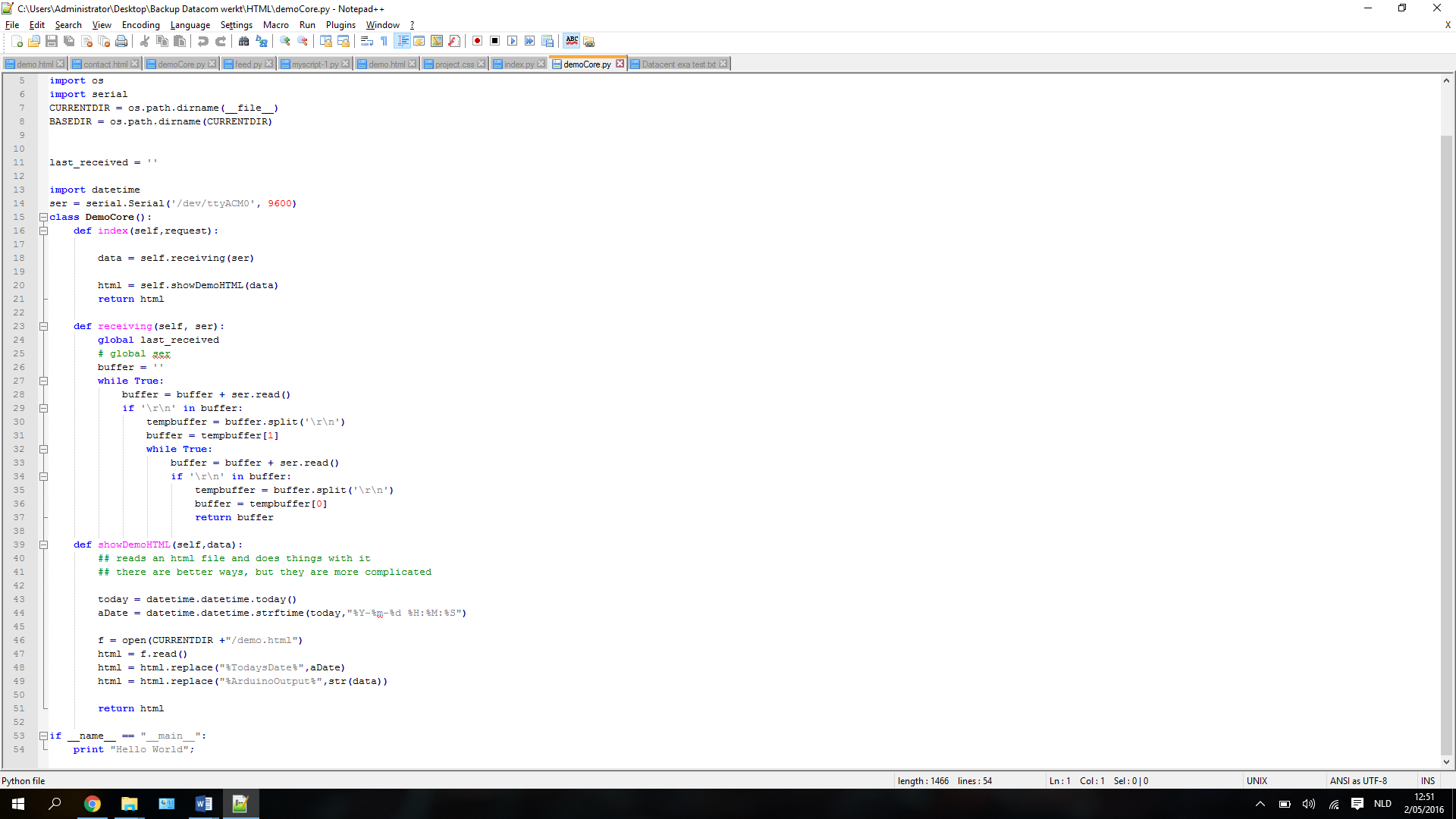
Eenmaal de apache server herstart is kunnen we een html pagina instellen als homepage. Aan de hand van Filezilla uploaden we een simpele html file over FTP, daarna hebben we een heel eind moeten zoeken naar hoe we de info van de Pi naar de interface kregen. Na enkele uren research bleek dit op te lossen met python.

We schreven dus een ‘index.py’ file die ervoor zorgde dat we op de correcte html file terecht kwamen. De index file zorgt er tevens ook voor dat onze klasse uitgevoerd wordt op de html file. Deze klasse werd geschreven in de ‘demoCore.py’ file, deze zorgt ervoor dat de seriële communicatie opgezet wordt en de data van de Pi naar de interface getransferd wordt.

