

**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 Erik

Klerkx Joris

Begeleider(s)

Charleer Sven

De Croon Robin

Santos Jose Luis

ACADEMIEJAAR 2014-2015

# INHOUDSTAFEL

[INHOUDSTAFEL 2](#_Toc406167613)

[Lijst van afbeeldingen en tabellen 2](#_Toc406167614)

[1. Teaminformatie 3](#_Toc406167615)

[2. Introductie 3](#_Toc406167616)

[Brainstorm 3](#_Toc406167617)

[Productbeschrijving 4](#_Toc406167618)

[3. Scenario’s 4](#_Toc406167619)

[User story 1 4](#_Toc406167620)

[User story 2 4](#_Toc406167621)

[4. Architectuur 5](#_Toc406167622)

[Fysieke architectuur 5](#_Toc406167623)

[Softwarearchitectuur 6](#_Toc406167624)

[Raspberry Pi en Arduino Nano 6](#_Toc406167625)

[Site 7](#_Toc406167626)

[5. Technologie 8](#_Toc406167627)

[Raspberry Pi 8](#_Toc406167628)

[Arduino 9](#_Toc406167629)

[JavaScript 9](#_Toc406167630)

[jQuery 9](#_Toc406167631)

[JSON 10](#_Toc406167632)

[Visualisatiebibliotheken 10](#_Toc406167633)

[6. Ontwikkelingsproces 10](#_Toc406167634)

[7. Vakintegratie 13](#_Toc406167635)

[8. Besluit 13](#_Toc406167636)

[9. Appendices 14](#_Toc406167637)

[Appendix I: Geleverde werk 14](#_Toc406167638)

[Appendix II: Planning 14](#_Toc406167639)

[Appendix III: Mindmap brainstorm 15](#_Toc406167640)

[Appendix IV: Oplijsting brainstormsessie 15](#_Toc406167641)

[10. Bronvermelding 16](#_Toc406167642)

# Lijst van afbeeldingen en tabellen

[Afbeelding 1: brainstormsessie 3](file:///\\psf\Home\Desktop\Eindverslag.docx#_Toc406167025)

[Afbeelding 2: overzicht van de verbindingen tussen alle componenten 6](#_Toc406167026)

[Afbeelding 3: overzicht van de softwarematige werking van Raspberry Pi en Arduino 7](#_Toc406167027)

[Afbeelding 4: schematische weergave van de opbouw van de website 8](#_Toc406167028)

[Afbeelding 5: overzichtspagina 11](file:///\\psf\Home\Desktop\Eindverslag.docx#_Toc406167029)

[Afbeelding 6: informatie over een trip 12](file:///\\psf\Home\Desktop\Eindverslag.docx#_Toc406167030)

[Afbeelding 7: foto en informatie van een specifiek datapunt 12](file:///\\psf\Home\Desktop\Eindverslag.docx#_Toc406167031)

[Afbeelding 8: Gantt-chart 14](file:///\\psf\Home\Desktop\Eindverslag.docx#_Toc406167032)

[Afbeelding 9: mindmap van de brainstormsessie 15](file:///\\psf\Home\Desktop\Eindverslag.docx#_Toc406167033)

[Tabel 1: gegevens van de teamleden 3](#_Toc406162051)

[Tabel 2: geleverde werk per teamlid in uren 14](#_Toc406162052)

# Teaminformatie

Alle groepsleden volgen de bachelor in de ingenieurswetenschappen. Tabel 1 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

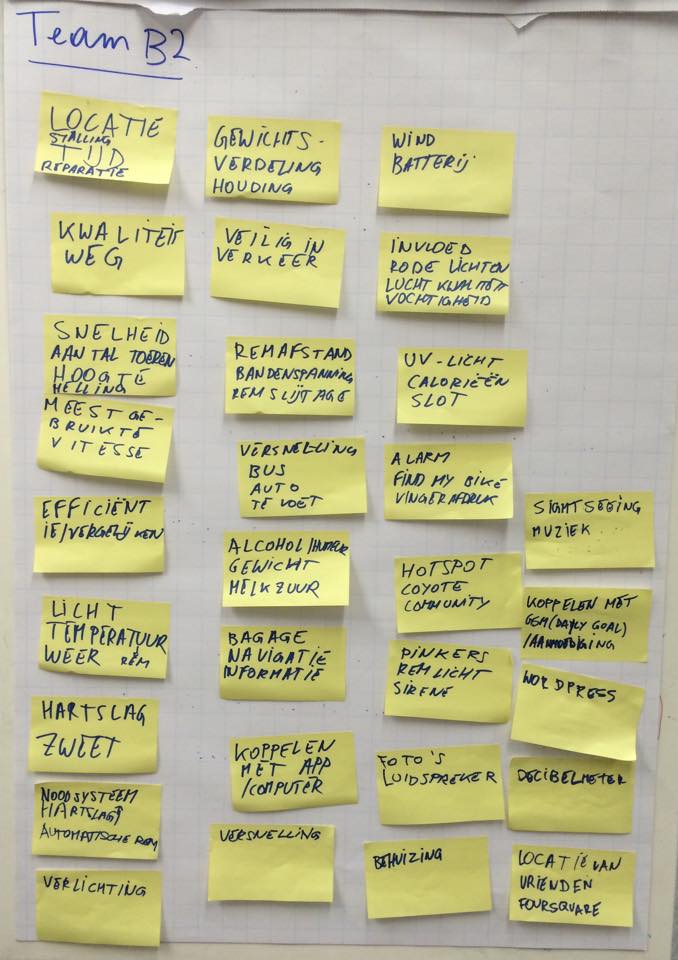
Tabel : gegevens van de teamleden

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

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

# Introductie

## Brainstorm

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 daarop mogelijk zou zijn. Appendix III bevat een mindmap van alle elementen uit deze brainstorm, opgedeeld in verschillende categorieën. Appendix IV geeft hiervan een oplijsting met uitleg.

