

**PROBLEEMOPLOSSEN EN ONTWERPEN, DEEL 3**

***CWB2***

*Schrooten Bernd*

*Schryvers Andreas*

*Sels Shoera*

*Switsers Sander*

*Van den Berghe Pieter*

*Van Laere Nathan*

**Quantified Bike**

Eindverslag

Co-titularis

Professor Duval E.

Begeleider(s)

Charleer Sven

De Croon Robin

Klerkx Joris

Santon Jose Luis

ACADEMIEJAAR 2014-2015

# INHOUDSTAFEL

[INHOUDSTAFEL 2](#_Toc405306500)

[1. Teaminformatie 3](#_Toc405306501)

[2. Introductie 3](#_Toc405306502)

[3. Scenario’s 4](#_Toc405306503)

[User story 1 4](#_Toc405306504)

[User story 2 4](#_Toc405306505)

[4. Architectuur 5](#_Toc405306506)

[Fysieke architectuur 5](#_Toc405306507)

[5. Softwarearchitectuur 6](#_Toc405306508)

[6. Technologie 6](#_Toc405306509)

[Raspberry Pi 6](#_Toc405306510)

[Arduino 7](#_Toc405306511)

[JavaScript 8](#_Toc405306512)

[jQuery 8](#_Toc405306513)

[JSON 8](#_Toc405306514)

[Visualisatie Bibliotheken 8](#_Toc405306515)

[7. Ontwikkelingsproces 9](#_Toc405306516)

[8. Vakintegratie 11](#_Toc405306517)

[9. Besluit 12](#_Toc405306518)

[10. Appendixen 13](#_Toc405306519)

[Appendix 1: Geleverde werk 13](#_Toc405306520)

[Appendix 2: Planning 13](#_Toc405306521)

[Appendix 3: Mind map brainstorm 13](#_Toc405306522)

[Bronvermelding 15](#_Toc405306523)

# Teaminformatie

Alle groepsleden volgen de bachelor in de ingenieurswetenschappen. Volgende lijst vat de nodige gegevens samen.

**Naam Jaar Richting (major – minor)**

Schrooten Bernd 2 Computerwetenschappen - Elektrotechniek

Schryvers Andreas 2 Computerwetenschappen - Elektrotechniek

Sels Shoera 2/3 Elektrotechniek - Computerwetenschappen

Switsers Sander 2 Computerwetenschappen - Elektrotechniek

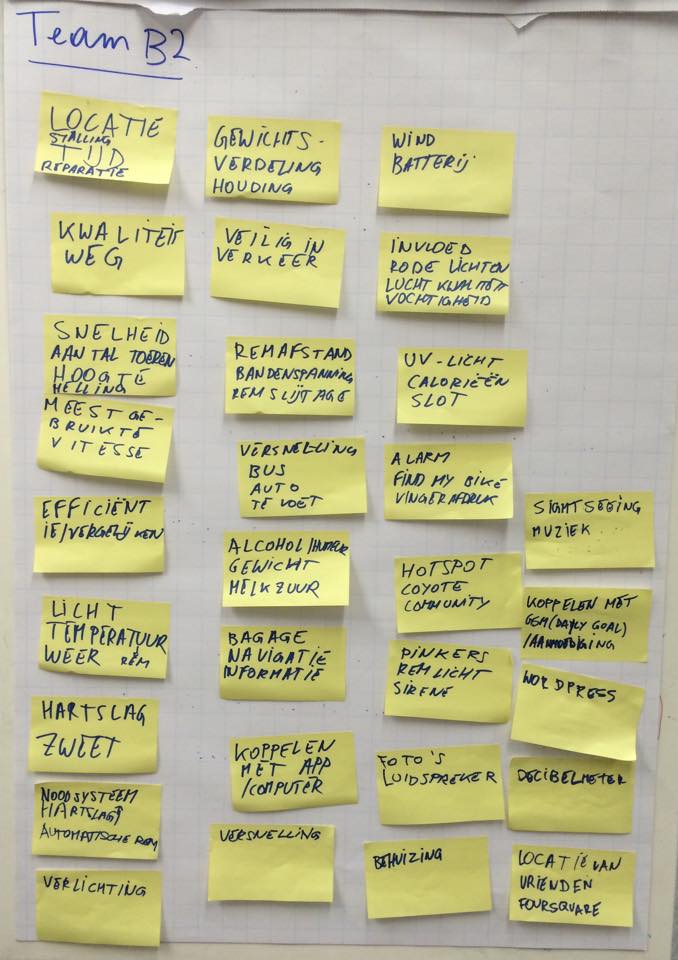
Van den Berghe Pieter 2 Elektrotechniek - Computerwetenschappen

Van Laere Nathan 2 Computerwetenschappen – Elektrotechniek

Informatie over het project kan gevonden worden via de volgende url’s:

1. Website: <http://goo.gl/zLV1cI>
2. Blog: <https://cw1b2.wordpress.com/page/2/>
3. Github: https://github.com/pietervdb/peno3bteam2

# Introductie

Afbeelding 1 toont het resultaat van de eerste brainstormsessie, waar het de bedoeling was om in eerste instantie na te gaan wat allemaal gemeten zou kunnen worden en hoe feedback aan de gebruiker gegeven zou kunnen worden. Appendix 3 bevat een mind map van alle elementen uit deze brainstorm, opgedeeld in verschillende categorieën.

