kennistransfer en bedrijfsopleidingen

Bijlagenboek

Einddocumentatie V5 – Smart Emission 2

Auteurs: Thomas Geurts van Kessel
Stefan Knoet
Datum: 26-06-2020

has hogeschool

Inhoudsopgave

Bijlage 1: Plan van Aanpak

kennistransfer en bedrijfsopleidingen

Smart Emission 2

Toolkit visualisatie sensor data

Plan van Aanpak

hogeschool

HAS Kennistransfer en Bedrijfsopleidingen
Onderwijsboulevard 221
Postbus 90108
5200 MA ’s-Hertogenbosch
Telefoon: (088) 890 36 37

Documenttitel:	Sensor Data Visualisatie Toolkit
Projectcode:	20400060
Status:	DEFINITIEF V.1.0
Opdrachtgever:	Radboud Universiteit
Contactpersoon:	Linda Carton
Accountmanager:	Marien de Bakker
Projectleider:	Vincent Wissink
Inhoudelijk expert:	Nader in te vullen
Projectteam:	Stefan Knoet Thomas Geurts van Kessel
Plaats:	Nijmegen
Datum:	13-03-2020

Voor akkoord:

Opdrachtgever	Projectleider	Contractmanager
Naam:……	Naam:……	Naam:……
Datum:…….	Datum:…….	Datum:…….
Handtekening:	Handtekening:	Handtekening:

Inhoudsopgave

1. Inleiding

Dit plan van aanpak dient als opzet voor de uitvoering van het Smart Emission 2 project, dat tot stand is gekomen vanuit de Radboud Universiteit te Nijmegen. Dit project is een vervolgproject op het reeds uitgevoerde Smart Emission 1 project. Smart Emission 1 richtte zich op burgerwetenschap, waar met behulp van sensoren, data over milieukwaliteiten zijn verzameld. Vanuit de ondervonden resultaten wordt er een vervolg gegeven in de vorm van het Smart Emission 2 project waarvoor, op basis van het huidige SE data platform, een toolkit wordt ontwikkelt. Dit project wordt uitgevoerd door Stefan Knoet en Thomas Geurts van Kessel, vierde jaars Geo Media & Design studenten aan de HAS Hogeschool. Het project dient binnen 20 weken uitgevoerd te worden en loopt van 10 februari tot 10 juli.

Gedurende het project wordt er een toolkit ontwikkelt met als uitgangspunt om eindgebruikers van dergelijke sensor data platformen de mogelijkheid te geven om, op een laagdrempelige manier, data te visualiseren en te analyseren. Uiteindelijk ligt er belang bij het brengen van bewustwording bij de eindgebruikers, die hiermee ruimere mogelijkheden en ondersteuning ondervinden bij het gebruiken en toepassen van de sensor data. De wensen die de eindgebruikers stellen aan de tools, dient als fundamentele input voor de ontwikkeling van de toolkit. Vanuit deze ontwikkeling zal de mogelijkheid ontstaan om als eindgebruiker eigen use cases op te zetten waarmee milieukwaliteiten onderzocht kunnen worden. Dit geeft de eindgebruikers uiteindelijk de flexibiliteit om samenwerkingen aan te gaan met overheidsinstanties en bedrijven. Tot slot zal het mogelijk worden om te starten met nieuwe burgerwetenschappelijke onderzoeken.

Binnen het plan van aanpak worden verschillende hoofdstukken behandeld. In het hoofdstuk projectdefinitie wordt er toegelicht wat het project inhoudt. In het hoofdstuk activiteiten worden alle activiteiten besproken die tijdens het project uitgevoerd gaan worden. Daarnaast wordt er een toelichting gegeven over de te verrichten onderzoeksmethodes, betreffende deadlines en taakverdeling. In het hoofdstuk planning wordt de planning toegelicht dat is gemaakt om van de te verrichten activiteiten een tijdsverdeling te maken. In het hoofdstuk organisatie worden verschillende belangrijke organisatieaspecten binnen het project besproken. Vervolgens worden in het volgende hoofdstuk de aanvullende onderdelen besproken. Als laatste worden de gebruikte bronnen weergegeven en zijn er in de bijlagen alle bestanden toegevoegd die van toepassing zijn voor het plan van aanpak.

2. Projectdefinitie

2.1 Achtergrond & aanleiding

De Radboud universiteit is een Nederlandse universiteit opgericht in 1923. De universiteit houdt zich bezig met brede, internationaal georiënteerde en studentgerichte onderwijs en onderzoek, waar nadruk wordt gelegd op wetenschappelijke nieuwsgierigheid en het belang voor de samenleving (Radboud Universiteit, 2020a). De Radboud universiteit heeft zeven faculteiten die het hele wetenschappelijke spectrum van onderzoek en onderwijs omvatten. Elke faculteit is verenigd uit een aantal verwante vakgebieden (Radboud Universiteit, 2020b). Het desbetreffende project wordt uitgevoerd voor de afdeling 'Geografie, Planologie en Milieu', wat valt onder de faculteit 'der Managementwetenschappen' (Radboud Universiteit, 2020c).

De afdeling Geografie, Planologie en Milieu neemt deel aan een breed scala van onderzoeksprojecten, zo ook het eerder uitgevoerde project Smart Emission 1. Gedurende dit project zijn, in de stad Nijmegen, de lokale- en fysieke milieukwaliteiten in kaart gebracht met behulp van burgers, innovatieve sensing en ICT-technologie (Radboud Universiteit, 2020d). Als vervolg is het project Smart Emission 2 van start gegaan waarin de top drie van meest dringende vervolgonderzoeksprioriteiten, resulterend uit het project Smart Emission 1, worden aangepakt (NWO TTW, 2017).

De aanleiding van het project komt voort uit één van de drie onderzoeksprioriteiten. Dit houdt in dat er een toolkit wordt gebouwd op basis van het huidige SE data platform (en de 'Heron Viewer'), waardoor de functionaliteiten beter aansluiten op de wensen en eisen van de eindgebruikers.

2.2 Probleemstelling

Vanuit het project Smart Emission 1 is er een actieve groep burgerwetenschappers doorgegaan met het meten en analyseren van de sensor data. Echter lopen de burgers tegen het probleem aan dat er op dit moment weinig ondersteuning is om deze data te kunnen analyseren en visualiseren. Op dit moment zijn er namelijk geen functionele tools aanwezig, waarmee burgers met de sensor data aan de slag kunnen. De drempel om actief gebruik te maken van de viewer ligt hierdoor hoog en dit is iets wat tijdens dit project verbeterd dient te worden. Door één of meerdere tools te ontwikkelen en het mogelijk te maken om deze te kunnen integreren binnen verschillende (sensor) data platformen zorg je ervoor dat burgers gestimuleerd worden om met sensor data aan de slag te gaan.

2.3 Doelstelling

Het doel van dit project is om een toolkit te ontwikkelen met als uitgangspunt het huidige SE Data Platform voor de visualisatie van sensor data. Deze sensor data bevat gegevens over diverse milieukwaliteiten, zoals luchtkwaliteit.

Met behulp van de tools moet het, op een functionele en gebruikersvriendelijke manier, mogelijk gemaakt worden voor burgers en experts om diverse sensor data te bekijken, analyseren en visualiseren. De eindgebruikers dienen betrokken te worden bij de ontwikkeling van de tools, zodat de data-analyse en -visualisatie tools goed aansluiten op de wensen en eisen van de gebruikers. Hierbij wordt onder andere gebruik gemaakt van een aantal aspecten uit de User Interface Design. Zo wordt er met behulp van een MoSCoW model prioriteiten gesteld over welke eisen in het platform verwerkt gaan worden. Hiernaast kunnen eventuele schetsen en (online) schermontwerpen bijdragen aan het ontwerp van de toolkit, waarbij ook nagedacht wordt over hoe de eisen in de toolkit verwerkt kunnen worden. Dit laatste is geen noodzakelijke stap in het gehele onderzoek, maar kan eventueel worden toegepast om de eisen beter te vertalen naar het platform. In overleg met de opdrachtgever dient er te worden bekeken of deze stap uitgevoerd dient te worden.

De toolkit wordt gebaseerd op ‘use case analyses’ die tot stand komen uit interviews met actieve burgers en experts. Hiermee dient het mogelijk te worden gemaakt om met behulp van de toolkit inzichten te krijgen over milieukwaliteiten in een bepaald gebied.

Door deze inzichten kunnen burgers en experts inzicht krijgen in lokale milieukwaliteiten en hun eigen leefomgeving analyseren. Dit moet uiteindelijk leiden tot meer interactie en discussie, wat uiteindelijk zorgt voor meer bewustwording over een specifieke omgeving. Deze bewustwording wordt ook gecreëerd doordat burgers zelf actiever bezig zijn met de sensor data, doordat zij hun eigen ‘use case analyse’ kunnen uitvoeren.

De toolkit dient aan te sluiten bij standaarden voor open source data en –software. Het is hierbij van belang dat de uiteindelijke gemaakte eindproducten openbaar worden gesteld, zodat deze met weinig moeite in andere sensor data platformen gebruikt kunnen worden. Dit wordt bereikt door gebruik te maken van API’s (Application Programming Interfaces), waarmee het mogelijk wordt om de toolkit te integreren binnen andere dataplatformen.

Het uiteindelijke doel van de ontwikkeling van de tools is om het voor actieve gebruikers van het SE data platform (Heron Viewer) en andere sensor data platformen eenvoudiger te maken om met sensor data aan de slag te gaan en om meer nieuwe gebruikers aan te trekken. Vanuit hier kunnen er vervolgens individueel of in groepsverband burgerwetenschappelijke onderzoeken gestart worden.

2.4 Onderzoek & ontwerp

Om antwoord te geven op de probleemstelling en het project een duidelijke richting te geven, is er een hoofdonderzoeksvraag opgesteld. De hoofdonderzoeksvraag is leidend voor het onderzoek en zorgt voor structuur, deze luidt als volgt:

“Hoe kan ervoor worden gezorgd dat, met behulp van data analyse en -visualisatie tools, het SE data platform en andere sensor data platformen op het gebied van gebruikersvriendelijkheid en functionaliteit verbeterd worden voor het stimuleren van burgersparticipatie?”

Om de hoofdonderzoeksvraag verder af te bakenen zijn er een aantal deelonderzoeksvragen opgesteld. De deelonderzoeksvragen dienen als verbreding en verdieping van het onderzoek. De vragen luiden als volgt:

<u>Inventarisatie bestaande functionaliteiten van sensor data platformen voor milieudata</u>

1. “ Wat is het SE data platform en welke huidige functionaliteiten bevat het? ”
2. “ Welke functionaliteiten bevatten vergelijkbare sensor data platformen? ”

<u>Wensen eindgebruikers</u>

3. “ Welke wensen, vanuit de eindgebruikers, zijn van invloed op het uiteindelijke doel van de toolkit? “

<u>Prioriteitenstelling van de belangrijkste gebruikerseisen (MoSCoW-model):</u>

4. ‘’ Welke wensen dienen omgezet te worden naar concrete eisen? ’’
5. “ Welke eisen zijn van belang voor het ontwikkelen van de tools? ’’

<u>Verkenning aanbod softwaremogelijkheden voor de ontwikkeling van een toolkit</u>

6. “ Wat zijn de mogelijkheden voor het ontwikkelen van tools? ’’

7. “ Welk softwareprogramma wordt gebruikt voor het programmeren van de tools? ’’

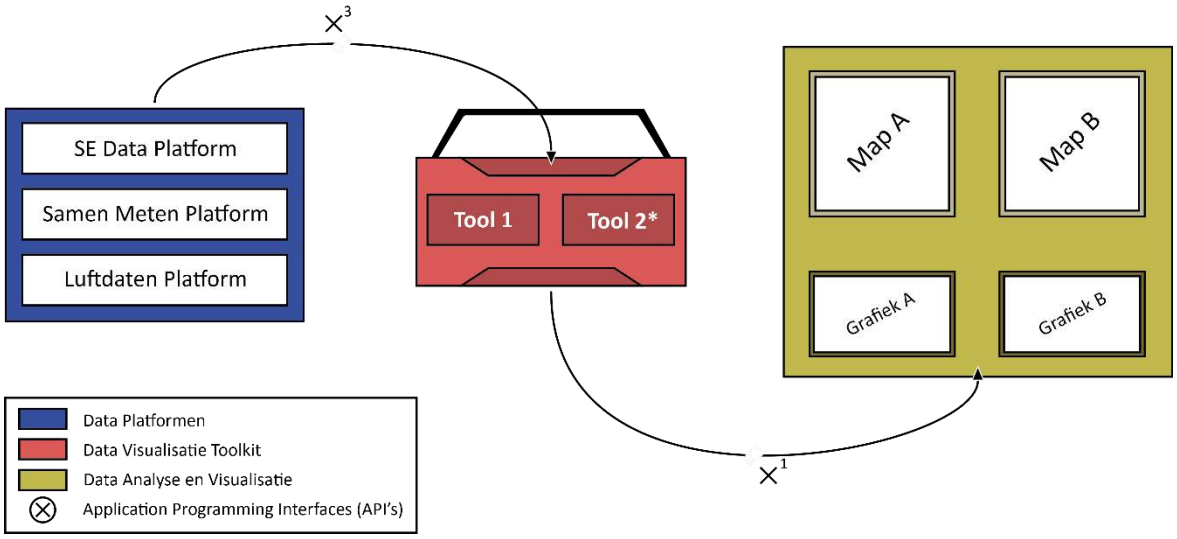
<u>Ontwikkeling toolkit (funtioneel en gebruikersvriendelijk)</u>

8. (“Hoe kan, met behulp van user interface design, de gebruikersvriendelijkheid van de tool(s) gewaarborgd worden? ”)
9. “Hoe kunnen tools er uit zien, die de functionaliteit voor sensor data analyse en visualisatie (voor het uitvoeren van burgerwetenschap) verbeteren? ”

<u>Evaluatie functionaliteit en gebruikersvriendelijkheid</u>

10. “ Welke ontwikkelde tools sluiten aan op de wensen van de eindgebruikers? "

Om de ontwerpvraag van de toolkit te verduidelijken, is hiervan een schematische tekekning gemaakt dat te vinden is in figuur 1.