Bij onze eerste brainstormsessie lag de nadruk te veel 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 IV).* Bij de categorie gezondheid was er weinig wat we echt zouden kunnen meten, behalve de hartslag. Bovendien zou het door tijdgebrek enerzijds en door beperking van het aantal sensoren anderzijds quasi onhaalbaar zijn om alles onder de categorie locatie en omstandigheden te meten. Om die reden werd besloten om enkel die gegevens te meten waar volgens ons gebruikers het meest geïnteresseerd in zijn: 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/augmented reality toepassing nagedacht. Dit was echter niet de opdracht en daarom is onze insteek veranderd naar een fotologboek van fietstochten met de hierboven vermelde gegevens.

Afbeelding 1: brainstormsessie

## Productbeschrijving

Met onze applicatie Bike-Lapse is een gebruiker in staat om informatie over zijn fietstochten bij te houden en te raadplegen op een website. Hiervoor moet hij ons apparaat, zijnde een combinatie van een Raspberry Pi, een Arduino Nano en enkele sensoren, op zijn fiets bevestigen. De gebruiker kan dan via een drukknop een trip starten. Tijdens zo’n trip worden verschillende gegevens (locatie, snelheid en temperatuur) bijgehouden. Ook wordt er om de 10 seconden een foto getrokken. De gebruiker kan bovendien via een drukknop zelf een foto trekken wanneer hij wil. Al deze informatie wordt opgeslagen en doorgestuurd naar onze website, wanneer de trip beëindigd wordt. Daar kan de gebruiker al deze informatie bekijken met behulp van een kaart en grafieken. Op deze kaart kan hij de gevolgde route bekijken en wanneer hij op een datapunt klikt, krijgt hij alle geregistreerde informatie op dat punt te zien. Daarenboven is het mogelijk met een time-lapse elke trip te herbeleven en te tonen aan vrienden en familie.

Tijdens een trip is er ook enige vorm van feedback: aan de hand van de groene of rode kleur van een led weet de gebruiker of hij op dat moment sneller of trager rijdt dan zijn gemiddelde snelheid over alle trips heen. Zo is het mogelijk om steeds het eigen record te verbreken. Daarnaast zal de led oranje worden wanneer er data verstuurd wordt naar de website.

# 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 fietsen. 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.

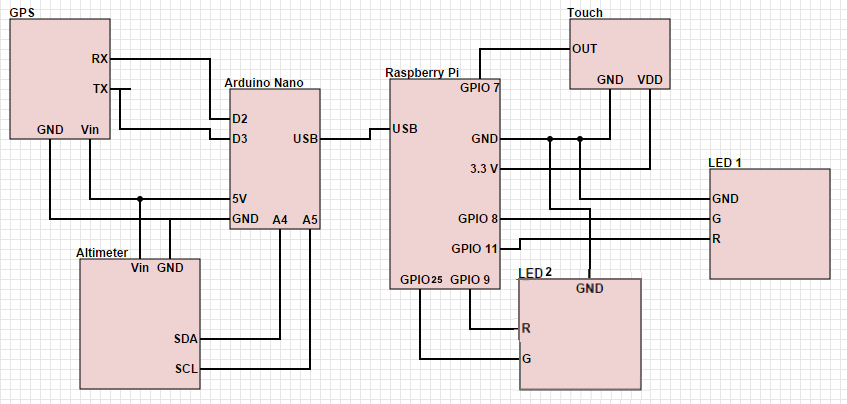
# Architectuur

In deze sectie geven we een overzicht van de fysieke en de softwarearchitectuur van de applicatie. Eerst volgt een bespreking van de fysieke architectuur. Dit is een overzicht van welke componenten op welke manier met elkaar interageren om de applicatie te laten werken. Daarna volgt een bespreking van de softwarearchitectuur van enerzijds de programma’s op de Arduino en de Raspberry Pi en van de site anderzijds.

## 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 deel 5: Technologie. Afbeelding 2 geeft een overzicht van de verbindingen tussen Raspberry Pi en Arduino en tussen de aangesloten componenten en deze twee apparaten zelf. Op dit overzicht is tevens te zien welke sensoren met welke pinnen verbonden worden.

De Arduino wordt aangesloten op de Raspberry Pi via een USB-kabel. Op de Arduino zelf worden de GPS-sensor en de altimeter aangesloten. Deze worden elk verbonden via twee 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 deze sensoren en stuurt deze informatie vervolgens door naar de Raspberry Pi. Op de Raspberry Pi zijn een druksensor, een cameramodule en twee driekleurige ledjes aangesloten. De camera hangt aan de CSI-poort van de Raspberry Pi. De ledjes zijn verbonden via drie digitale pinnen, die tevens voor stroomtoevoer zorgen. Elke pin vertegenwoordigt een ander kleurtje van de led. De drie mogelijke kleuren zijn blauw, rood en groen, maar de blauwe component van beide ledjes zijn in deze opstelling niet in gebruik, gezien we enkel de rood-groen combinaties gebruiken. De aanraakgevoelige drukknop krijgt via de 3.3V-pin van de Raspberry Pi de nodige stroom en is aangesloten op de grond. De drukknop stuurt een signaal naar de Raspberry Pi via een digitale pin. De Raspberry Pi is zo geprogrammeerd dat wanneer de drukknop wordt ingedrukt, een trip begint en de eerste led op groen springt. Wanneer deze led groen is, draait het Pythonprogramma op de Pi zoals beschreven in de productbeschrijving en de softwarearchitectuur. Het tweede ledje werkt op een analoge manier. De Raspberry Pi vergelijkt de huidige snelheid met een gemiddelde snelheid, die wordt bijgehouden en na elke trip wordt geüpdatet. Wanneer de huidige snelheid hoger is dan de gemiddelde kleurt de led groen, in het andere geval kleurt hij rood. Wanneer de drukknop opnieuw wordt ingedrukt, kleurt de eerste led weer rood en stopt de trip. De Raspberry Pi stuurt alle verkregen data naar de server, waarna deze kan weergegeven worden op de site.