Bij onze eerste brainstormsessie lag de nadruk te hard op het creëren van (navigatie)diensten voor de gebruiker, en te weinig op de eigenlijke opdracht: het vergaren en in beeld brengen van informatie. Hierdoor vielen al een heleboel ideeën weg, namelijk alles onder de categorie diensten *(zie Appendix 4).* Bij de categorie gezondheid was er weinig wat we echt zouden kunnen meten, behalve de hartslag. Bovendien zou het quasi onhaalbaar zijn om alles onder de categorie locatie en omstandigheden te meten. Om die reden werd besloten om (momenteel) enkel de voor ons relevante gegevens te meten: locatie, snelheid, hoogte, temperatuur en foto’s van de omgeving. De insteek van de toepassing, die de naam Bike-Lapse krijgt, is veranderd: tijdens de eerste brainstormsessie werd er over een dienst/augemented reality toepassing nagedacht. Na feedback van de assistenten veranderde dit: momenteel is het de bedoeling om een fotologboek van fietstochten bij te houden. Aan de gebruiker wordt ook feedback over zijn snelheid gegeven. Hierover meer in het volgend deel.

Afbeelding 1: Brainstormsessie

# Scenario’s

## User story 1

Joris is enorm gefocust op het bijhouden van allerhande informatie en doet dan ook aan lifelogging. Op aanraden van een vriend installeert hij Bike-Lapse. Allerhande informatie zoals locatie, snelheid en temperatuur wordt automatisch bijgehouden tijdens het fietsen en op een overzichtelijke manier gepresenteerd op de bijhorende webtoepassing. Een kaart laat zien welke weg hij juist genomen heeft, en op elk datapunt van de kaart vindt Joris de gegevens die op die locatie geregistreerd zijn Bovendien neemt een camera om de 10 seconden een foto. Zo kan hij met behulp van een time-lapse elke tocht aan vrienden laten zien.

## User story 2

Piet studeert burgerlijk ingenieur in de computerwetenschappen aan de KU Leuven en gaat elke dag met de fiets naar de les. Hij heeft de slechte gewoonte om te laat in de les aan te komen en professor Dutré kan daar niet meer mee lachen. Laatst kreeg hij zelfs een krijtje naar zijn hoofd geslingerd. Om dit probleem op te lossen maakt hij gebruik van Bike-Lapse. Deze toepassing houdt zijn gemiddelde snelheid bij en geeft, met behulp van een ledje op zijn stuur, aan wanneer hij sneller of trager rijdt dan gemiddeld. Het ledje wordt rood wanneer hij trager rijdt en groen wanneer hij sneller rijdt dan gemiddeld. Op die manier wordt hij aangemoedigd om steeds sneller naar de les te fietsem. Bovendien kan Piet via een webapplicatie verschillende fietstochten vergelijken om zo de voor hem efficiëntste weg te vinden. Sinds hij Bike-Lapse gebruikt, hoeft hij niet meer te vrezen voor de krijtjes van professor Dutré.

Met onze applicatie is de gebruiker in staat om allerlei informatie over zijn fietstochten bij te houden en te raadplegen op een webapplicatie. Wat de applicatie onderscheidt van andere, is dat een fotologboek van de fietstochten wordt bijgehouden. Zo kan de gebruiken met een time-lapse elke fietstocht herbeleven en tonen aan vrienden en familie. In deze time-lapse krijgt de gebruiker ook extra informatie over zijn trip, zoals bv: de temperatuur, de snelheid en de hoogte gemeten op dezelfde momenten als de foto’s worden getrokken. Tenslotte krijgt de gebruiker tijdens de fietstocht feedback over zijn snelheid tegenover zijn gemiddelde snelheid. Wanneer hij trager fietst dan zijn gemiddelde snelheid brand er een rood lampje, wanneer hij sneller rijdt een groen lampje.

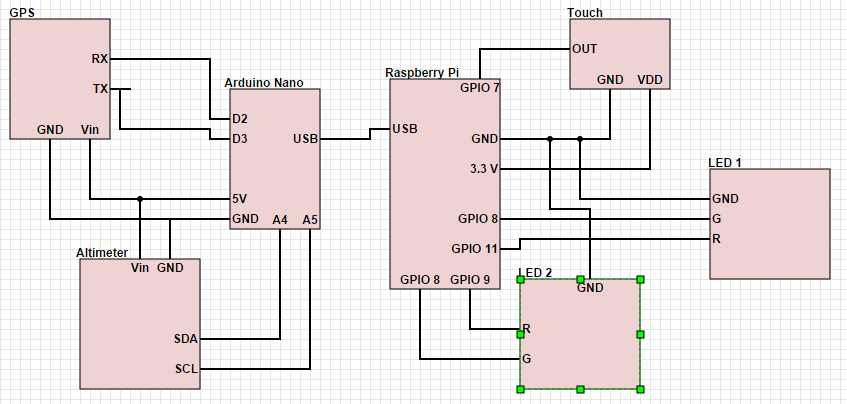
Om een duidelijk beeld te geven van de functionaliteiten van de applicatie worden er nu drie user stories beschreven.