Figuur 1: Schematische weergave van een visualisatie toolkit

2.5 Producten

- *Plan van aanpak:* om duidelijkheid te creëren rondom het project, wordt er een plan van aanpak opgezet waarin alle elementen worden behandeld die van toepassing zijn bij het voorbereiden, uitvoeren en afronden van het project.

- *Toolkit:* gedurende het project worden er tools ontwikkeld voor data visualisatie op basis van het SE data platform. Om deze tools voor de opdrachtgever in een duidelijk overzicht weer te geven, worden alle ontwikkelde en goedgekeurde tools samengevoegd in een toolkit. Deze toolkit wordt ontwikkeld met behulp van open source software. Hierbij is het belangrijk dat het mogelijk is om de toolkit te kunnen integreren binnen verschillende sensor data platformen. Dit wordt bereikt door gebruik te maken van API’s (Application Programming Interfaces), waarmee het mogelijk wordt om de toolkit te integreren binnen diverse sensor data platformen.

- *Tools:* om gebruikers van sensor data platformen meer analyse en visualisatie mogelijkheden te geven met de beschikbare sensor data, worden er tools ontwikkeld. Elke tool zal één of meerdere functies vervullen voor de verbreding van de analyse en visualisatie mogelijkheden op de platformen. Hierbij wordt er meer belang gehecht aan het opleveren van één of twee werkende tools, dan aan een brede indicatie van mogelijkheden en adviezen.
  Het is hierdoor erg belangrijk dat er onderzoek wordt gedaan naar de mogelijkheden vanuit bepaalde software en programmeertalen en van hieruit te bekijken welke tools haalbaar zijn om te ontwikkelen binnen de gestelde tijdsperiode.

- *Toolbeschrijvingen:* om de gebruikersvriendelijkheid van de tools te waarborgen, worden de tools getest en gevalideerd. Wanneer gebruikers moeite hebben met het gebruiken en toepassen van een tool, wordt er een toolbeschrijving ontwikkeld waarin stapsgewijs wordt uitgelegd hoe de tool gebruikt moet worden.

- *Einddocumentatie:* om weer te geven hoe er gedurende het project producten zijn ontwikkeld, wordt er in een einddocument het projectproces vastgelegd. In het document wordt, als dit van toepassing is, ook een vervolgadvies toegevoegd.

3. Activiteiten

3.1 Te ondernemen stappen

Stap 1: Om goed van start te gaan met het project, is het van belang om desbetreffende documenten, internetbronnen en vergelijkbare platformen te bestuderen. Deze stap dient als fundament voor het project, waarmee vervolgens het plan van aanpak kan worden opgesteld. Vanuit de eerste stap worden de overige stappen, waarmee het project kan worden uitgevoerd, geformuleerd.

Stap 2: Bij de tweede stap is het van belang dat de wensen van de eindgebruikers en opdrachtgever overzichtelijk worden weergegeven. Hierbij wordt gebruik gemaakt van een bestaande lijst waar vooraf opgestelde wensen in staan geformuleerd. Hiernaast dient er met behulp van een interactieve sessie concreter te worden achterhaald welke wensen voor ons onderzoek van belang zijn en welke niet.

Stap 3: Bij de derde stap worden de wensen van de eindgebruikers en opdrachtgever omgezet naar een aantal concrete eisen. Om de eisen te prioriteren, wordt gebruik gemaakt van een MoSCoW model. In het model worden de eisen onderverdeeld in vier categorieën: Must-, Should, Could- en Would haves. Aan de hand van het MoSCoW model wordt bepaald welke eisen van belang zijn voor het ontwikkelen van de tools.

Stap 4: Bij de vierde stap worden de tools gerealiseerd. De manier waarop de tools worden ontwikkeld hangt af van de voorgaande stappen en het te verrichten literatuur- en praktijkonderzoek. Bij het literatuuronderzoek wordt gekeken wat de mogelijkheden zijn bij het ontwikkelen van de tools. Hiernaast kunnen we ook inzichten opdoen door in gesprek te gaan met verschillende partijen zoals de HAS Den Bosch en bij professionele supporters en eindgebruikers van het Smart Emission project. Voorbeelden hiervan zijn het RIVM, Kadaster, Gemeente Nijmegen en Geonovum.

Bij het praktijkonderzoek wordt vervolgens onderzocht welke softwareprogramma's worden gebruikt voor het programmeren van de tools. Na het afronden van het praktijkonderzoek wordt bepaald welke programmeertaal gebruikt gaat worden, waar inhoudsdeskundigen bij worden betrokken. Nadat de programmeer- en softwaremogelijkheden bekend zijn kunnen er eventueel schetsen en (online) schermontwerpen gemaakt worden om te kijken hoe de eisen in de toolkit verwerkt kunnen worden.

Na het voltooien van een tool, zal deze getest en gevalideerd worden voor een statusbepaling. Wanneer een tool als ongeschikt wordt verklaard, kan de tool verwikkeld zijn in een iteratief proces voor verbetering.

Stap 5: Bij de vijfde stap worden de geschikte tools uit stap vier nogmaals getest en gevalideerd. Ditmaal wordt het testen en valideren georganiseerd met de betrokken eindgebruikers en opdrachtgever (gedurende de tweede interactieve sessie). Bij het testen en valideren wordt gebruikersvriendelijkheid centraal gesteld voor de statusbepaling van de tools. Wanneer een tool als ongeschikt wordt verklaard, zullen er eventuele verbeteringen doorgevoerd worden. Als dit door tijdsnood niet meer mogelijk is, zal hiervoor een vervolgadvies opgesteld worden. Daarnaast zal, gedurende het testen en valideren, naar voren komen voor welke tools een beschrijving opgesteld moet worden.

Stap 6: Bij de zesde stap worden de eindproducten geleverd; de toolkit met alle datavisualisatie tool(s) voor de sensor data platformen, de toolbeschrijvingen en de documentatie van het projectproces met het vervolgadvies. Als afsluiting van het project wordt er een presentatie gegeven over het uitgevoerde onderzoek.

3.2 Overzicht deelvragen met bijbehorende stappen

Deelvragen	**Bijbehorende stappen**
Wat is het SE data platform en welke huidige functionaliteiten bevat het?	Stap 1
Welke functionaliteiten bevatten vergelijkbare sensor data platformen?	Stap 1
Welke wensen, vanuit de eindgebruikers, zijn van invloed op het uiteindelijke doel van de toolkit?	Stap 2
Welke wensen dienen omgezet te worden naar concrete eisen?	Stap 3
Welke eisen zijn van belang voor het ontwikkelen van de tools?	Stap 3
Wat zijn de mogelijkheden voor het ontwikkelen van tools?	Stap 4
Welk softwareprogramma wordt gebruikt voor het programmeren van de tools?	Stap 4
(Hoe kan, met behulp van user interface design, de gebruikersvriendelijkheid van de tool(s) gewaarborgd worden?)	Stap 4
Hoe kunnen tools er uit zien, die de functionaliteit voor sensor data analyse en visualisatie (voor het uitvoeren van burgerwetenschap) verbeteren?	Stap 5
Welke ontwikkelde tools sluiten aan op de wensen van de eindgebruikers?	Stap 5

Tabel 1: overzicht deelvragen met bijbehorende stappen

3.3 Stappenplan met hoofd- en subcategorieën

1. Inlezen en verkenning
    1.1 Bestuderen van documenten, internetbronnen en vergelijkbare sensor data platformen
    1.2 Opstellen van het plan van aanpak

2. Bruikbaarheidslijst ophalen bij eindgebruikers
    2.1 Eerste interactieve sessie
        2.2.1 Workshop/presentatie
    2.2 Houden van interviews
    2.3 Opzet van een wensenlijst

3. Opstellen prioriteitenlijst
    3.1 Wensen omzetten naar concrete eisen
    3.2 Opzet van het MoSCoW model
    3.3 Terugkoppeling van het MoSCoW model
    3.4 Afronding van het MoSCoW model

4. Ontwikkeling toolkit
    4.1 Noodzakelijke tools halen uit het MoSCoW model
    4.2 Uitvoeren literatuuronderzoek voor ontwikkeling tools
    4.3 Na aanvang van de uitkomsten bij het literatuuronderzoek: starten met uitvoeren praktijkonderzoek geschikte softwareprogramma's (HAS Hogeschool, professionele supporters en eindgebruikers van het Smart Emission project).
    4.4 (Maken van eventuele schetsen en schermontwerpen)
    4.5 Programmeren van de tools
        4.4.1 (hulp van inhoudsdeskundigen)
        4.4.2 Eigen test en evaluatie
        - Geschikt: doorgaan naar stap 5
        - Ongeschikt: blijven werken aan stap 4.4
    4.6 Tussentijdse terugkoppelingen met opdrachtgever en projectbegeleider

5. Evaluatie en testen tools
    5.1 Testen van de ontwikkelde tools uit stap 4
    5.2 Evaluatie van de tooltest: kijken naar gebruikersvriendelijkheid
        - Geschikt: doorgaan naar stap 5.3
        - Ongeschikt: terug naar stap 4.4
    5.3 Opstellen van eventuele toolbeschrijvingen
    5.4 Opstellen van een eventueel vervolgadvies

(De tools worden in samenwerking met de eindgebruikers getest en geëvalueerd)

6. Oplevering toolkit, -beschrijving en documentatie
    6.1 Samenvoegen van geschikte tools met bijhorende beschrijving
    6.2 Opstellen van een einddocumentatie
        6.2.1 Beschrijving projectproces
        6.2.2. Samenvoegen toolbeschrijvingen en vervolgadvies
    6.3 Tweede interactieve sessie
        6.3.1 Workshop/eindpresentatie

3.3 Methoden

Interviews: voor het verzamelen van benodigde informatie is het belangrijk om bij bepaalde onderwerpen verdiepend op de stof in te gaan. Het houden van interviews dient als input voor twee hoofddoelen; het ophalen van de eindgebruikerswensen en verdieping op het ontwikkelen van tools. Door interviews op te stellen en te houden met personen met de benodigde kennis en expertise, wordt de benodigde informatie vergaard. Omdat het voor het onderzoek van belang is om verdiepend in te gaan op een aantal onderwerpen, zullen er interviews gehouden worden bij meerdere stappen in het proces; in dit geval stappen 2 tot 5.

Literatuuronderzoek: voor het project is het van belang dat er kennis wordt verzameld over het onderzoeksonderwerp. Door wetenschappelijke literatuur, artikelen, internetbronnen, platformen en boeken te bestuderen wordt ervoor gezorgd dat huidige kennis en nieuwe kennis zo effectief mogelijk wordt gebruikt en toegepast binnen het onderzoek. De onderwerpen de aan bod komen staan op chronologische volgorde: mogelijkheden tools, ontwikkelsoftware en programmeren. Omdat het van belang is dat niet alleen één deel van de kennis gebruikt wordt, zal literatuuronderzoek verwikkeld zijn in de zes stappen van het project.

Interactieve sessies: voor het project is het van belang dat de eindproducten aansluiten op de wensen en verwachtingen van de eindgebruikers en de opdrachtgever. Om dit te waarborgen worden er twee interactieve sessies georganiseerd. De eerste interactieve sessie focust zich op het in kaart brengen van de verschillenden wensen. Dit wordt gerealiseerd in vorm van een workshop.
Bij de tweede interactieve sessie ligt de focus op het evalueren van de gemaakte producten, om zo erachter te komen of deze voldoen aan de verwachtingen. Dit zal, evenals de eerste sessie, gerealiseerd worden in vorm van een workshop.