Afbeelding : overzicht van de verbindingen tussen alle componenten

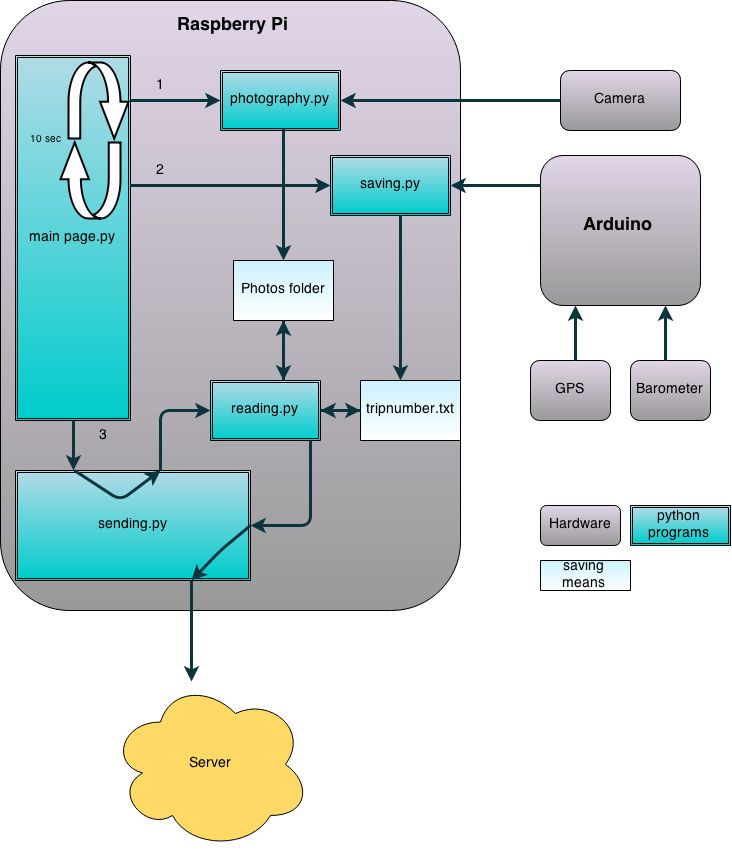
Er zijn verschillende redenen waarom er voor deze opstelling is gekozen. In het begin waren de leds en een drukknop beiden aan de Arduino verbonden. Dit is veranderd: we gebruiken nu een capacive touch druksensor die op de Pi dient aangesloten te worden. Anderzijds zijn beide leds nu aan de Raspberry aangesloten, aangezien deze allebei informatie nodig hebben afkomstig van de Pi. Dit zorgt er voor dat de Arduino enkel een doorstuurfunctie heeft. De controlerende functies worden door de Pi verzorgd.

## Softwarearchitectuur

### Raspberry Pi en Arduino Nano

Afbeelding 3 geeft een overzicht van de softwarematige werking van Raspberry Pi en Arduino. Voor de Raspberry Pi is de hoofdfunctie te vinden in *main page.py.* Daarin zit een while-lus die nagaat of er al dan niet een trip bezig is en die meerdere trips kan uploaden indien nodig. Tijdens een trip worden om de tien seconden de functies uit de files *photography.py* en *saving.py* aangeroepen (stap 1 en 2). Zij slaan data op in respectievelijk een *Photos folder* (foto’s) en een *tripnumber.txt* file (data van GPS en altimeter). Wanneer één of meerdere trips beëindigd zijn, kan de gebruiker deze trips uploaden met behulp van een langdurige aanraking op de druksensor (stap 3). Hierop roept de while-lus uit *main page.py* functies van *sending.py* op. Deze functies halen eerst data via *reading.py* op om die data vervolgens naar de server te sturen. De Arduino gebruikt één while-lus om data van de aangesloten sensoren door te sturen wanneer dit gevraagd wordt door functies uit *saving.py*.

Een nadeel van onze code is dat we geen gebruik van classes hebben gemaakt. Dit zou de code namelijk overzichtelijker en beter aanpasbaar maken. Er is echter niemand van onze groep die classes al goed genoeg beheerst, en we legden onze focus liever op de werking dan op de vorm.