# Architectuur

In deze sectie geven we een overzicht van de fysieke en de softwarearchitectuur van de applicatie. We beginnen met het bespreken van de fysieke architectuur. Dit is een overzicht van welke componenten op welke manier met elkaar interageren om de applicatie te laten werken.

## Fysieke architectuur

Voor dit project hebben we twee single-board computers ter beschikking gekregen, een Arduino Nano en een raspberry Pi. Voor een beschrijving van deze computers, zie sectie 5. In figuur X is er een overzicht van welke pinnen gebruikt worden. De Arduino wordt aangesloten op de Raspberry Pi via een USB-kabel. Op de Arduino zelf worden de gps-sensor en de barometer aangesleten. Deze worden beiden verbonden via 2 pinnen die de data doorsturen. Ook worden deze aan de grond en aan de 5V-pin van de Arduino verbonden. De Arduino leest data van de sensors en stuurt deze door naar de Raspberry Pi. Op de raspberry Pi zitten een druksensor, twee driekleurige LED’jes en de cameramodule. Deze laatste staat op de CSI-poort van de Raspberry Pi. De LED’jes zijn verbonden via drie digitale pinnen, die tevens voor stroomtoevoer zorgen. Elke pin vertegenwoordigt een ander kleurtje van de LED’jes. De drie mogelijke kleuren zijn blauw, rood en groen, maar de blauwe component van beide LED’jes zijn in deze opstelling niet in gebruik. De drukknop krijgt rechtstreeks van de Raspberry Pi (via de 3.3V-pin van de Raspberry Pi, zie figuur) de nodige stroom en is aangesloten aan de grond. De drukknop stuurt een signaal naar de Raspberry Pi door een verbinding via een digitale pin. De Raspberry Pi is zo geprogrammeerd dat wanneer de drukknop wordt ingedrukt, de LED op groen springt. Zolang de LED groen is, worden er om de vijf seconden foto’s getrokken door de camera. Het tweede LED’je werkt op een analoge manier als beschreven voor de eerste LED. De Raspberry Pi vergelijkt de huidige snelheid met een gemiddelde snelheid, die wordt bijgehouden en na elke trip wordt geupdatet. Wanneer de huidige snelheid hoger ligt dan de gemiddelde wordt de LED groen, anders wordt hij rood. Wanneer de drukknop opnieuw wordt ingedrukt, wordt de LED rood en stopt de camera met foto’s te trekken. De Raspberry Pi stuurt de getrokken foto’s en de verkregen data van de barometer en de gps-sensor naar de server, waarna deze kunnen weergegeven worden op de site.



Figuur 1: Elektrisch schema

Er zijn verschillende redenen waarom er voor deze architectuur is gekozen. In het begin waren de LED en de drukknop beiden ook aan de arduino verbonden. Wanneer dan de GPS-sensor en de altimeter werden bijgevoegd, leverde dit problemen op. Er trad namelijk interferentie op tussen de data. In de huidige architectuur is dit geen probleem meer. De tweede LED heeft informatie nodig die wordt opgeslagen op de Raspberry Pi. Daarom bestaat er geen twijfel dat deze LED op de Raspberry Pi moet aangesloten worden.

# Softwarearchitectuur

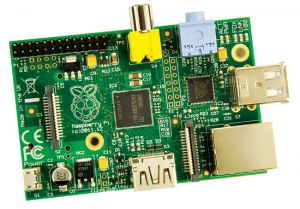
Bespreek, afzonderlijk, de fysieke en softwarearchitectuur van je applicatie.

Wat zijn de voor- en nadelen van de voorgestelde architectuur?

Gebruik duidelijk schema's die jullie architectuur duidelijker maken

# Technologie

## Raspberry Pi

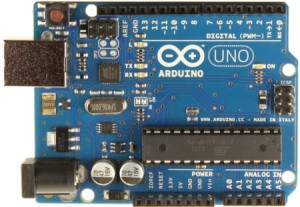


Figuur : Raspberry Pi

De Raspberry Pi is een goedkope singleboardcomputer. Het heeft de grootte van een kredietkaart, maar kan enorm makkelijk tot een computer omgebouwd worden, gewoon een scherm, muis en toetsenbord inpluggen. Binnen ons project is dit het voornaamste voordeel. Zijn grootte maakt het mogelijk om een computer te laten draaien tijdens het fietsen, die de fietser niet stoort. De Raspberry Pi heeft vele aansluitmogelijkheden: niet alleen de “basis” aansluitingen zoals USB, HDMI en Ethernet zijn aanwezig, men kan ook een camera en enorm veel andere sensors aansluiten, meestal via de digitale pinnen op het board. Ook qua software zijn de mogelijkheden legio: meer dan tien verschillende besturingssystemen zijn te gebruiken. Naast deze vele voordelen, heeft de Raspberr y Pi ook enkele nadelen: het gebruik vraagt wat kennis van Linux, de rekenkracht is niet enorm en er is geen ingebouwde Wi-Fi.