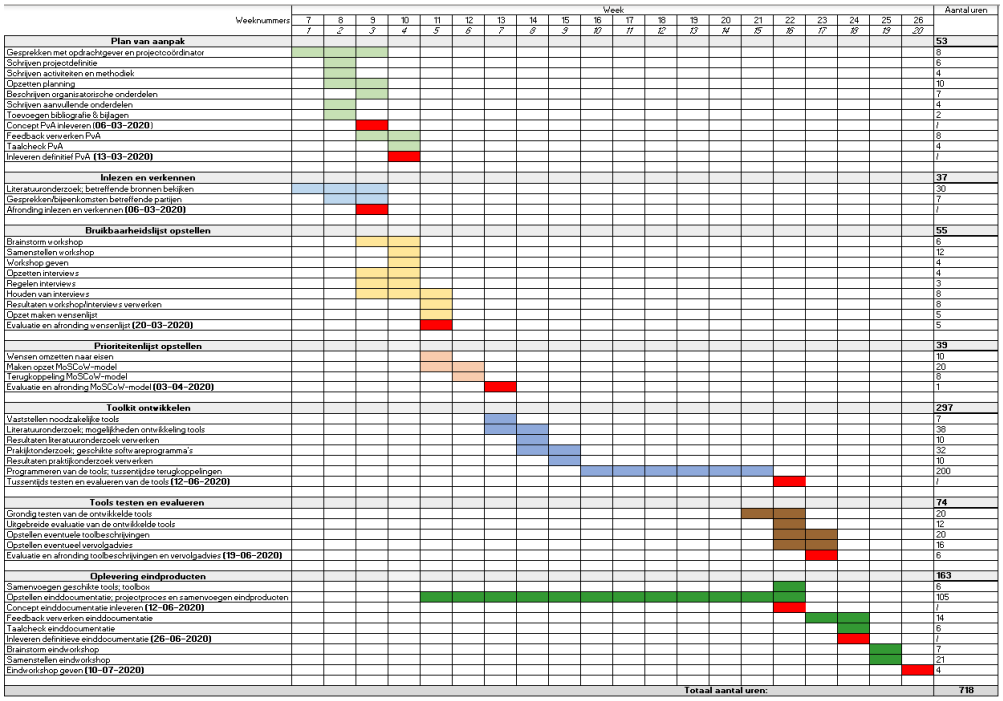
Praktijkonderzoek: aan het einde van het project wordt er een toolkit geleverd bestaande uit een of meerdere datavisualisatie tools, om deze tools te ontwikkelen moet er geprogrammeerd worden. Om te programmeren kunnen er meerdere softwareprogramma's gebruikt worden, het is hierbij van belang om de juiste programma's te vinden waarmee de tools ontwikkeld kunnen worden. Door in stap vier van het projectproces te onderzoeken welke programma's geschikt zijn voor het ontwikkelen van de tools, kan de stap gemaakt worden naar het programmeren.

Schets- en schermontwerpen: na het vaststellen van de software- en programmeermogelijkheden kunnen er eventueel schetsen en (online) schermontwerpen gemaakt worden om te kijken hoe de eisen in de toolkit verwerkt kunnen worden. Dit is echter geen noodzakelijke methode voor het onderzoek, maar kan eventueel worden toegepast om de eisen beter te vertalen naar het platform. In overleg met de opdrachtgever dient er te worden bekeken of deze methode uitgevoerd gaat worden.

Programmeren: na het vaststellen van de benodigde softwareprogramma's kan er geprogrammeerd worden. Met behulp van programmeren worden er tools ontwikkelt die aansluiten bij de wensen en verwachtingen van de eindgebruiker en opdrachtgever. Wanneer een of meerdere tools niet volledig voldoen aan de verwachtingen, is het mogelijk om nog enige aanpassingen door te voeren. Het programmeren zal plaats vinden in stappen 4 en 5 van het projectproces.

4. Planning

4.1 Overzicht planning



Weeknummers	Week 7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	Aantal uren
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	
Plan van aanpak																					**53**
Gesprekken met opdrachtgever en projectcoördinator																					8
Schrijven projectdefinitie																					6
Schrijven activiteiten en methodiek																					4
Opzetten planning																					10
Beschrijven organisatorische onderdelen																					7
Schrijven aanvullende onderdelen																					4
Toevoegen bibliografie & bijlagen																					2
Concept PvA inleveren **(06-03-2020)**																					/
Feedback verwerken PvA																					8
Taalcheck PvA																					4
Inleveren definitief PvA **(13-03-2020)**																					/
Inlezen en verkennen																					**37**
Literatuuronderzoek; betreffende bronnen bekijken																					30
Gesprekken/bijeenkomsten betreffende partijen																					7
Afronding inlezen en verkennen **(06-03-2020)**																					/
Bruikbaarheidslijst opstellen																					**55**
Brainstorm workshop																					6
Samenstellen workshop																					12
Workshop geven																					4
Opzetten interviews																					4
Regelen interviews																					3
Houden van interviews																					8
Resultaten workshop/interviews verwerken																					8
Opzet maken wensenlijst																					5
Evaluatie en afronding wensenlijst **(20-03-2020)**																					5
Prioriteitenlijst opstellen																					**39**
Wensen omzetten naar eisen																					10
Maken opzet MoSCoW-model																					20
Terugkoppeling MoSCoW-model																					8
Evaluatie en afronding MoSCoW-model **(03-04-2020)**																					1
Toolkit ontwikkelen																					**297**
Vaststellen noodzakelijke tools																					7
Literatuuronderzoek; mogelijkheden ontwikkeling tools																					38
Resultaten literatuuronderzoek verwerken																					10
Praktijkonderzoek; geschikte software programma's																					32
Resultaten praktijkonderzoek verwerken																					10
Programmeren van de tools; tussentijdse terugkoppelingen																					200
Tussentijds testen en evalueren van de tools **(12-06-2020)**																					/
Tools testen en evalueren																					**74**
Grondig testen van de ontwikkelde tools																					20
Uitgebreide evaluatie van de ontwikkelde tools																					12
Opstellen eventuele toolbeschrijvingen																					20
Opstellen eventueel vervolgadvies																					16
Evaluatie en afronding toolbeschrijvingen en vervolgadvies **(19-06-2020)**																					6
Oplevering eindproducten																					**163**
Samenvoegen geschikte tools; toolbox																					6
Opstellen einddocumentatie; projectproces en samenvoegen eindproducten																					105
Concept einddocumentatie inleveren **(12-06-2020)**																					/
Feedback verwerken einddocumentatie																					14
Taalcheck einddocumentatie																					6
Inleveren definitieve einddocumentatie **(26-06-2020)**																					/
Brainstorm eindworkshop																					7
Samenstellen eindworkshop																					21
Eindworkshop geven **(10-07-2020)**																					4
Totaal aantal uren:																					**718**

De bovenstaande planning geeft de tijdsverdeling weer van de te ondernemen stappen tijdens het project voor één persoon. Voor alle stappen is een globale inschatting gemaakt over hoeveel weken en uren het in beslag zal nemen. Deze planning dient ter ondersteuning voor het uitvoeren van de stappen zonder hierbij in tijdnood te komen. In de planning zijn namelijk verschillende deadlines verwerkt, wanneer deze deadlines worden nagestreefd kan bij het voltooien van het project een passend resultaat worden geleverd. In bijlage 1 is een uitvergrootte versie van de planning te vinden.

4.2 Deadlines gedurende het project

Stappen met betrekking tot het project	**Aanvang datum**	**Deadline Datum**
1. Inlezen en verkenning	10-02-2020	06-03-2020
2. Bruikbaarheidslijst ophalen bij eindgebruikers	02-03-2020	20-03-2020
3. Opstellen prioriteitenlijst	16-03-2020	03-04-2020
4. Ontwikkeling toolkit	30-03-2020	12-06-2020
5. Evaluatie en testen tools	01-06-2020	19-06-2020
6. Oplevering toolkit, -beschrijving en documentatie	22-06-2020	10-07-2020

Tabel 2: Deadlines met betrekking tot het project

* Opmerking: De resultaten uit stap 5 en 6 dienen grotendeels afgerond te zijn voor de inleverdatum van het einddocument v1.0. Dit, doordat het van belang is om deze resultaten in grote lijnen mee te nemen in de einddocumentatie.

Deadlines met betrekking tot de HAS Hogeschool	**Datum**
Plan van aanpak (v0.0)	06-03-2020
Plan van aanpak (v1.0)	13-03-2020
Plan van aanpak (v2.0)	20-03-2020
Einddocumentatie (v0.0)	12-06-2020
Einddocumentatie (v1.0)	26-06-2020
Einddocumentatie (v2.0)	Blok 1, leerjaar 5
Portfolio professionele houding (tussenbeoordeling)	17-04-2020
Portfolio professionele houding	26-06-2020

Tabel 3: Deadlines met betrekking tot school

= Herkansing

4.3 Taakverdeling

Het project dient in grote lijnen in teamverband uitgewerkt te worden. Dit betekent dat alle stappen uit het stappenplan grotendeels uitgevoerd gaan worden door onderling samen te werken. Echter kan het voorkomen dat verschillende taken hierbij toch verdeeld moeten worden. Dit gaat dan veelal om kleine taken waarbij het niet noodzakelijk is om het te vermelden in het plan van aanpak.

Bij het opstellen van de toolkit gaan er wel grote taken verdeeld worden. Zo worden de verschillende tools die ontwikkelt moeten worden verdeeld onder ons twee. Ieder gaat dus voor zich een tool ontwikkelen, waarbij wel intensief wordt samengewerkt om elkaar hierbij te ondersteunen. Uit de resultaten van het MoSCoW model en het literatuur- en praktijkonderzoek moet blijken hoe deze taken verdeeld gaan worden.

5. Organisatie

5.1 Begroting

Tijdens het gehele project is er de mogelijkheid om inhoudsdeskundige in te huren. In totaal is hier 40 uur voor geserveerd. Er zijn verschillende partijen betrokken bij dit project die deze uren kunnen invullen. Het gaat om de volgende partijen

- HAS Hogeschool
- Kadaster
- RIVM
- Gemeente Nijmegen
- Geonovum
- Interno

In de loop van het project moet het duidelijk worden op welke manier deze uren worden ingezet en daarbij ook welke partijen hierbij betrokken worden. Wanneer er uren gemaakt worden door externe partijen wordt de onderstaande tabel ingevuld.

Inhoudsdeskundige	Betrokken bij	Resultaat	Uren	Kosten
...	...	...	...	€...
…	...	...	...	€...
…	...	...	...	€...

Tabel 4: Opzet van een tabel voor het invullen van gemaakte uren door externe partijen

In bijlage 2 is een declaratieformulier te vinden, waar (hulp)middelen in staan genoteerd die van toepassing zijn op dit project en gedeclareerd dienen te worden.

5.2 Afspraken resultaten/producten

Tijdens dit project worden er twee belangrijke producten opgeleverd, namelijk een toolkit en een betreffende einddocumentatie. Voor beiden producten is er een beoordelingsformulier opgesteld, wat een toelichting geeft op welke manier deze producten door de projectbegeleider beoordeeld gaan worden. Hieronder wordt voor beiden producten kort toegelicht hoe deze beoordelingsformulieren tot stand zijn gekomen.

<u>Einddocumentatie</u>

Het beoordelingsformulier van de einddocumentatie is tot stand gekomen vanuit het beoordelingsformulier voor het technisch rapport van de specialisatie ‘Analist’. De beoordelingseenheden en de te verdelen percentages zijn voor dit rapport aangepast, maar de basis is hetzelfde gebleven. Binnen dit formulier wordt er in totaal 95% verdeeld onder alle beoordelingseenheden. Hoe groot het percentage per beoordelingseenheid is hangt af van het belang dat er aan wordt gekoppeld. Hiernaast bevat het formulier een tweetal vereisten, waar de einddocumentatie minimaal aan moet voldoen. Wanneer er niet aan deze vereisten wordt voldaan kan de einddocumentatie niet met voldoende afgesloten worden. Het beoordelingsformulier is te vinden in bijlage 3.1.

<u>Toolkit</u>

Het beoordelingsformulier van de toolkit is tot stand gekomen vanuit het beoordelingsformulier voor het marktplan van de module ‘Business & Consultancy’. De beoordelingseenheden en de te verdelen wegingen zijn voor dit rapport aangepast, maar de basis is hetzelfde gebleven. Binnen elke beoordelingseenheid kan er onvoldoende, voldoende of goed behaald worden. Hier zijn de scores 1 t/m 3 aan gekoppeld.

Daarnaast bevat elke beoordelingseenheid een eigen score die vermenigvuldigd wordt met de daaruit behaalde score. In totaal kunnen er op deze manier 39 punten worden behaald, waarbij 21,5 punten het minimum vormt voor een voldoende. Dit beoordelingsformulier is te vinden in bijlage 3.2.