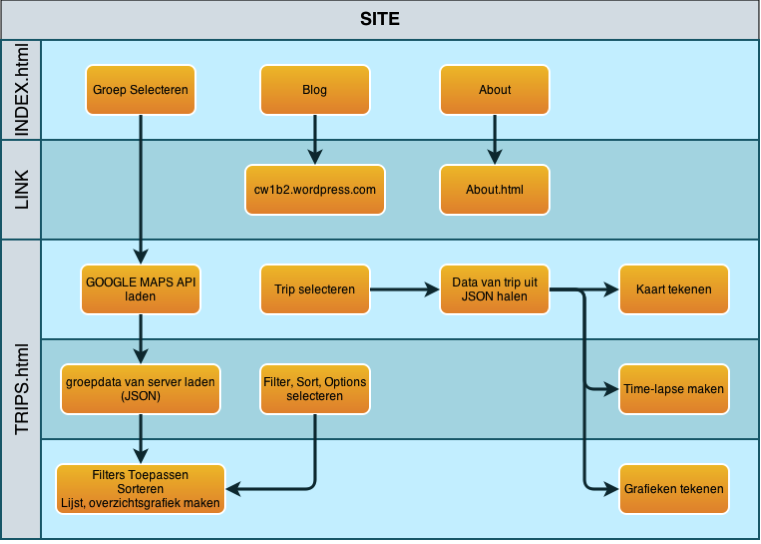


Afbeelding : overzicht van de softwarematige werking van Raspberry Pi en Arduino

### Site

De site bestaat uit drie pagina’s: *index*, *trips* en *about*. De globale werking van de site wordt afgebeeld in Afbeelding 4. Op de hoofdpagina (*index*) is het mogelijk een groep te selecteren, naar de blog te gaan of naar een pagina met algemene informatie (*about*). Deze functies zijn ook beschikbaar in *trips*. Wanneer men een groep selecteert of zelf ingeeft, wordt men naar *trips* geleid. Hier wordt de data van de desbetreffende groep geladen als een JSON en vervolgens weergegeven en gesorteerd na het toepassen van enkele filters. Het is ook mogelijk eigen filters toe te passen en de trips te sorteren op datum, snelheid of afgelegde weg. Wanneer men een trip selecteert, wordt deze trip uit de JSON gehaald. De data van de trip wordt dan gevisualiseerd d.m.v. een kaart met de route, een time-lapse en enkele grafieken *(zie ook productbeschrijving)*. Voor de kaart en de grafieken maken we gebruik van bibliotheken van Google: Google Maps API en Google Visualisation Library.

Voor de algemene weergave van de site gebruiken we een framework, namelijk Bootstrap. Dit is een mobile-first framework, waardoor onze site ook gemakkelijk te gebruiken is op smartphones. Ook is het met dit framework vrij eenvoudig om de lay-out te programmeren. We gebruiken daarenboven vier plug-ins. De eerste is een functie voor het formatteren van een *Date-object*. De tweede is een functie die een laadicoon genereert. Plug-in drie voegt een knop toe aan de kaart om deze in full screen weer te geven. De laatste plug-in is er één voor jQuery die ervoor zorgt dat een door ons gekozen functie opgeroepen wordt als alle afbeeldingen geladen zijn.

Voor de bronvermelding van deze plug-ins verwijzen we naar deel 10: Bronvermelding.

Afbeelding : schematische weergave van de opbouw van de website

# Technologie

## Raspberry Pi

De Raspberry Pi is een goedkope singleboardcomputer ter grootte van een kredietkaart waarop een scherm, een muis en een toetsenbord kunnen worden aangesloten. Binnen ons project is dit laatste het voornaamste voordeel. Zijn grootte maakt het mogelijk om hem te gebruiken tijdens het fietsen, zonder de fietser te storen. De Pi heeft vele aansluitmogelijkheden: naast de reeds genoemde aansluitmogelijkheden (HDMI, USB), kan men er ook een camera en enorm veel andere sensoren op aansluiten, meestal via de digitale pinnen op het board. Naast deze vele voordelen, heeft de Raspberry 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 verschillende keren problemen gehad met de Pi. Doordat er zoveel foto’s getrokken worden, en dus op de Pi opgeslagen moeten worden, is er vrij veel opslaggeheugen nodig. Om dit probleem op te lossen comprimeren we de foto’s direct nadat ze getrokken zijn. Een ander probleem was dat we de GPS-sensor niet konden aansluiten op de Pi. We moesten daarvoor in de boot van de Pi iets aanpassen en toen we dat gedaan hebben, is de Pi gecrasht. Daarom hebben we nadien de GPS-sensor op de Arduino aangesloten.

## Arduino

De Arduino is een microcontroller die gebruikt kan worden om allerlei interactieve toepassingen te maken. De beschikbaarheid van enorm veel aansluitmogelijkheden is een van de grote voordelen van de Arduino. Een nadeel van de Arduino is dat het soms vrij lastig is om meerdere sensoren tegelijk werkend te krijgen. Dit was ook een probleem dat wij hebben ondervonden tijdens het werken met de Arduino. 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 efficiënte manier geïnterpreteerd kon worden. Tussen de altimeter en de GPS-sensor bestaat er ook interferentie, maar we zijn erin geslaagd de temperatuur en de informatie afkomstig van de GPS-sensor foutloos te extraheren.

## JavaScript

JavaScript wordt gebruikt in combinatie met HTML en CSS om websites te bouwen en 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. Een mogelijk alternatief is PHP. Dit is echter een server-side scripting taal die bovendien moeilijker is om te leren bij een matige voorkennis programmeren. 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.

Wij gebruiken JavaScript om onze website interactief te maken: grafieken, knoppen en kaarten zijn er mee toegevoegd. JavaScript zorgde niet meteen voor concrete problemen.

## 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.

Wij gebruiken JQuery om de code van onze website overzichtelijker en korter te maken. We hadden een tijd lang niet door dat de functie voor het verwijderen van elementen ervan uitgaat dat die elementen zelf bestaan. Dit hebben we opgelost door een functie te gebruiken die in één keer een bepaalde groep leegmaakt.

## 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.

Wij gebruiken JSON om ervoor te zorgen dat onze data gestructureerd naar de server wordt verstuurd. Bij JSON hadden we op een bepaald moment problemen omdat er te veel data op de server stond.