In de html file zijn er placeholders aanwezig om de data weer te geven. Deze placeholders worden door de demoCory.py file aangepast wanneer er informatie van de Arduino naar de Pi wordt gestuurd.

Dit is de table in de html file, ‘%ToydaysDate%’ en ‘%ArduinoOutput%’ zijn de placeholders voor tijd en data.

Deze placeholders worden dan later vervangen door echte data aan de hand van enkele lijnen code in de demoCore.py file. (hieronder)

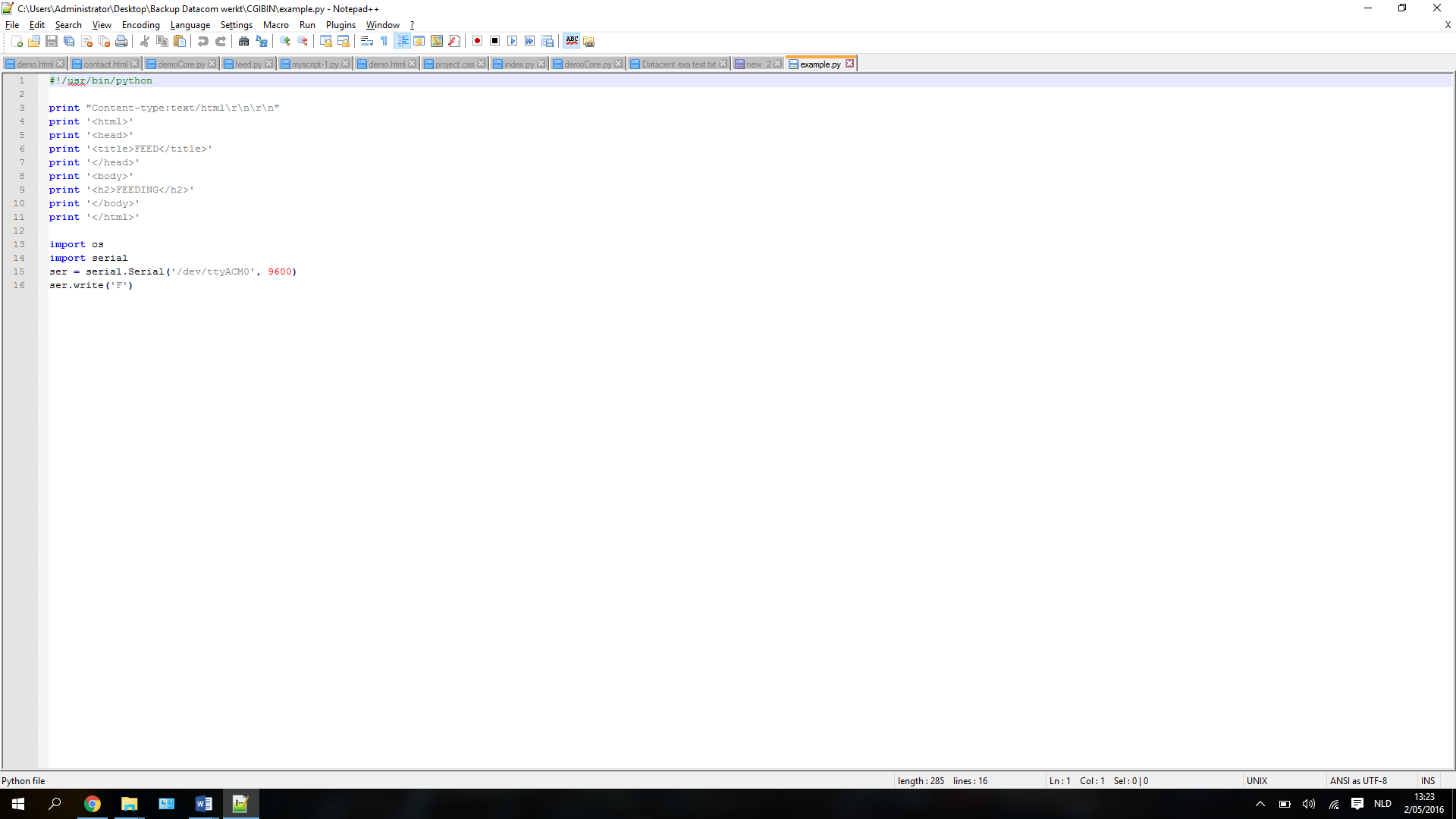


#### Cgi scripts.

Om ervoor te zorgen dat we ook in de andere richting kunnen werken hebben we iets nodig dat serverside kan communiceren met de pi. Na even researchen kwamen we terecht op een site die ons voorstelde te werken met CGI scripts.