In de onderstaande tabel is de verdeling te zien van de weging voor alle beoordelingseenheden die het eindcijfer bepalen. Hier dragen de eindproducten (30% toolkit en 20% einddocumentatie) en de professionele houding beiden voor 50% aan mee. Om het gehele project met een voldoende af te sluiten dient er voor beiden onderdelen minimaal een 5,5 gehaald te worden (bodemcijfer).

Beoordelingseenheid	**Weging**
Producten	**50%**
Toolkit	20%
Einddocumentatie	30%
Professionele houding	**50%**

Tabel 5: beoordelingseenheden met betrekking tot het eindcijfer

5.3 NAW gegevens

Naam	**Functie**	**Taken**	**Contactinformatie**
Linda Carton	Opdrachtgever	Overlegt en communiceert met projectteam over ontwikkeling en levering product(en)	*Mail:* *l.carton@fm.ru.nl* *Telefoon:* *+31 6 24548646*
Vincent Wissink	Projectleider, contractmanager & coach	Stuurt team aan, houdt kosten bij en verzorgt persoonlijke ondersteuning	*Mail:* *V.Wissink@has.nl* *Telefoon:* +31 6 24410069
John Ypma	BO-coördinator	Verantwoordelijk gesteld voor waarborgen product- en beoordelaar kwaliteit	*Mail:* *J.Ypma@has.nl* *Telefoon:* +31 6 55175834
Marien de Bakker	Acquisiteur	Voert relatiebeheer uit bij de opdrachtgever	*Mail:* *M.deBakker@has.nl* *Telefoon:* *+31 6 46141177*
Thomas Geurts van Kessel	Projectlid	Voert project in groepsverband uit en houdt individuele ontwikkelingen bij	*Mail:* *T.GeurtsvanKessel@student.has.nl* *Telefoon:* +31 6 19149167
Stefan Knoet	Projectlid	Voert project in groepsverband uit en houdt individuele ontwikkelingen bij	*Mail:* *S.Knoet@student.has.nl* *Telefoon:* +31 6 34157288

6. Aanvullende onderdelen

6.1 Afbakening

Het project zal een looptijd van 20 weken hebben; van 10 februari 2020 tot 10 juli 2020.
Tijdens het opstellen van de wensenlijst wordt gebruik gemaakt van een bestaande wensenlijst en de input van actieve gebruikers van het platform. Wensen van gebruikers die niet actief gebruik maken van het platform worden niet meegenomen.

Het doel van dit project is om de gebruiksvriendelijkheid en functionaliteit van het SE data platform en andere sensor data platformen te verbeteren. Dit wordt gerealiseerd met behulp van een aantal tools, die op basis van het huidige SE Data platform (Heron Viewer) ontwikkeld dienen te worden. Het is dus niet de bedoeling dat het gehele platform her ontwikkeld dient te worden.

Het eindresultaat van dit project is dat er een toolkit op basis van het huidige SE Data Platform wordt opgeleverd, waardoor het voor (actieve) gebruikers eenvoudiger wordt om met sensor data aan de slag te gaan. Dit wordt bereikt door gebruik te maken van API's (Application Programming Interfaces), waarmee het mogelijk wordt om de toolkit te integreren binnen andere dataplatformen. De toolkit wordt ondersteund met behulp van een einddocumentatie en een eindpresentatie.

6.2 Risico's

- Medewerking krijgen van betrokken partijen: Bij een aantal fases binnen het project dienen verschillende partijen betrokken te worden. Het risico hierbij is dat de medewerking niet helemaal loopt zoals van tevoren is gedacht. In de planning is al wat extra tijd ingepland voor deze fases om het risico zo klein mogelijk te maken. Wanneer de medewerking niet loopt zoals verwacht dient dit te worden teruggekoppeld met de opdrachtgever en de projectbegeleider. Dit maakt het mogelijk om in een vroeg stadium het project wellicht anders aan te pakken en om ervoor te zorgen dat er tijd genoeg overblijft om het beoogde resultaat te behalen.

- Beperkte kennis over de ontwikkeling van tools: Bij de ontwikkeling van de tools dient er enige kennis van programmeren aanwezig te zijn. Echter hebben wij hier beperkte kennis over opgedaan, waardoor het voor ons erg lastig kan worden om bepaalde tools te ontwikkelen. In de planning is hiervoor al wat extra tijd ingepland om het risico hierin zo klein mogelijk te maken. Daarnaast kan er gebruik worden gemaakt van een vakinhoudelijk expert, die ons kan helpen bij lastige opgaves.

- *Wegvallen van student en/of stagebegeleider*
  Tijdens het onderzoek bestaat de kans dat de student en/of projectbegeleider wegvalt door bijvoorbeeld privézaken of ziekte. Wanneer dit geval zich voordoet kan het project vertraging oplopen. Hierdoor is het van belang dat dit direct wordt gemeld bij het andere groepslid en projectbegeleiders zodat hier goede afspreken over gemaakt kunnen worden. Wanneer dit vaker voorkomt en/of de afwezigheid voor een lange duur is dient er naar een andere oplossing gezocht te worden.

  Op dit moment krijgt het Corona virus steeds meer vat op Nederland, waardoor hier steeds meer hinder door wordt ondervonden. De kans is op dit moment reëel dat het ook invloed gaat hebben op ons project. Wanneer deze situatie zich voordoet dient er snel met de betrokken afgestemd te worden hoe het project verder wordt voorgezet, zonder dat dit ten koste van het resultaat

- *Het crashen van de apparatuur, wat leidt tot het verliezen van informatie:* Tijdens het project zal er gewerkt worden met een eigen laptop. Het kan voorkomen dat tijdens het onderzoek het apparaat crasht, met als gevold dat belangrijke informatie (bestanden) kwijtraken. Om dit risico te voorkomen is het van belang om zoveel mogelijk informatie online op een Cloud omgeving te zetten.

6.3 Betrokkenen

Het project wordt door twee projectleden uitgevoerd, maar er zijn een aantal andere partijen betrokken bij dit project. Zo moeten burgers en ondernemers die een actieve bijdrage leveren aan het (SE) platform zorgen voor input van de wensenlijst. Deze input is vervolgens weer belangrijk voor de totstandkoming van het MoSCoW model. Hiernaast wordt er een vakinhoudelijke expert(s) bij dit project betrokken om te helpen bij het programmeren van tools.

Bij de interactieve sessies komen diverse partijen langs die betrokken zijn, of zijn geweest, bij project Smart Emission 1 en 2. Zij kunnen, net als de actieve gebruikers van het platform, input geven voor de wensenlijst. Daarnaast kunnen zij ons feedback geven over de gerealiseerde tools, waardoor er eventuele verbeteringen doorgevoerd kunnen worden.

Tot slot kunnen de opdrachtgever en de projectbegeleider invloed uitoefenen op het project, om zo de kwaliteit, van de te leveren producten, te verbeteren en te verzekeren. Dit zorgt ervoor dat het project succesvol afgesloten kan worden.

6.4 Randvoorwaarden

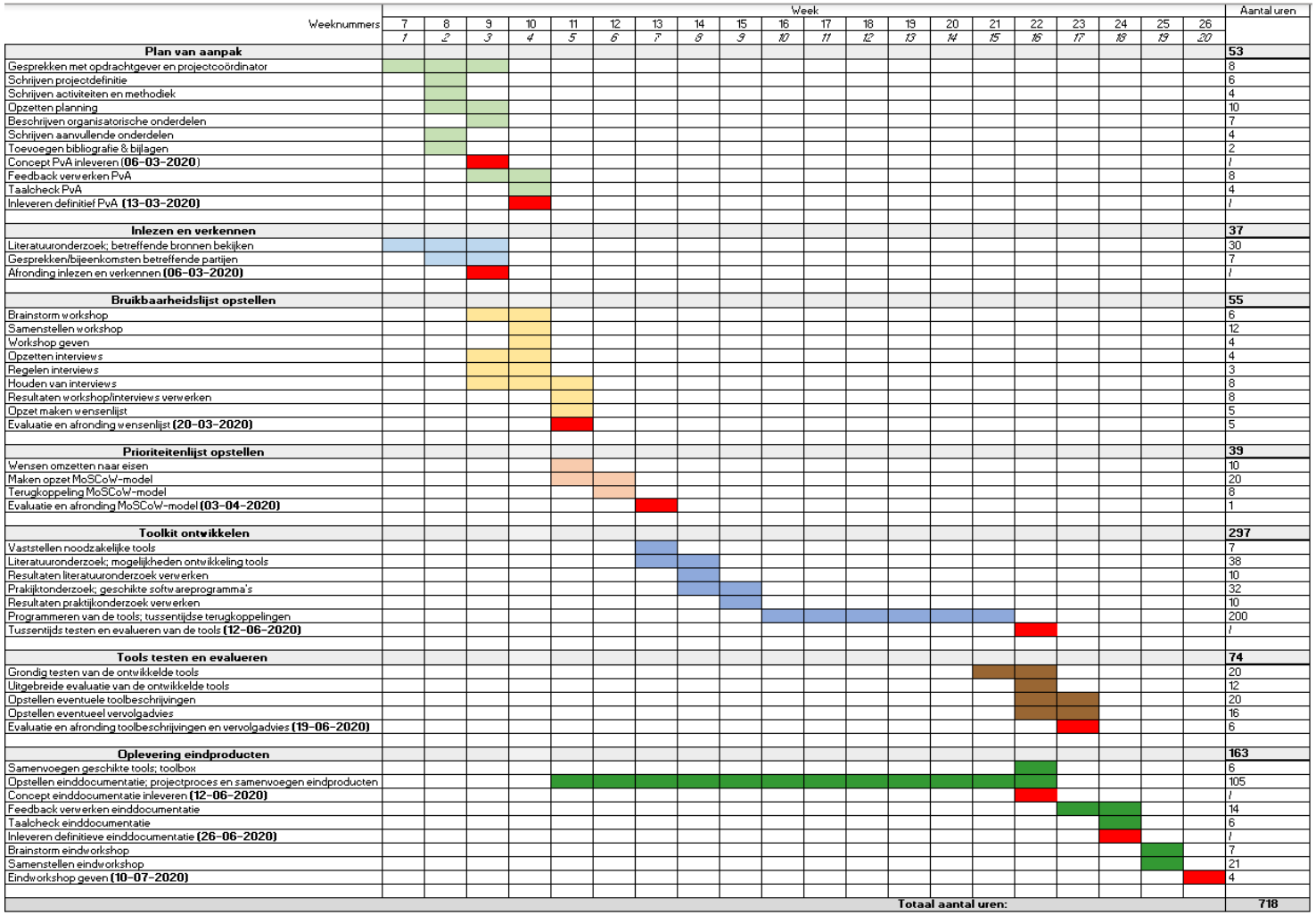
- Er dienen één of twee tools ontwikkeld te worden dat de gebruiksvriendelijkheid en functionaliteit van het SE data platform (Heron Viewer) en andere sensor data platformen verbeterd.
- Bij lastig uitvoerbare tools dient een extra uitleg te worden toegevoegd.
- De projectleden hebben 20 weken de tijd om het project uit te voeren.
- Samenwerking met externe partijen is noodzakelijk voor de uitvoering van het project.
- Bij het opstellen van de wensenlijst worden gebruik gemaakt wensen van actieve gebruikers. Wensen van gebruikers die niet actief gebruik maken van het platform worden niet meegenomen.
- Om het beoogde resultaat te behalen is het van belang dat er regelmatig contactmomenten worden ingepland met de projectbegeleider en de opdrachtgever.
- Tijdens het project dienen twee workshops gehouden te worden. Dit heeft als doel om inzicht te krijgen in de wensen van gebruikers en om het uiteindelijk behaalde resultaat te presenteren.