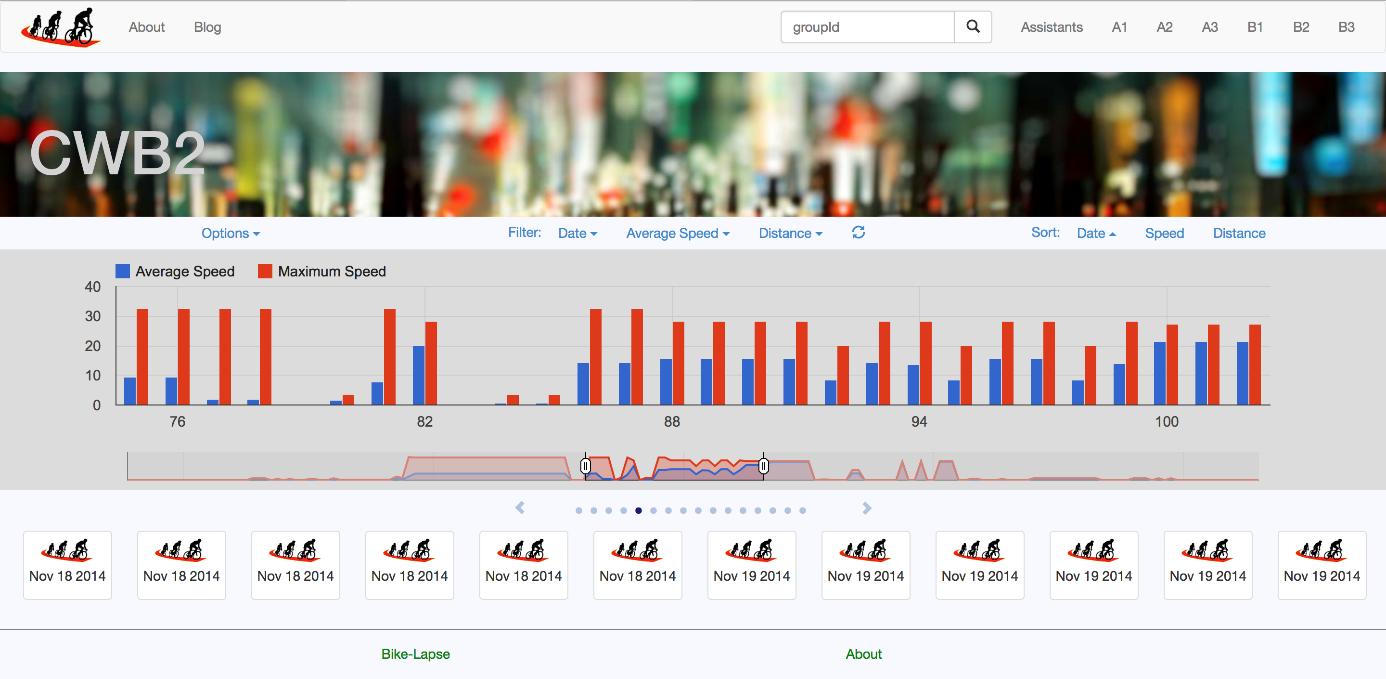
## Visualisatiebibliotheken

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 gemakkelijk 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.

Wij gebruiken bovengenoemde zaken voor de toepassingen die genoemd werden. Problemen met visualisatiebibliotheken hebben we eigenlijk niet gehad.