Standaard is er in een Linux directory een map genaamd cgi-bin, deze is te vinden onder /usr/lib. Deze folder is gemaakt om cgi scripts in op te slaan, als je deze folder en files de correcte permissions geeft dan kan de webserver hier gebruik van maken. Tijdens de setup van onze apache2 hebben we ook de config-file hiervoor aangepast (zie instructies apache2), zodat deze gebruik kon maken van de cgi-bin folder.

Hieronder zie je een voorbeeld van zo’n CGI script die ervoor zorgt dat we een serialwrite kunnen gaan triggeren vanaf onze webserver. Hierin doen we eigenlijk ongeveer hetzelfde als in de demoCore.py, maar in plaats van data uit te lezen gaan we eigenlijk gewoon data serieel gaan versturen.



In het CGI script kunnen we in verschillende talen programmeren, het moet enkel eerst gedefinieerd worden. Zo kun je zien dat we hier gekozen hebben voor python. We hebben eerst enkele lijnen html code geschreven om geen lege pagina te verkrijgen als dit script uitgevoerd wordt. Daarna importeren we de nodige libraries en sturen we serieel een ‘F’ door, die we dan in ons Arduino programma gaan opvangen en een bepaalde functie triggeren.

# PROBLEMEN

## Serieel uitlezen

Voor we goed wisten waar we aan begonnen wilden we zo snel mogelijk al data kunnen versturen tussen Arduino en Pi. Door deze slechte beslissing zijn we direct begonnen met Minicom te installeren op de Pi en alles klaar te zetten voor seriële communicatie tussen beide. We waren snel blij dat we zaken van de Arduino konden displayen op de Pi, maar in principe konden we verder niks meer doen met Minicom. Het is pas in het labo erna dat we dit door hadden, waardoor we veel tijd hebben verloren. Minicom hebben we dus uiteindelijk vervangen door pySerial, omdat je daarmee meer kon bereiken.

## CGI Scripts

Op het moment dat de eindmeet in zicht kwam, waren we er in geslaagd om data van de Arduino op onze interface te displayen. Nu restte ons nog het omgekeerde proces. Dit was een lastige opdracht want we hadden niet veel tijd meer en er was niet zo veel te vinden online over deze soort van communicatie. Gelukkig vonden we dan een website die ons een beetje in de juiste richting had gestuurd. Ons probleem was op te lossen met CGI Scripting!

Na een hele labo les te vloeken en te blazen waren we er uiteindelijk in geslaagd om de scripts op onze server te krijgen, maar deze werden dan nog niet uitgevoerd. Een collega snelde ons te hulp en probeerde het één en het andere waardoor er uiteindelijk tien minuten voor tijd niks meer werkte. Gelukkig liet Mr. Desloovere ons toe om dit nog even op te lossen vooraleer hij het klaslokaal ging afsluiten.

Uiteindelijk zijn we dan opnieuw begonnen met onze Pi te configureren en hebben we onze webinterface dan toch opnieuw aan de praat gekregen. Erna zijn we er ook nog in geslaagd om de scripts te laten werken, dit was gewoon een kwestie van de folders en files de correcte rechten toe te kennen.

# TAAKVERDELING

|  |  |  |
| --- | --- | --- |
|  | Michiel Zyde | Tiziano Colpaert |
| OPZETTEN PI |  | X  45min |
| GITHUB STRUCTUUR | X  1h40 |  |
| WEBINTERFACE AANMAKEN |  | X  3h30 |
| ARDUINO CODE schrijven + TESTEN | X  2h30 | X  45min |
| Seriële COmmunicatie Arduino en pi | X  2h | X  2h30 |
| PYTHON CODE uitproberen |  | X  1h15 |
| INTERACTIEVE INTERFACE | X  1h | X  3h |
| document opstellen | X  2h | X  4h |
| Presentatie maken | X  30min |  |
|  | +-9h40 | +-15h45 |

# SAMENVATTING

## POSITIEVE PUNTEN

Het was leuk dat we zelf moesten uitzoeken hoe we ons project tot een goed einde gingen brengen. Hierdoor moest je echt wel zelfstandig te werk gaan en alles opzoeken. Het gevolg was dat onze kennis op het gebied van datacommunicatie toch wel een stuk uitgebreid werd.

We mochten in een team van 2 personen samenwerken en hadden een vrije keuze om te kiezen welk project we gingen doen, logischerwijs hetgeen ons het meeste interesseerde. Doordat we met 2 waren was het ook veel makkelijker om fouten of mogelijke problemen op te lossen.

## NEGATIEVE PUNTEN

Het was jammer dat we thuis niet konden verder werken aan ons project aangezien alle benodigdheden in school lagen en daardoor niet beschikbaar waren. Sommige projecten hadden dit probleem niet maar hierdoor zijn wij redelijk wat tijd verloren, vooral als we problemen hadden konden we deze maar in het labo oplossen.