Bibliografie

- NWO TTW (2017, November 20), Maps4Society Programme, *Smart Emission 2*, geraadpleegd op: 18-02-2020
- Radboud Universiteit (2020a), *Oprichting*, https://www.ru.nl/over-ons/overradboud/geschiedenis/oprichting, geraadpleegd op: 17-02-2020
- Radboud Universiteit (2020b), *Faculteiten Radboud Universiteit*, https://www.ru.nl/over-ons/organisatie/faculteiten/, geraadpleegd op: 17-02-2020
- Radboud Universiteit (2020c), *Geografie, Planologie en Milieu*, https://www.ru.nl/gpm/, geraadpleegd op: 17-02-2020
- Radboud Universiteit (2020d), *Project Smart Emission*, https://www.ru.nl/nsm/imr/our-research/departments/geography-planning-environment, geraadpleegd op: 17-02-2020

Bijlage

Bijlage 1: Planning



	Week																				Aantal uren
Weeknummers	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	
Plan van aanpak																					**53**
Gesprekken met opdrachtgever en projectcoördinator																					8
Schrijven projectdefinitie																					6
Schrijven activiteiten en methodiek																					4
Opzetten planning																					10
Beschrijven organisatorische onderdelen																					7
Schrijven aanvullende onderdelen																					4
Toevoegen bibliografie & bijlagen																					2
Concept PvA inleveren **(06-03-2020)**																					*1*
Feedback verwerken PvA																					8
Taalcheck PvA																					4
Inleveren definitief PvA **(13-03-2020)**																					*1*
Inlezen en verkennen																					**37**
Literatuuronderzoek; betreffende bronnen bekijken																					30
Gesprekken/bijeenkomsten betreffende partijen																					7
Afronding inlezen en verkennen **(06-03-2020)**																					*1*
Bruikbaarheidslijst opstellen																					**55**
Brainstorm workshop																					6
Samenstellen workshop																					12
Workshop geven																					4
Opzetten interviews																					4
Regelen interviews																					3
Houden van interviews																					8
Resultaten workshop/interviews verwerken																					8
Opzet maken wensenlijst																					5
Evaluatie en afronding wensenlijst **(20-03-2020)**																					5
Prioriteitenlijst opstellen																					**39**
Wensen omzetten naar eisen																					10
Maken opzet MoSCoW-model																					20
Terugkoppeling MoSCoW-model																					8
Evaluatie en afronding MoSCoW-model **(03-04-2020)**																					1
Toolkit ontwikkelen																					**297**
Vaststellen noodzakelijke tools																					7
Literatuuronderzoek; mogelijkheden ontwikkeling tools																					38
Resultaten literatuuronderzoek verwerken																					10
Prakijktonderzoek; geschikte softwareprogramma's																					32
Resultaten praktijkonderzoek verwerken																					10
Programmeren van de tools; tussentijdse terugkoppelingen																					200
Tussentijds testen en evalueren van de tools **(12-06-2020)**																					*1*
Tools testen en evalueren																					**74**
Grondig testen van de ontwikkelde tools																					20
Uitgebreide evaluatie van de ontwikkelde tools																					12
Opstellen eventuele toolbeschrijvingen																					20
Opstellen eventueel vervolgadvies																					16
Evaluatie en afronding toolbeschrijvingen en vervolgadvies **(19-06-2020)**																					6
Oplevering eindproducten																					**163**
Samenvoegen geschikte tools; toolbox																					6
Opstellen einddocumentatie; projectproces en samenvoegen eindproducten																					105
Concept einddocumentatie inleveren **(12-06-2020)**																					*1*
Feedback verwerken einddocumentatie																					14
Taalcheck einddocumentatie																					6
Inleveren definitieve einddocumentatie **(26-06-2020)**																					*1*
Brainstorm eindworkshop																					7
Samenstellen eindworkshop																					21
Eindworkshop geven **(10-07-2020)**																					4
Totaal aantal uren:																					**718**

Bijlage 2: Declaratieformulier

Declaratieformulier HAS Kennistransfer en Bedrijfsopleidingen voor STUDENTEN 2019

Naam	Stefan Knoet	**Projectcode**	20400080
Adres	Meijhorst 6175	**Naam bedrijf**	Radboud Universiteit
Woonplaats	Nijmegen		
IBAN-nr	NL04 ABNA 0419 1036 35		

A Reiskosten

datum (dd-mm)	reis van	reis naar	auto vergoeding aantal km € 0,19	totaal €	openbaar vervoer €
				€ 0,00	
				€ 0,00	
				€ 0,00	
				€ 0,00	
				€ 0,00	
				€ 0,00	
				€ 0,00	
				€ 0,00	
				€ 0,00	
			subtotaal A	€ 0,00	€ 0,00

B Leermiddelen

datum (dd-mm)	Omschrijving	bedrag €
26-02 tot 03-07	Adobe Creative Cloud (4 maanden)	€ 78,64
	subtotaal B	€ 78,64

C Lunchkosten

datum (dd-mm)	Omschrijving	bedrag €
	subtotaal C	€ 0,00

D Administratie

datum (dd-mm)	Omschrijving	bedrag €
	subtotaal D	€ 0,00

Totaalbedrag declaratie (A+B+C+D) -> € € 78,64

Ondergetekende verklaart deze declaratie naar waarheid te hebben ingevuld.		Naam en paraaf projectleider	Paraaf HAS Kennistransfer en Bedrijfsopleidingen
Datum 4 maart 2020	Handtekening		
Plaats			

Bijlage 3: Beoordelingsformulieren

Bijlage 3.1 Beoordelingsformulier einddocumentatie

	Weging	Cijfer	Feedback
Minimale vereisten			
A. In het rapport worden de verschillende stappen van het proces beschreven.			Ja/Nee
B. De tekst is in goed Nederlands geschreven en voldoet daarmee aan de eisen voor verslaglegging[1]			Ja
SAMENVATTING	**5%**		
Bevat: onderzoeksvraag en het belang ervan, korte beschrijving methoden en resultaten, interpretatie en discussie resultaten en conclusie die daaruit getrokken worden	10,0%	10	
INLEIDING			
Probleem	**10%**	**10**	
Het onderzoek bevat een duidelijk afgebakende onderzoeksvraag, eventueel met duidelijk omschreven deelvragen.	5,0%	10	
Het schaalniveau van het vraagstuk is duidelijk beschreven	2,0%	10	
De eindgebruiker van het rapport wordt duidelijk beschreven, met een terugkoppeling naar het belang van de onderzoeksvraag voor de eindgebruiker.	3,0%	10	
MATERIAAL EN METHODE			
Onderzoeken	**25%**		
Er wordt een beschrijving gegeven van de ondernomen stappen die zijn genomen bij het literatuuronderzoek naar software mogelijkheden	3,0%	10	
Er wordt een goed onderbouwde keuze gemaakt over bij welke softwareprogramma's een praktijkonderzoek wordt uitgevoerd.	5,0%	10	
Het rapport bevat een beschrijving van de ondernomen stappen die zijn genomen bij het praktijkonderzoek naar het meest geschikte software programma	6,0%	10	
Er wordt een duidelijke beschrijving gegeven van hoe data sets geschikt zijn gemaakt voor analyse, bijvoorbeeld door middel van data transformatie, formatering en structurering.	5,0%	10	
Er wordt een duidelijke uitleg gegeven over de stappen die zijn genomen om te kunnen bepalen wat voor eindproduct(en) ontwikkeld gaan worden.	6,0%	10	
Uitvoering (ontwikkeling toolkit)	**20%**		
Er wordt een uitleg gegeven over de ondernomen stappen die zijn gemaakt gedurende het ontwikkelen van de toolkit en er dient inzicht te worden gegeven van de ondernomen stappen van de inhoudsdeskundigen.	10,0%	10	
Er wordt een onderbouwing gegeven van belangrijke overwegingen die zijn gemaakt gedurende het proces.	6,0%	10	
Terugkoppel- en feedback momenten dienen uitgewerkt en verwerkt te worden in dit rapport.	4,0%	10	
RESULTATEN	**15%**		
De resultaten worden gepresenteerd in een duidelijk en eenduidig te interpreteren vorm. Hierbij dient gebruik gemaakt te worden van illustraties.	8,0%	10	

In de tekst wordt verwezen naar betreffende illustraties en de benoemde resultaten.	7,0%	10	
DISCUSSIE EN CONCLUSIE	**20%**		
Er wordt aan de hand van de sleutelresultaten een antwoord gegeven op de hoofdvraag en deelvragen.	7,5%	10	
Er wordt aan de hand van de validatie zwakte- en verbeterpunten van de analyse(s) geïdentificeerd en besproken.	7,5%	10	
Er wordt een vervolgadvies opgesteld, met meerwaarde voor de opdrachtgever	5,0%	10	

Bijlage 3.2 Beoordelingsformulier toolkit

Beoordelingscriteria	Weging	Goed (3)	Voldoende (2)	Onvoldoende (1)	Opmerkingen	Score
Gebruiksvriendelijkheid	3	De gebruiker kan zonder enige moeite, met eigen kennis, gebruik maken van de tool.	De gebruiker kan gebruik maken van de tool, maar moet moeite steken om de werking te begrijpen.	De gebruiker kan niet zonder hulpmiddelen, die buitenom de tool beschikbaar zijn gesteld, gebruik maken van de tool.		
Functionaliteit	3	De tool werkt naar behoren en de gebruiker kan hier, zonder dat er technische problemen voorkomen, gebruik van maken.	De tool werkt naar behoren, maar het kan voorkomen dat kleine technische problemen zich voordoen.	De tool werkt niet naar behoren en met enige regelmaat doen zich technische problemen voor.		
Integratiemogelijkheden	3	De tool kan zonder problemen geïntegreerd worden binnen verschillende Sensor Data Platformen.	De tool kan geïntegreerd worden binnen verschillende Sensor Data Platformen, maar hier moeten enige aanpassingen voor doorgevoerd worden.	De tool kan in geen enkel Sensor Data Platform geïntegreerd worden.		
Toepasbaarheid	4	De tool geeft de gebruiker de mogelijkheid om data te visualiseren en te analyseren.	De tool geeft de gebruiker de mogelijkheid om data te visualiseren, maar niet om te analyseren.	De tool geeft de gebruiker geen mogelijkheden om data te visualiseren of te analyseren.		

Bijlage 2: Platformhandleidingen

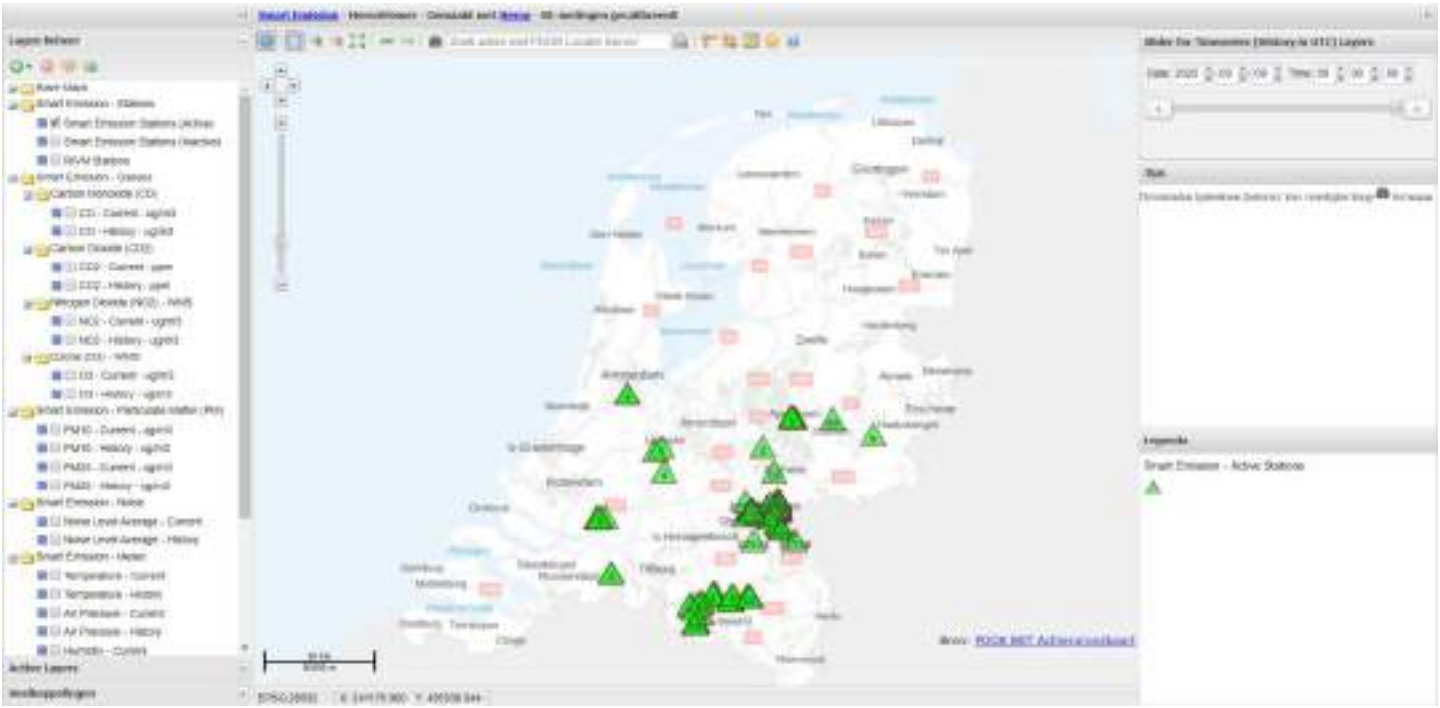
2.1 Heron Viewer

Naam: "….-…."

Functie: "….-…."