# Ontwikkelingsproces

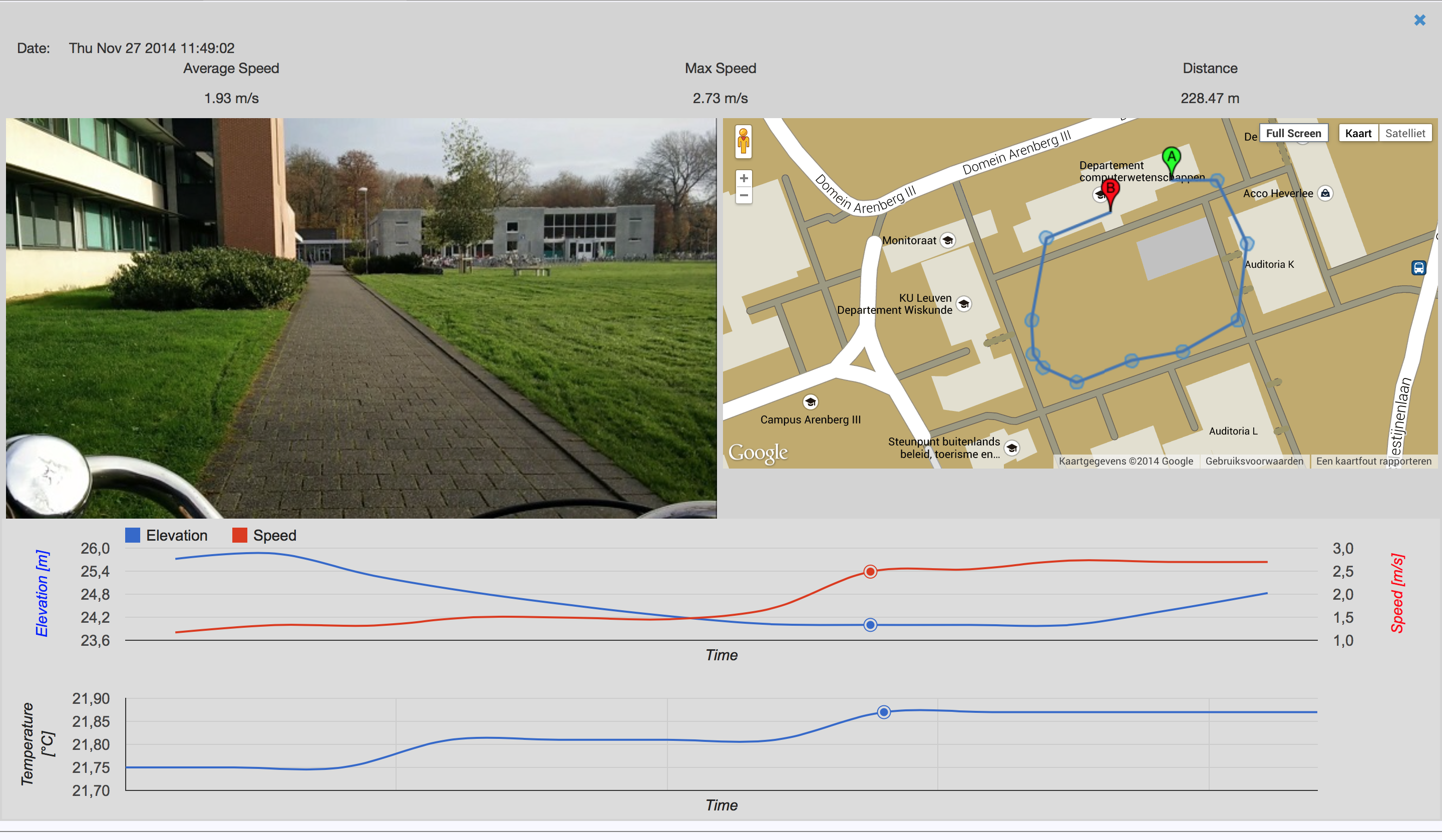
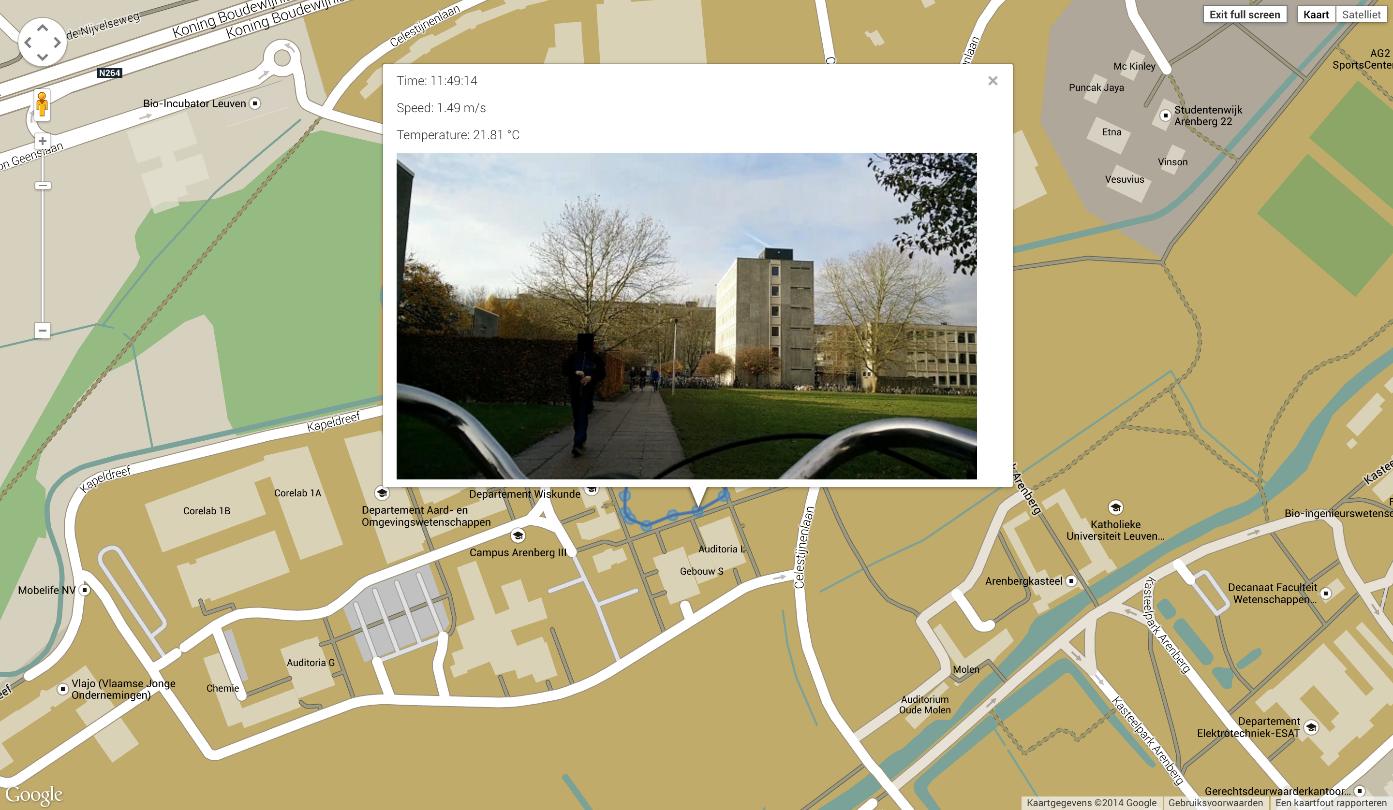
Wanneer een gebruiker een trip start door op de drukknop te drukken, wordt alle data verzameld met sensoren die aangesloten zijn op de Raspberry Pi of op de Arduino Nano *(voor de exacte aansluitingen: zie deel 4. Architectuur)*. Wanneer de GPS met genoeg satellieten verbonden is, worden tijd, locatie en snelheid opgenomen. De cameramodule zal ook om de 10 seconden een foto nemen. Deze foto wordt vervolgens gecomprimeerd (van ± 4 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 snelheden 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 5 voor een voorbeeld van zo’n overzichtspagina.

Afbeelding 5: overzichtspagina

Wanneer men op een trip klikt, krijgt men daaronder een nieuw venster met informatie over die trip *(zie Afbeelding 6).* 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 7.

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 de 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.



Afbeelding 6: informatie over een trip

Afbeelding 7: foto en informatie van een specifiek datapunt

# 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 wordt gebruikt. Kennis van *Elektrische Netwerken (Materie en Energie)* is nodig voor de aansluiting en verbindingen tussen alle componenten.