Tijdens het project hebben we toch verschillende keren problemen gehad met de Pi. Doordat we zoveel foto’s trekken, en dus opslaan op de Raspberry Pi tijden het fietsen, is er enorm veel geheugen nodig. De Raspberry Pi heeft hier nier genoeg geheugen voor. Om dit probleem op te lossen comprimeren we de foto’s direct wanneer ze getrokken worden.

## Arduino

Arduino is een opensource-computerplatform. Met Arduino is het mogelijk apparaten en objecten te creëren die reageren op hun omgeving door middel van digitale en analoge inputsignalen. Input kan bijvoorbeeld worden gegenereerd door schakelaars, lichtsensoren, bewegingsmeters, afstandsmeters, temperatuursensoren etc. Outputsignalen kunnen bijvoorbeeld motoren, lampjes, pompjes en beeldschermen aansturen, maar ook input genereren voor een andere Arduinomodule. [2] Die beschikbaarheid van enorm veel aansluitmogelijkheden is een van de grote voordelen van de Arduino. Qua software is het ook toegankelijker dan een Pi: de Arduino kan bestuurd worden via Windows, OS X en Linux. De Arduino is geen computer zoals de Pi: er is geen beeldscherm en internetaansluiting (doch beschikbaar via shields) en er draait niet echt een besturingssysteem op. De Arduino wordt namelijk bestuurd en geprogrammeerd vanaf een andere computer. .

Figuur : Arduino

De problemen die wij hebben ondervonden tijdens het werken met de Arduino bestond er vooral in dat er interferentie bestaat tussen de data van de verschillende sensors. Eerst was onze drukknop aangesloten op de Arduino, samen met de gps-sensor. Dit hebben we veranderd, omdat de data dan niet meer op een efficiënt manier geïnterpreteerd kan worden. Tussen de barometer en de gps-sensor bestaat er ook interferentie, maar we zijn erin geslaagd

## JavaScript

JavaScript wordt gebruikt in combinatie met HTML en CSS om websites te bouwen. JavaScript maakt webpagina’s interactief en dynamisch. Het grote voordeel van JavaScript is dat elke browser het ondersteunt en iedereen het gebruikt, aangezien er weinig alternatieven zijn. De alternatieven die bestaan werken via een compiler in JavaScript. Op internet vindt men vele tutorials en voorbeelden waardoor het relatief gemakkelijk is om met JavaScript te leren werken. Een nadeel is dat een functionaliteit vrij veel code vergt.

PROBLEMEN DIE TEGENGEKOMEN ZIJN …

## jQuery

jQuery is een bibliotheek voor JavaScript die browseronafhankelijk is. jQuery zorgt ervoor dat vele commando’s makkelijk opgeroepen kunnen worden, bijvoorbeeld het selecteren van HTML-elementen. Bovendien maakt jQuery code veel compacter en makkelijker te lezen. Een nadeel is dat het niet altijd even goed werkt wanneer er frameworks zoals Bootstrap worden gebruikt. Voor de niet-werkende commando’s is vaak wel een alternatief voorzien binnen het framework.

PROBLEMEN DIE TEGENGEKOMEN ZIJN

## JSON

JSON is een gestructureerd formaat voor het versturen en het opvragen van data via een server. Het voordeel van JSON is dat het goed leesbaar is, beter dan XML. Het is mogelijk om data van anderen op te vragen en te gebruiken wanneer iedereen dezelfde structuur gebruikt. Zo is men niet beperkt tot het gebruik van enkel onze eigen gegevens. Een ander voordeel van JSON is dat het onafhankelijk is van programmeertalen: het kan met andere woorden over verschillende talen heen gebruikt worden.

PROBLEMEN DIE TEGENGEKOMEN ZIJN

## Visualisatie Bibliotheken

Als visualisatie bibliotheek gebruiken we Google tezamen met een bibliotheek van JQuery: JQRangeSlider. Google biedt veel gemakkelijk toegankelijke en aanpasbare templates aan waar veel gebruik van is gemaakt. Een voorbeeld hiervan is Google Maps: dit wordt gebruikt om de coördinaten afkomstig van de GPS om te zetten in een route op een kaart. Een tweede voorbeeld is Google Charts: dit bevat een groot aantal grafieken, die makkelijke te bewerken zijn. Er moet enkel gelet worden op de structuur van de data die wordt gegeven. Met een beperkte hoeveelheid code is het mogelijk om uitgebreide grafieken weer te geven en alle data goed te visualiseren. Google biedt ook een mogelijkheid aan om in een grafiek te scrollen maar dit werkt niet op een touchscreen. JQuery kan hier een oplossing voor bieden in de vorm van een slider die wel op een touchscreen werkt: JQRangeSlider. Hierdoor kan de site ook op een smartphone gebruikt worden.