Link: https://data.smartemission.nl/heron

Bij het opstarten van de Heron viewer krijgt u het volgende beeld op uw laptop te zien:



Afbeelding 1: startscherm HeronViewer

Het platform bestaat uit een aantal belangrijke elementen:

Lagen beheer: bevindt zich links op het platform. Bij de lagen beheer wordt de mogelijkheid gegeven om met informatie binnen het platform te spelen.

De kaart: bevindt zich in het midden van het platform. Op de kaart wordt de informatie weergegeven die in de lagen beheer aan wordt gevinkt.

Tijdschaal: bevindt zich rechts bovenin het platform. Op de tijdschaal kan er een specifieke dag en tijdstip uitgekozen worden die de tijdstempel van de data vertegenwoordigd.

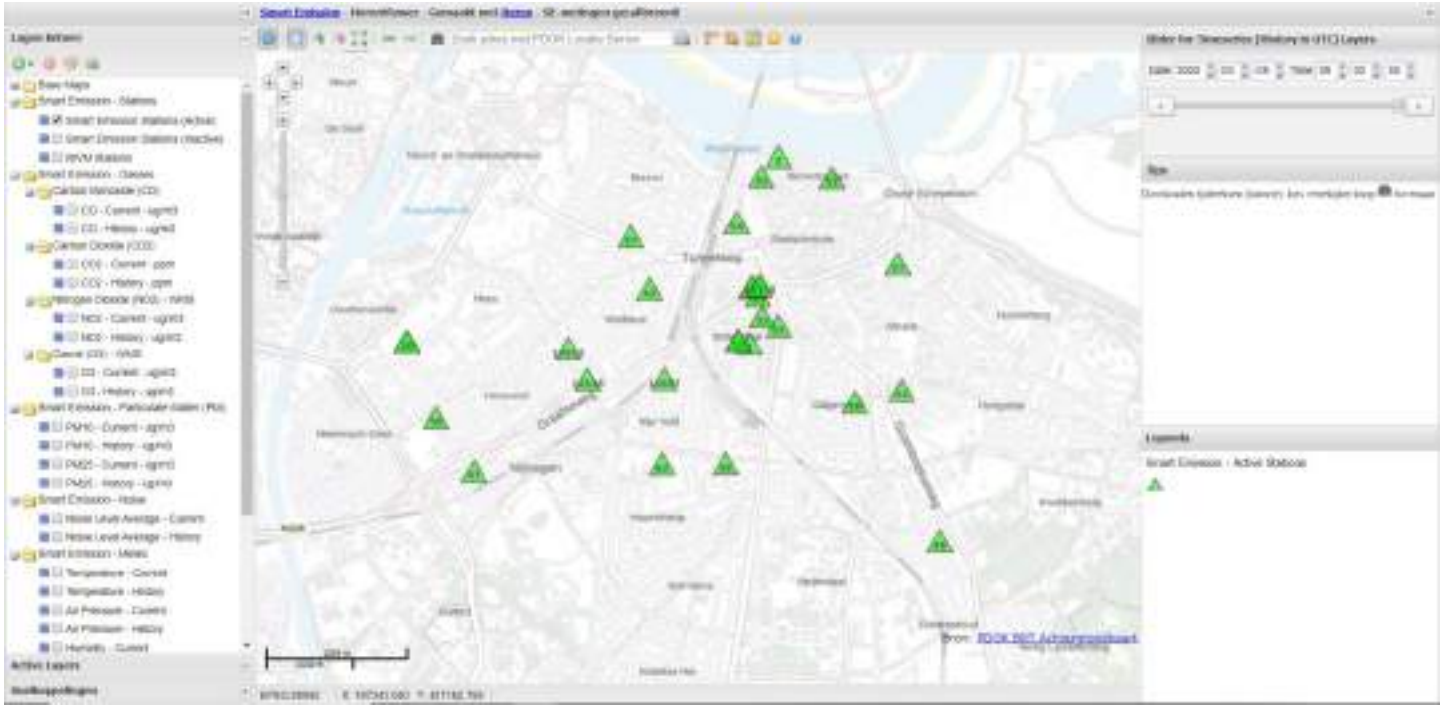
Legenda: bevindt zich rechts onderin het platform. In de legenda worden alle actieve lagen weergegeven die op dat moment te zien zijn op de kaart.

Stappenplan HeronViewer

Om meer over het platform te weten te komen, is het de bedoeling dat het volgende stappenplan stap voor stap wordt uitgevoerd. Tussen de stappen door worden vragen gesteld, die ingevuld dienen te worden.

Als er enige vragen of opmerkingen zijn, dan staan Stefan en Thomas klaar om u te assisteren.

1. Zoom in op de stad Nijmegen (zie afbeelding 2). Alle actieve sensorstations worden op de kaart weergegeven. Probeer nu te spelen met de verschillende soorten stations in het platform.



Afbeelding 2: de stad Nijmegen met alle actieve sensorstations

Vraag 1: "Wat is uw algemene eerste indruk van het platform?"

"….-…."

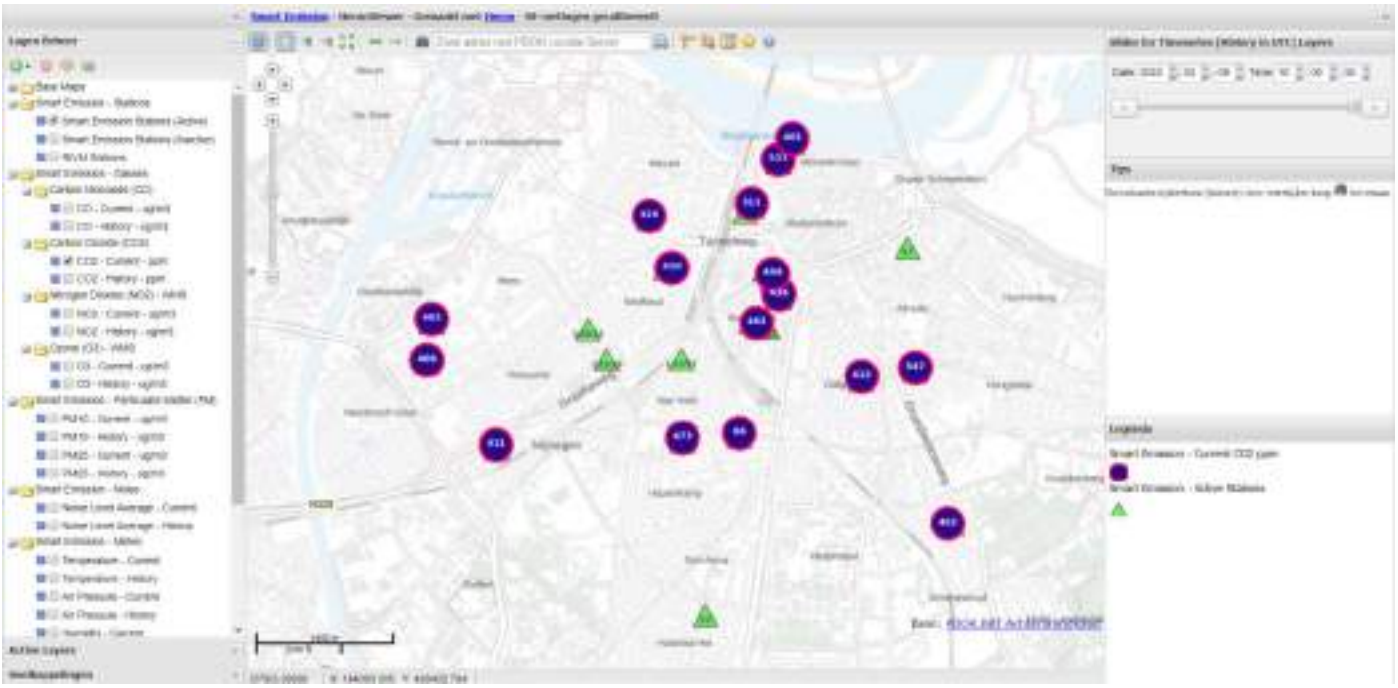
2. Zet nu alleen de actieve sensorstations aan en klik nu een willekeurig sensorstation aan. Wanneer een sensorstation wordt geselecteerd komt er een pop-up tevoorschijn met objectinformatie.

Vraag 2: "Wat vindt u van de weergegeven informatie en de huidige mogelijkheden in de objectinformatie pop-up?"

"….-…."

3. Sluit vervolgens de pop-up. Vervolgens wordt er wat dieper ingegaan op de data die wordt verzameld door de sensorstations.

Ga naar “Smart Emission – Gasses” -> “Carbon Dioxide (CO2)” -> “*CO2 – Current – ug/m3*”



Afbeelding 3: weergave huidig niveau van carbon koolstofdioxide

Vervolgens is te zien (zie afbeelding 3) dat er voor een aantal sensorstations een icoon staat die de huidige gemeten koolstofdioxide weergeeft. Klik vervolgens een willekeurige “Current CO2 ppm” icoon aan. In het objectinformatie pop-up wordt nu informatie weergegeven over twee objecten.

4. Sluit vervolgens de pop-up en deactiveer “CO2 – Current - ppm” om vervolgens de tijdschaal te gaan gebruiken. Nu moeten er een aantal dingen gebeuren:

Ga naar “Smart Emission – Gasses” -> “Carbon Dioxide (CO2)” -> “*CO2 – History – ug/m3*”

Ga vervolgens naar de tijdschaal en vul in: “Date = 2019-12-31 & time = 00:00:00”. Speel vervolgens met de datum door er een dag bij of af te doen.

Door te spelen met de tijdschaal zie je goed dat er verschillen naar voren springen bij de sensorstations.

5. Het platform heeft naast koolstofdioxide, nog veel meer data over milieuaspecten wat bekeken kan worden. Het is hierbij mogelijk om de data op datum en tijdstip te downloaden in diverse bestandformaten.

Vraag 3: “Wat vindt u van de huidige (visualisatie) mogelijkheden op het platform?”

“….-….”

U heeft nu een goede eerste impressie gekregen over de HeronViewer. U weet hoe het platform in z'n werking gaat en wat de mogelijkheden hierin zijn.

Op dit moment is het dus binnen het platform alleen mogelijk om de sensordata te visualiseren en niet te analyseren. Wanneer je met deze data analyses zou willen uitvoeren dient het eerst gedownload te worden. Eenmaal gedownload kan het geanalyseerd worden met behulp van programma's, zoals Excel.

Vraag 4: "Stel je krijgt de mogelijkheid om het platform te verbeteren, zodat het aan jouw wensen voldoet. Wat zou je dan aan het platform willen verbeteren als je deze mogelijkheid had?"

"….-…."

Opdracht:

Nu u kennis heeft gemaakt met het platform, vragen we of u enige plus- en minpunten kunt noteren over het platform. Probeer vervolgens deze plus- en minpunten te prioriteren en hier nog een korte motivatie aan toe te voegen. Uw input bij deze opdracht zal waardevol zijn en wij vragen dan ook aan u om dit zo eerlijk mogelijk in te vullen. Nogmaals bedankt voor uw tijd en moeite!

Voor het invullen van de plus- en minpunten en de motivatie, kunt u de twee tabellen invullen. Daarnaast kunt u op het einde nog enige opmerkingen of tips noteren.

Plus- en minpunten HeronViewer

Pluspunt	**Motivatie**
1.	
2.	
3.	
4.	
5.	
6.	
7.	

Minpunt	Motivatie
1.	
2.	
3.	
4.	
5.	
6.	
7.	

Opmerkingen/tips:

“….-….”

2.2 Shiny applicatie RIVM (Hollandse Luchten)

Naam: "….-…."

Functie: "….-…."

Link: https://rivm.shinyapps.io/app_HLL/

Voor u ziet u het dashboard 'Hollandse Luchten' van het RIVM. Dit platform is tot stand gekomen om dat burgers, overheden en andere partijen de mogelijkheid te geven om op een snellere manier zelf een use case (probleem) kunnen analyseren en visualiseren. Het gehele dashboard zit nog in een experimentele fase. De getoonde sensordata is afkomstig van het 'samen meten dataportaal' en worden eens in de maand geüpdatet.

Op het dashboard ziet u links in het kaartje een aantal sensoren weergegeven. Deze sensoren winnen data in dat vervolgens op dit platform wordt weergegeven. Deze data bevat gegevens over verschillende milieuaspecten die in een grafiek te visualiseren zijn. De grafiek wordt getoond, wanneer u op één van de sensoren te klikt. Hierbij is het ook mogelijk om van meerdere sensoren grafieken te laten tonen, waardoor je ze één op één met elkaar kunt vergelijken. Dit is mogelijk door een vierkant of een cirkel in de kaart te tekenen of door meerdere sensoren aan te klikken (zie figuur 1). De gemaakte selectie is te verwijderen door op het prullenbark icoontje te klikken of door nogmaals op een sensor te klikken.