Verder komen ook enkele belangrijke principes uit de vakken horend bij *Wiskunde* aan bod 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 ook 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…

# Appendices

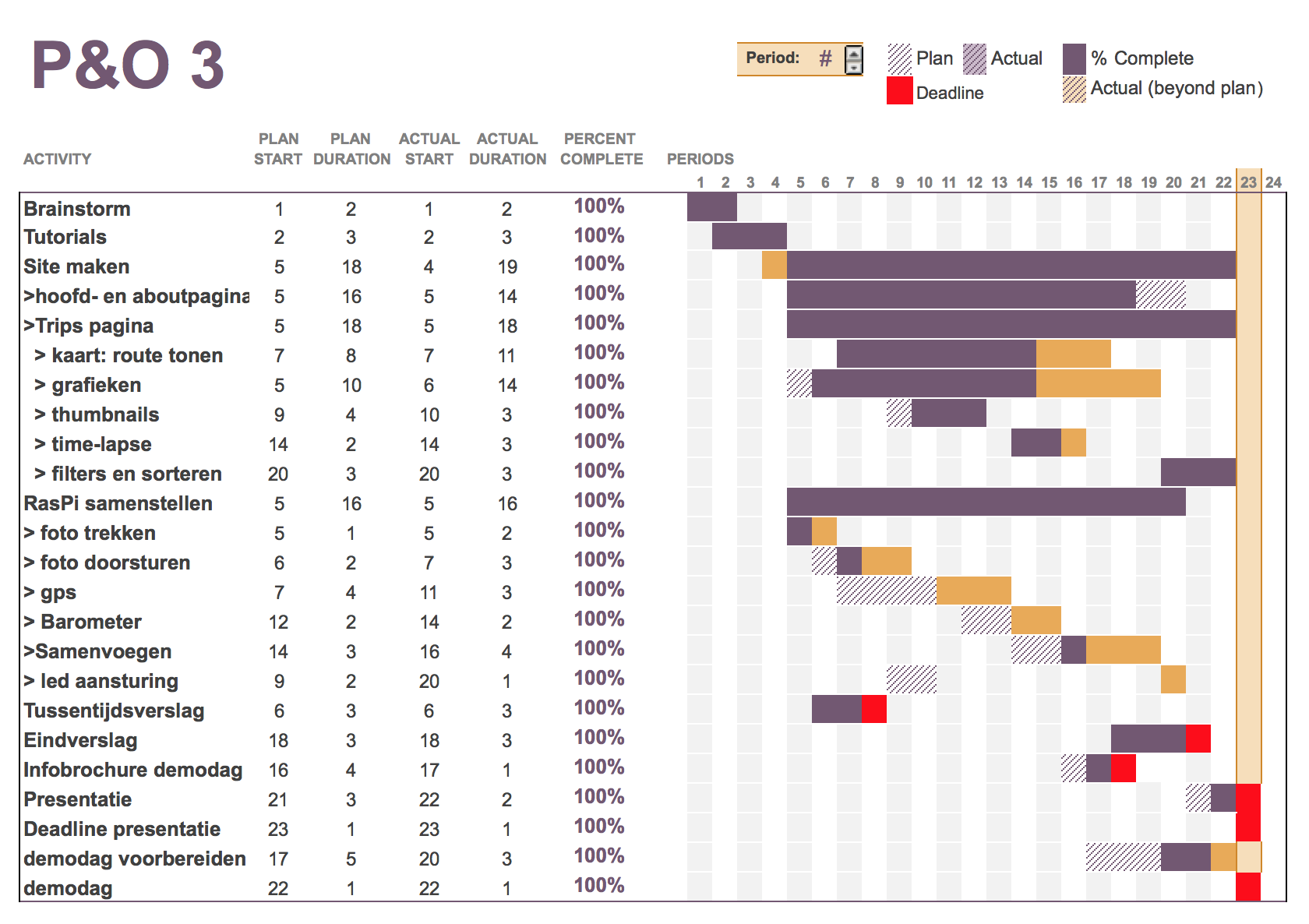
## Appendix I: Geleverde werk

Tabel 2 vat het geleverde werk per teamlid in uren samen.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Andreas | Bernd | Nathan | Pieter | Sander | Shoera | Totaal |
| introductiesessies + brainstorm | 14 | 14 | 14 | 14 | 14 | 14 | 84 |
| user stories | 1 | 1 | 1 | 1 | 1 | 1 | 6 |
| tussentijds verslag | 10 | 0,5 |  | 1 | 2 | 4 | 17,5 |
| website |  | 47 |  | 47 | 3 |  | 97 |
| Raspberry Pi | 23 |  | 50 |  | 47 | 28 | 148 |
| GitHub tutorial | 2 | 2 | 2 | 2 | 2 | 2 | 12 |
| productbeschrijving | 2 |  |  |  |  |  | 2 |
| poster |  |  |  |  |  | 3 | 3 |
| eindverslag | 14 | 2 | 1 | 2 | 1 | 14 | 34 |
| presentatie | 4 | 3,5 | 2 | 3 |  | 4 | 16,5 |
| Totaal | 70 | 70 | 70 | 70 | 70 | 70 | 420 |