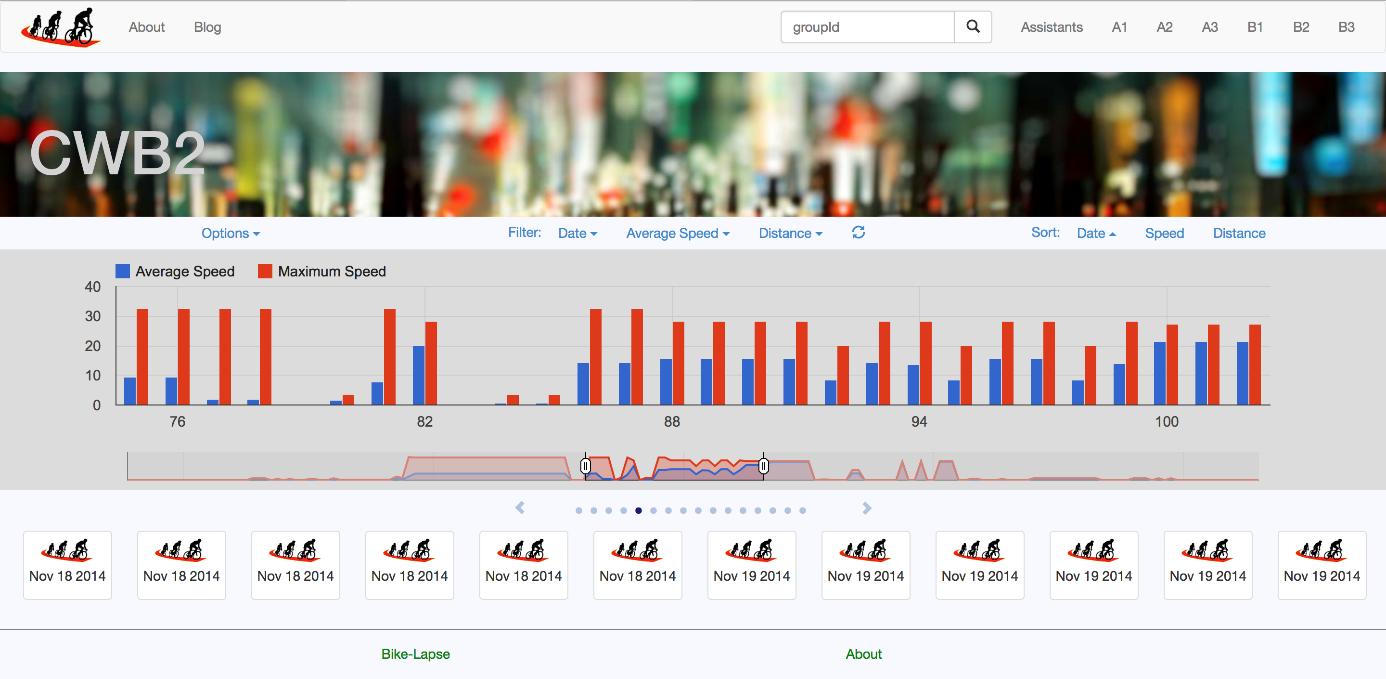
UITGEBREIDER, PROBLEMEN DIE TEGENGEKOMEN ZIJN

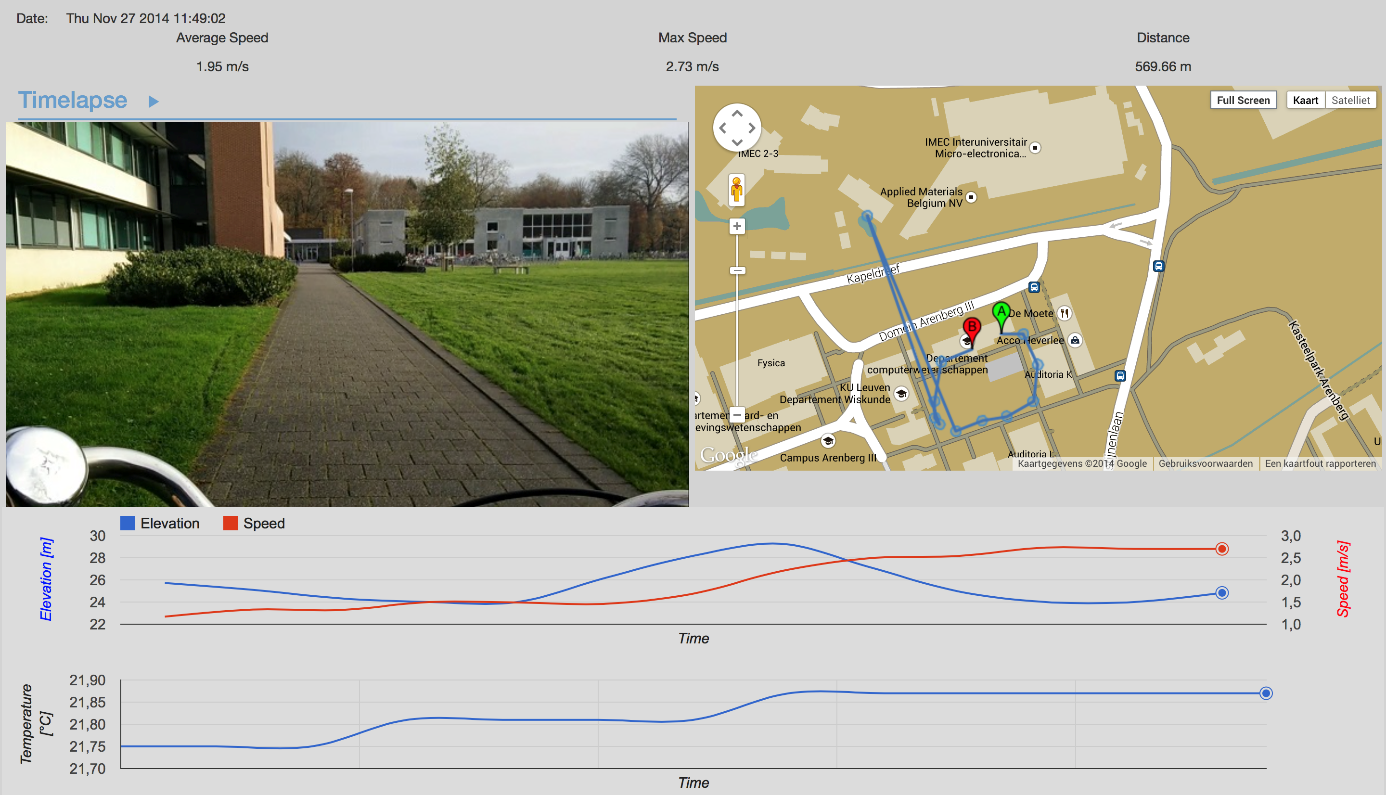
MOET NOG HALF PAPIER MINDER ZIJN

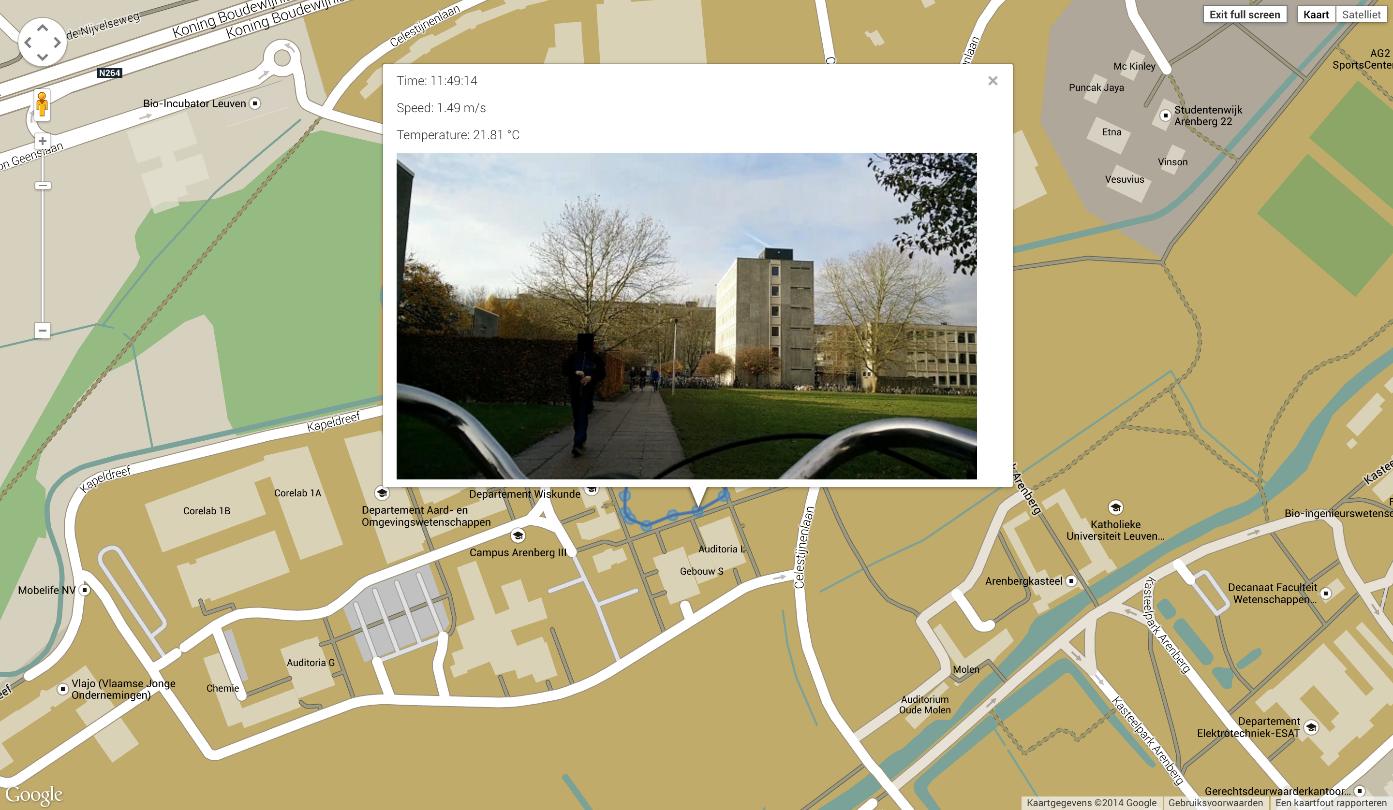
# Ontwikkelingsproces

Wanneer een gebruiker een trip start door op de drukknop te drukken, wordt alle data wordt verzameld met sensoren die aangesloten zijn op de Raspberry Pi of op de Arduino Nano (voor de exacte aansluitingen: zie deel 4. Architectuur). Wanneer de GPS met genoeg satellieten verbonden is, worden tijd, locatie en snelheid opgenomen. De cameramodule zal ook om de 10 seconden een foto nemen. Deze foto wordt vervolgens gecomprimeerd (van ± MB naar ± 15 kB) en gesorteerd in een map, specifiek voor een bepaalde trip. Met de altimeter wordt de temperatuur gemeten. Al deze informatie wordt om de 10 seconden opgeslagen. Wanneer een trip beëindigd wordt door de drukknop weer in te drukken, wacht de Raspberry Pi op een volgend commando. De gebruiker moet immers eerst de Pi aansluiten op internet alvorens al deze data naar de server gestuurd kan worden. Wanneer de gebruiker meent dat er een internetverbinding is, drukt hij iets langer dan 5 seconden op de drukknop. De Pi zal dan weten dat de informatie doorgestuurd mag worden.

Op de site wordt de doorgestuurde informatie op verschillende manieren weergegeven. Allereerst kan men kiezen van welke groep (A1, B1 etc.) men de trips wil bekijken. Wanneer men op een groep klikt, komt men op een overzichtspagina. Hier