Selecteer de drie sensoren bij de haven van IJmuiden (vierkant in figuur 1). Wanneer u dit doet krijgt u de drie onderstaande grafieken te zien.

Figuur 1: Startscherm met de visualisatie van diverse sensoren

Vraag 1: "Wat is uw algemene eerste indruk van het platform?"

"….-…."

We gaan de data nu goed bekijken om te zien wat er mee gedaan kan worden. Zoals hierboven al is uitgelegd kan je de sensordata bekijken en vergelijken door op verschillende sensoren te klikken. Wanneer je in het bovenstaande menu op ‘kalender’ klikt krijg je van elke dag uit de geselecteerde tijdsreeks de gemiddelde dagwaarde te zien. Deze tijdsreek is onderin het dashboard te zien en kan worden aangepast. Verschuif de tijdsreeks naar ongeveer de helft van de balk en kijk wat er met de kalender gebeurd.

Ga nu naar net kopje ‘gemiddelden’. Binnen dit kopje worden er gemiddeldes weergegeven van de geselecteerde tijdsreeks in verschillende tijdsperiodes. Deze tijdsperiodes worden weergegeven in uur, week- en maandgemiddeldes. Zo geeft bijvoorbeeld het weekgemiddelde het gemiddelde weer van alle weken samen in de geselecteerde tijdsreeks. Tot slot wordt er een tijdsperiode weergegeven, waarbinnen de gemiddelde uur waardes, uitgesplitst naar weekdagen te zien zijn. Dit alles is in de onderstaande figuur te zien.

Figuur 2: Visualisatie van diverse sensoren in diverse tijdsperiodes

Vraag 2: “Wat is uw eerste indruk van de visualisatie van de sensordata?”

“….-….”

Binnen dit dashboard heb je ook de mogelijkheid om analyses uit te voeren. Zo is het mogelijk om van een geselecteerde sensor een pollutieroos te bekijken (zie figuur 3). Deze pollutieroos geeft per windrichting het gemiddelde van de sensormetingen weer (op basis van de tijdsselectie), wanneer de wind uit die richting waait.

Daarnaast is het mogelijk om van een geselecteerde sensor een windroos te bekijken. Deze windroos geeft weer in hoeveel procent van de tijd de wind uit een bepaalde richting komt en met welke snelheid dit dan gebeurd. Door deze resultaten te vergelijken met de resultaten uit de pollutieroos kan je analyseren of er verbanden te zien zijn.

Figuur 3: Analyseren van de sensordata op basis van een pollutieroos

Vraag 3: "Wat vindt u van de huidige (visualisatie)mogelijkheden op het platform?"

"….-…."

U heeft nu een goede eerste impressie gekregen over dashboard 'Hollandse Luchten' . U weet hoe het platform in z'n werking gaat en wat de mogelijkheden hierin zijn.

Op dit moment is het dus binnen het platform mogelijk om de sensordata te visualiseren en te analyseren. De analyses kunnen uitgevoerd worden door de visualisaties (afbeeldingen) van bepaalde fenomenen, die iets kunnen vertellen over de sensorwaardes (pollutie en wind), met elkaar te vergelijken. Deze visualisaties worden door het dashboard zelf gemaakt. Het is dus niet mogelijk om met behulp van een tool deze sensordata zelf te analyseren of te visualiseren. Tot slot is het niet mogelijk om deze sensordata te downloaden.

Vraag 4: " Stel je krijgt vanuit het RIVM de mogelijkheid om het dashboard te verbeteren, zodat het aan jouw wensen voldoet. Wat zou je dan aan het platform willen verbeteren als je deze mogelijkheid had?"

"….-…."

Opdracht:

Nu u kennis heeft gemaakt met het platform, vragen we of u enige plus- en minpunten kunt noteren over het platform. Probeer vervolgens deze plus- en minpunten te prioriteren en hier nog een korte motivatie aan toe te voegen. Uw input bij deze opdracht zal waardevol zijn en wij vragen dan ook aan u om dit zo eerlijk mogelijk in te vullen. Nogmaals bedankt voor uw tijd en moeite!

Voor het invullen van de plus- en minpunten en de motivatie, kunt u de twee tabellen invullen. Daarnaast kunt u op het einde nog enige opmerkingen of tips noteren.

Plus- en minpunten Shiny applicatie RIVM

Pluspunt	Motivatie
1.	
2.	
3.	
4.	
5.	
6.	
7.	

Minpunt	Motivatie
1.	
2.	
3.	
4.	
5.	
6.	
7.	

Opmerkingen/tips:

“….-….”

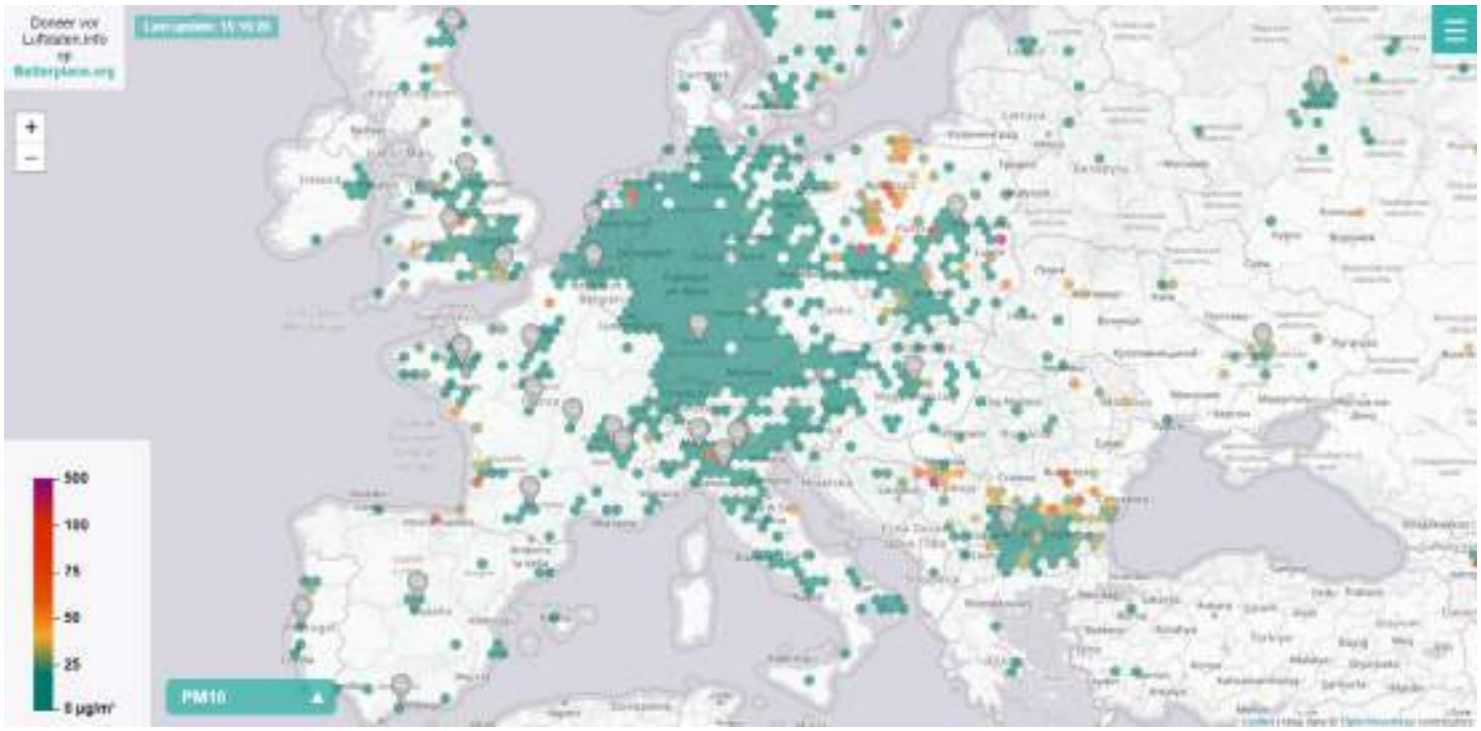
2.3 Luftdaten

Naam: "….-…."

Functie: "….-…."

Link: https://deutschland.maps.sensor.community/#6/51.165/10.455

Bij het opstarten van de Luftdaten viewer krijgt u het volgende beeld op uw laptop te zien:



Afbeelding 1: startscherm Luftdaten platform

Luftdaten is een platform waar sensorstations worden weergegeven die wereldwijd luchtkwaliteit gerelateerde data verzamelen. Het doel van het platform is om transparantie, open data en citizen science te bevorderen.

Het platform bestaat uit drie belangrijke elementen:

Hexagoon vakken: worden over de gehele kaart weergegeven. In de hexagoon vlakken worden de sensoren weergegeven. Hoe verder je uitgezoomd bent, hoe meer sensoren er in één hexagoon vlak bevinden.

De legenda: bevindt zich links onderaan het platform. In de legenda wordt per milieuaspect weer gegeven tussen welke waarden de hexagoon vakken liggen.

Milieuaspecten menu: bevindt zich links onderin naast de legenda. In het milieu aspecten menu worden alle milieuaspecten, die door de sensoren worden verzameld, weergegeven.

Vraag 1: "Wat is uw algemene eerste indruk van het platform?"

"….-…."

Stappenplan Luftdaten platform

Om meer over het platform te weten te komen, is het de bedoeling dat het volgende stappenplan stap voor stap wordt uitgevoerd. Tussen de stappen door worden vragen gesteld, die ingevuld dienen te worden.

Als er enige vragen of opmerkingen zijn, dan staan Stefan en Thomas klaar om u te assisteren.

1. Zoom in op de stad Nijmegen (zie afbeelding 2). Alle actieve sensorstations worden op de kaart weergegeven.

Afbeelding 2: de stad Nijmegen met alle actieve sensorstations

2. Klik vervolgens op een sensorstation naar keuze. Vervolgens komt er een zij-menu tevoorschijn waarin wordt weergegeven hoeveel sensoren er binnen het hexagoonvlak vallen en welke milieuaspect waarde hier bij hoort (zie afbeelding 3).

Vraag 2: "Wat is uw eerste indruk van de visualisatie van de sensordata?"

"….-…."

Afbeelding 3: hexagoon vlak informatie

3. Wanneer een hexagoonvlak is geselecteerd wordt een tabel weergegeven met de sensor(en) die zijn meegenomen binnen dit vlak. Het is vervolgens mogelijk om elke sensor individueel aan te klikken. Wanneer er een sensor aangeklikt wordt komen er twee grafieken tevoorschijn (zie afbeelding 4). Deze grafieken dienen als visualisatie van de data die bij de desbetreffende sensor hoort.

Bij de grafieken is het nog mogelijk om in te zoomen op specifieke stukken, om zo een betere inzage te krijgen op specifieke tijdstippen. Daarnaast krijg je ook de mogelijkheid om annotaties toe te voegen en de kleuren in de grafiek aan te passen.

Afbeelding 4: sensordata visualisaties

Vraag 3: "Wat vindt u van de huidige (visualisatie)mogelijkheden op het platform?"

"….-…."

U heeft nu een goede eerste impressie gekregen over de Luftdaten platform. U weet hoe het platform in z'n werking gaat en wat de mogelijkheden hierin zijn.

Op dit moment is het dus binnen het platform alleen mogelijk om de sensordata te visualiseren en niet te analyseren. Het is niet mogelijk om de sensordata te downloaden.

Vraag 4: "Stel je krijgt de mogelijkheid om het platform te verbeteren, zodat het aan jouw wensen voldoet. Wat zou je dan aan het platform willen verbeteren als je deze mogelijkheid had?"

"….-…."

Opdracht:

Nu u kennis heeft gemaakt met het platform, vragen we of u enige plus- en minpunten kunt noteren over het platform. Probeer vervolgens deze plus- en minpunten te prioriteren en hier nog een korte motivatie aan toe te voegen. Uw input bij deze opdracht zal waardevol zijn en wij vragen dan ook aan u om dit zo eerlijk mogelijk in te vullen. Nogmaals bedankt voor uw tijd en moeite!

Voor het invullen van de plus- en minpunten en de motivatie, kunt u de twee tabellen invullen. Daarnaast kunt u op het einde nog enige opmerkingen of tips noteren.

Plus- en minpunten Luftdaten

Pluspunt	**Motivatie**
1.	
2.	
3.	
4.	
5.	
6.	
7.	

Minpunt	Motivatie
1.	
2.	
3.	
4.	
5.	
6.	
7.	

Opmerkingen/tips:

“….-….”

2.4 Samen Meten

Naam: "….-…."

Functie: "….-…."