Tabel : geleverde werk per teamlid in uren

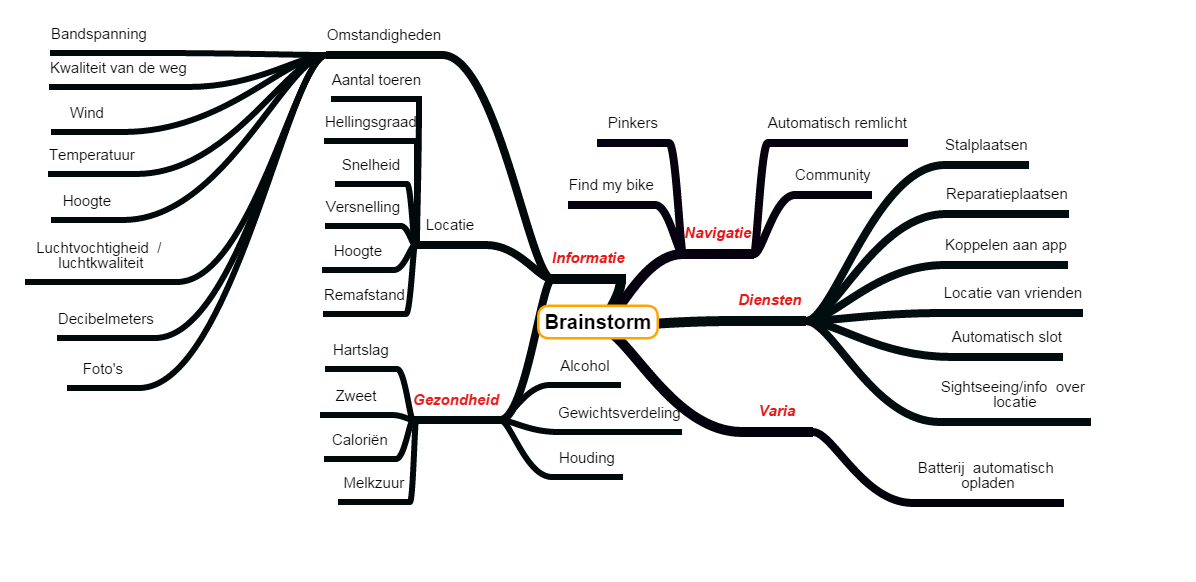
## Appendix II: Planning

Afbeelding 8 toont de Gantt-chart.

Afbeelding 8: Gantt-chart

## Appendix III: Mindmap brainstorm

Afbeelding 9 toont de mindmap van de brainstormsessie.



Afbeelding 9: mindmap van de brainstormsessie

## Appendix IV: Oplijsting brainstormsessie

**Locatie en omstandigheden**

Hoogte

Helling

Snelheid

Versnelling

Aantal toeren

Meest gebruikte versnelling

Temperatuur

Wind

Luchtvochtigheid

Luchtkwaliteit

Kwaliteit van de weg

UV-licht

Decibelmeter

Bandenstapping

Remafstand

**Foto’s**: *er* *worden foto’s van de omgeving van de gebruiker doorgestuurd*

**Gezondheid**

Hartslag

Zweet

Calorieën

Alcohol

**Melkzuur**: *hoeveelheid melkzuur in de spieren van de gebruiker*

**Gewichtsverdeling**: *de verdeling van het gewicht van de gebruiker op het frame van de fiets*

**Houding**: *informatie over de houding van de gebruiker*

**Navigatie**

**Find my Bike:** *laatste locatie van de fiets weergeven*

**Bus/auto/te voet:** *een vergelijking van de reistijd per fiets met die per bus, per de auto of te voet (mogelijk voor en na een fietstocht)*

**Pinkers:** *mogelijkheid om pinkers te gebruiken met knoppen links en rechts op het stuur*

**Remlicht:** *automatisch remlicht*

**Coyote/community:** *informatie van andere gebruikers over opstopping etc.*

**Diensten**

**Stalling:** *de mogelijkheid om de dichtstbijzijnde fietsenstallingen weer te geven*

**Reparatie:** *de mogelijkheid om de dichtstbijzijnde reparatielocaties weer te geven*

**Noodsysteem hartslag:** *wanneer de hartslag van de gebruiker ongezond hoog wordt, wordt hij hiervan verwittigd. Indien de hartslag van de gebruiker stopt, worden de hulpdiensten hiervan verwittigd, waarbij bovendien de locatie van de gebruiker doorgegeven wordt*

**Kopellen met een app:** *in een app kan de gebruiker allerhande informatie op een overzichtelijke manier raadplegen*

**Foursquare/Locatie van vrienden:** *op een kaart ziet de gebruiker de locatie van vrienden*

**Slot:** *een automatisch slot op de fiets wanneer de gebruiker de fietstocht beëindigt (eventueel inclusief vingerafdrukbeveiliging)*

**Sightseeing:** *de gebruiker krijgt informatie over bezienswaardigheden nabij hem*

**Hotspot:** *de gebruiker wordt verwittigd wanneer er een internet hotspot nabij hem is*

Muziek/luidspreker

**Varia**

**Batterij:** *mogelijkheid om de batterij van de Raspberry Pi op te laden met een dynamo*

**Invloed rode lichten:** nagaan hoe lang bepaalde lichten gemiddeld op rood staan, en aan de hand hiervan de invloed op de reistijd berekenen

# Bronvermelding

(sd). Opgeroepen op december 5, 2014, van https://sites.google.com/site/bus237raspberrypi/customize

(sd). Opgeroepen op december 5, 2014, van http://cdnjs.cloudflare.com/ajax/libs/jquery.waitforimages/1.5.0/jquery.waitforimages.min.js

(sd). Opgeroepen op december` 9, 2014, van bit-tech: http://www.bit-tech.net/modding/2013/03/22/raspberry-pi-case-competition-update/1

(sd). Opgeroepen op december 9, 2014, van Arduino: http://arduino.cc/en/uploads/Main/ArduinoNanoFront\_3\_sm.jpg

*Arduino (Computerplatform)*. (sd). Opgeroepen op Oktober 21, 2014, van Wikipedia: http://nl.wikipedia.org/wiki/Arduino\_(computerplatform)

*Arduino vs Raspberry Pi Comparison*. (sd). Opgeroepen op december 5, 2014, van Codeduino: http://codeduino.com/tutorials/arduino-vs-raspberry-pi/

Bell, C. (sd). *Full screen Google Maps*. Opgeroepen op december 5, 2014, van http://www.doogal.co.uk/FullScreen.php

Gnass, F. (2014, april 24). *spin.js*. Opgeroepen op december 5, 2014, van http://fgnass.github.io/spin.js

Levithan, S. (2007, juni 9). *JavaScript Date Format*. Opgeroepen op december 5, 2014, van http://blog.stevenlevithan.com/archives/date-time-format

*Raspberry Pi*. (sd). Opgeroepen op Oktober 21, 2014, van Wikipedia: http://nl.wikipedia.org/wiki/Raspberry\_Pi