krijgt men een grafiek met de gemiddelde en de maximum snelheid van de trips. Onder deze grafiek worden de trips chronologisch weergegeven, maar er kan ook gesorteerd worden op maximum snelheid. Via filters kunnen bepaalde trips makkelijker gevonden worden, zoals alle trips in november 2014. Zie afbeelding \*\*\* voor een voorbeeld van zo’n overzichtspagina.



Wanneer men op een trip klikt, krijgt men daaronder een nieuw venster met informatie over die trip (zie afbeelding\*\*\*). Snelheid, hoogte (afkomstig van Google Maps) en temperatuur worden met behulp van grafieken weergegeven. Men kan gemakkelijk de time-lapse van de trip starten en pauzeren. Op een kaartje ziet men de gevolgde weg. Wanneer men inzoomt op een datapunt en er op klikt, krijgt men de foto en de informatie op dat punt te zien, zoals weergegeven in afbeelding\*\*\*.



Voor de snelheid, hoogte en temperatuur hebben we gekozen voor grafieken omdat dit volgens ons een zeer overzichtelijke manier is om deze gegevens te bekijken en te interpreteren. Mensen zijn immers vaker geïnteresseerd in het verloop van een waarde dan in de waarde op een specifiek punt (zoals een tabel het zou weergeven). Indien een gebruiker toch geïnteresseerd zou zijn in een waarde op een specifieke plaats, kan hij altijd via het kaart op een datapunt klikken om daar de gegevens te bekijken. De kaart lijkt ons een logische keuze voor het weergeven van de gevolgde route. Foto’s worden zowel weergegeven op de bijhorende locatie op de kaart als via een time-lapse.

# Vakintegratie

In eerste instantie sluit het project aan bij het vak *Methodiek van de informatica (Informatie)* uit het tweede semester van het eerste jaar, aangezien daar de basisbeginselen van programmeren werden aangeleerd. Bovendien wordt alles met betrekking tot de Raspberry Pi in Python geschreven, de taal die in dat vak werd gebruikt. Verder komen ook enkele belangrijke principes uit de vakken horend bij *Wiskunde* aan bod voor de snelheids- en versnellingsbepaling en voor de statistieken die bij de gegevens horen.

# Besluit

Tijdens dit project leerden we o.a. hoe we moeten werken met microcomputers (Raspberry Pi, Arduino Nano), hoe we sensoren op deze microcomputers aansluiten en aansturen, hoe we de data van die sensoren kunnen extraheren en visualiseren op een webapplicatie enzovoort. Ook de beginselen van HTML, CSS, JavaScript en JQuery werden aangeleerd via tutorials en vervolgens gebruikt in onze toepassing. Verder is ook het gebruik GitHub en WebStorm uitgelegd en geïmplementeerd.

De kans is groot dat velen van ons GitHub in de toekomst nog zullen gebruiken. Het blijkt de standaard te zijn voor het delen van en samenwerken aan code. Ook WebStorm heeft zijn nut bewezen. Dit programma zorgt voor heel wat meer overzicht in de code dan bv. Kladblok. Dat sommigen onder ons nu beter overweg kunnen met HTML, CSS of JavaScript zal bovendien zeker nog van toepassing komen in het vervolg van onze studie.

Wat de technische introductiesessies betreft: deze waren zeker handig om onszelf vertrouwbaar te maken met de Raspberry Pi en de Arduino. Tijdens deze sessies konden we hulp vragen aan de assistenten, en die hulp was zeker in het begin nog nodig. Ook de indeling in ontwerpfasen was logisch en nuttig. Het moeilijkere punt daar was om nieuwe user stories te blijven verzinnen. De tussentijdse demo zorgde voor feedback en nieuwe input, en is zeker aangewezen: waar nodig, kan immers bijgestuurd worden. Het zou wel aangenamer zijn mocht de demo iets vroeger georganiseerd worden, zodat er zeker tijd genoeg is om met alle tips rekening te kunnen houden.

De verhouding tussen de focus op ontwerpen, documenteren en implementeren zat goed. Er was niet één van deze drie delen die onevenwichtig veel aandacht vroeg. De opdracht zelf was ook ruim genoeg naar onze mening. We konden zelf kiezen welke sensoren we gebruikten, hoe we alles aan elkaar verbonden, hoe we de website ontwierpen, waarop we de gebruiker feedback gingen geven enzovoort.

Het gebruik van de Pi viel zeer goed mee. Eens je ermee weg bent, is er niet veel “moeilijks” meer aan. De vele aansluitmogelijkheden maken dit tot het geschikte apparaat voor een opdracht als deze. De Pi is twee keer gecrasht, maar dit was geen probleem aangezien we een back-up van alle code hadden.

Een verandering die we zelf zouden doorvoeren, is het verlaten van het tussentijds verslag. We waren nog maar net écht bezig met de opdracht en het zou gemakkelijker zijn als we ons toen volledig hierop hadden kunnen concentreren. Het was ook zo dat het seminarie over rapporteringstechnieken (ongeveer een week) viel nà het indienen van het tussentijds verslag, maar dit is waarschijnlijk niet iets dat jullie in de hand hebben…

# Appendixen

## Appendix 1: Geleverde werk

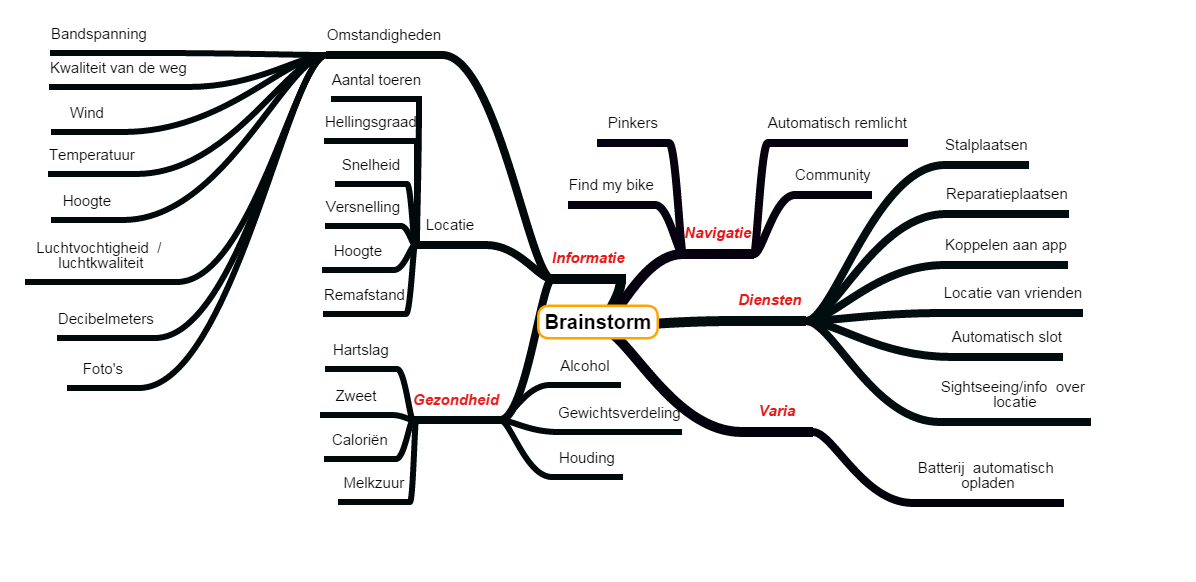
(In tabelvorm) Per teamlid (kolom) een nauwkeurige indicatie hoe lang (in uren) er aan elke fase (rij) werd besteed.

Geef ook de totalen aan per teamlid (extra kolom) en per fase (extra rij).

## Appendix 2: Planning

Licht de taakverdeling toe door middel van een gedetailleerde Gantt-chart.

## Appendix 3: Mind map brainstorm



## Bronvermelding

<http://codeduino.com/tutorials/arduino-vs-raspberry-pi/>

<https://sites.google.com/site/bus237raspberrypi/customize>