Link: https://samenmeten.rivm.nl/dataportaal/

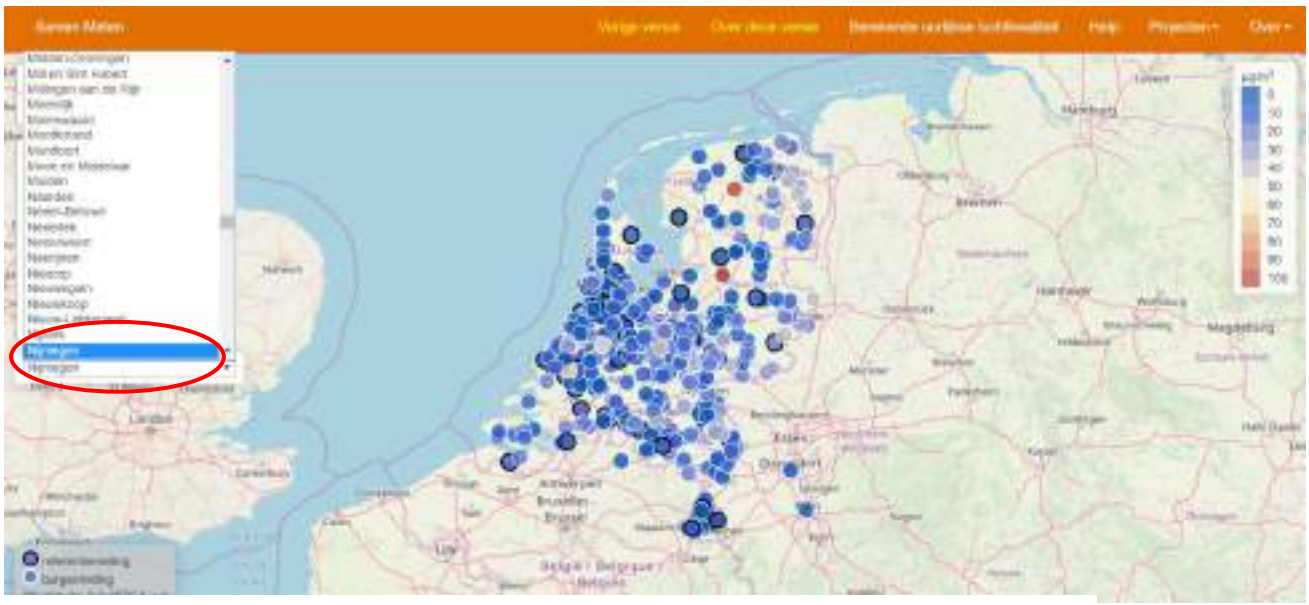
Voor u ziet u het samen meten sensor data platform van het RIVM. Dit platform is tot stand gekomen om het mogelijk te maken om sensordata van de leefomgeving gemeten door burgers, overheden en andere partijen openbaar te stellen en te visualiseren. Het gehele platform zit nog in een experimentele fase.

Op het platform ziet u een aantal sensoren weergegeven. Deze sensoren winnen data in dat vervolgens op dit platform wordt weergegeven. Deze data bevat gegevens over verschillende milieuaspecten die in dit scherm te visualiseren zijn.

Vraag 1: "Wat is uw algemene eerste indruk van het platform?"

"….-…."

We gaan deze data nu goed bekijken om te zien wat er mee gedaan kan worden. Hiervoor focussen we ons op de gemeente Nijmegen. Om hier gemakkelijk op in te zoomen is er de mogelijkheid om te focussen op de gemeente (zie figuur 1). Wanneer je op deze optie klikt komt er een lijst met gemeentes in beeld. In plaats van de gemeente Nijmegen in deze lijst te zoeken kan je de naam ook intypen. Wanneer je de gemeente Nijmegen hebt geselecteerd zoomt de kaart er automatisch op in.



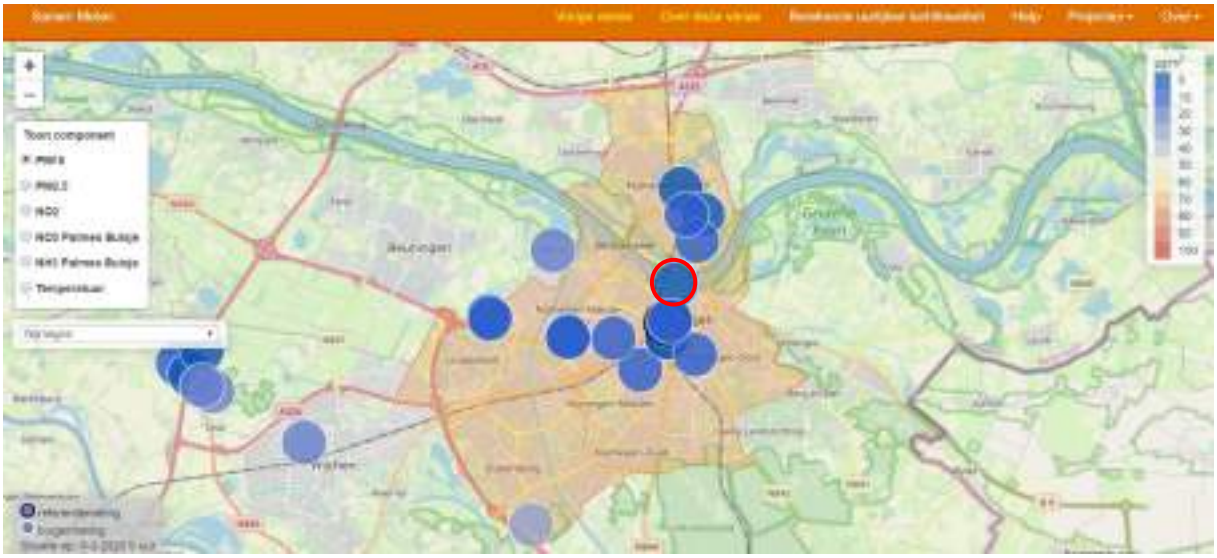
Figuur 1: Selectie op gemeente

Nadat je bent ingezoomd op de gemeente Nijmegen focussen we ons op de sensor met de rode cirkel uit figuur 2. Wanneer je op deze sensor klikt wordt een pop-up scherm geladen met de betreffende data.

Vraag 2: "Wat is uw eerste indruk van de visualisatie van de sensordata?"

"….-…."

Het platform geeft u verschillende mogelijkheden om de data te bekijken. Zo is het mogelijk om de tijdsspan van de grafiek te veranderen, de grafiek te downloaden in verschillende bestandstypen, verschillende milieuaspecten te bekijken en om de gegevens te vergelijken met referentiestations (zie figuur 3). Bekijk nu zelf deze opties om te zien hoe deze (visualisatie)mogelijkheden in z'n werking gaan.



Figuur 2: Selectie op sensor

Figuur 3: Data visualisatie van sensordata

Vraag 3: "Wat vindt u van de huidige (visualisatie)mogelijkheden op het platform?"

"….-…."

U heeft nu een goede eerste impressie gekregen over het samen meten platform. U weet hoe het platform in z'n werking gaat en wat de mogelijkheden hierin zijn.

Op dit moment is het dus binnen het platform alleen mogelijk om de sensordata te visualiseren en niet te analyseren. Wanneer je met deze data analyses zou willen uitvoeren dient dit te worden uitgevoerd in andere programma's, zoals Excel.

Vraag 4: " Stel je krijgt vanuit het RIVM de mogelijkheid om het platform te verbeteren, zodat het aan jouw wensen voldoet. Wat zou je dan aan het platform willen verbeteren als je deze mogelijkheid had?"

"….-…."

Opdracht:

Nu u kennis heeft gemaakt met het platform, vragen we of u enige plus- en minpunten kunt noteren over het platform. Probeer vervolgens deze plus- en minpunten te prioriteren en hier nog een korte motivatie aan toe te voegen. Uw input bij deze opdracht zal waardevol zijn en wij vragen dan ook aan u om dit zo eerlijk mogelijk in te vullen. Nogmaals bedankt voor uw tijd en moeite!

Voor het invullen van de plus- en minpunten en de motivatie, kunt u de twee tabellen invullen. Daarnaast kunt u op het einde nog enige opmerkingen of tips noteren.

Plus- en minpunten Samen meten

Pluspunt	**Motivatie**
1.	
2.	
3.	
4.	
5.	
6.	
7.	

Minpunt	Motivatie
1.	
2.	
3.	
4.	
5.	
6.	
7.	

Opmerkingen/tips:

“….-….”

Bijlage 3: Documentatie packages RStudio

In deze bijlage worden de packages die binnen RStudio zijn gebruikt tijdens de ontwikkeling van de toolbox verder toegelicht.

Packages:

devtools: Het R-pakket 'devtools' bevat verschillende functies die binnen RStudio veel worden gebruikt.

dplyr: dplyr is een R-pakket voor het werken met data.frame objecten. Zo is het mogelijk om objecten te filteren, selecteren en samen te vatten.

DT: Deze functie creëert een HTML Widget om data te visualiseren (matrix of data.frame), door gebruik te maken van de Javascript 'DataTables' Library (bibliotheek).

geoshaper: geoshaper is een package die vanuit GitHub is te installeren en verschillende functies bevat om vormen in een kaart te tekenen. Met behulp van deze vormen is het mogelijk om data in de kaart te selecteren.

ggplot2: ggplot2 is een systeem, waarmee het mogelijk is om grafieken te genereren. Zelf geef je aan op welke manier de data gevisualiseerd dient te worden en in welke geografische vorm en het systeem genereert de grafiek.

htmltools: Het R-pakket 'htmltools' bevat tools voor het generen en uitvoeren van HTML. Een voorbeeld hiervan is om met behulp van HTML de paginagrootte van de applicatie aan te geven.

htmlwidgets: Het R-pakket 'htmlwidgets' bevat tools voor het gebruiken van Javascript visualisatiemogelijkheden binnen RStudio.

httr: httr is een R-pakket waarmee URL's en HTTP kan worden binnengehaald. Op deze manier is het dus ook mogelijk om API's (HTTP verzoek) binnen te halen.

jsonlite: jsonlite is een R-pakket waarmee JSON data kan worden omgezet naar R objecten. Op deze manier is het mogelijk om data uit een API in te lezen en om te zetten naar objecten die binnen RStudio bewerkt kunnen worden.

Leaflet: Leaflet is een R-pakket waarmee interactieve kaarten gemaakt kunnen worden met behulp van de Javascript 'Leaflet' Library (bibliotheek). Deze interactieve kaarten kunnen worden gebruikt binnen de gehele R software (Shiny applicaties, RStudio en R Markdown).

Leaflet.extras: Het R-pakket 'Leaflet.extras' bevat extra functionaliteiten voor de Leaflet kaart. Deze functies worden vanuit de Javascript 'Leaflet' Library (bibliotheek) beschikbaar gesteld.

Markdown: Met behulp van het R-pakket 'Markdown' is het mogelijk om je analyses en resultaten om te zetten naar hoogstaande documenten, rapporten, presentaties en dashboards.

Openair: Openair is een R-pakket dat verschillende tools bevat om luchtvervuilingsdata te analyseren, interpreteren en begrijpen. Veel functies kunnen ook worden toegepast voor andere doeleinden, zoals: meteorologische -en verkeer data.

plotly: Het R-pakket 'plotly' bevat functies, waarmee grafieken uit ggplot2 interactief gemaakt kunnen worden.

plyr: Het R-pakket 'plyr' bevat verschillende tools, die veelvoorkomende problemen binnen RStudio kan oplossen. Wanneer grote problemen zich voordoen dienen deze eerst in stukken te worden gesplitst, voordat er gebruik kan worden gemaakt van de tools.

purrr: purrr bevat een complete set met tools, die gebruikt kunnen worden bij het werken met verschillende functies en vectoren. Zo bevat het diverse kaartfuncties en functies om lijsten aan te passen.

Shiny: Shiny is een R-pakket waarmee u eenvoudig rechtstreeks vanuit R interactieve webapps kunt bouwen. Binnen Shiny heb je verschillende functies die gepubliceerd kunnen worden, zoals interactieve grafieken- en kaarten. Hiernaast kan je de Shiny applicaties ook uitbreiden met CSS-thema's, html-widgets en Javascript acties.

shinythemes: shinythemes bevat verschillende thema's die gebruikt kunnen worden binnen Shiny. Zo is het onder andere mogelijk om verschillende Bootstrap pagina's te gebruiken.

shinyWidgets: shinyWidgets is een R-pakket dat diverse inputbesturingen en interfaces bevat voor Shiny applicaties. Hiermee kan je je Shiny applicatie een nieuwe lay-out en interface geven.

sp: sp is een R-pakket dat verschillende klassen en methodes bevat om met ruimtelijke (GIS) data te werken. Klassen vertellen wat over waar de ruimtelijke data zich moet bevinden en wat voor data het is (2D, 3D). Methodes zijn functies die toegepast kunnen worden, zoals het ophalen van coördinaten en het uitvoeren van een ruimtelijke selectie.

tidyr: tidyr is een pakket dat is ontworpen om op een eenvoudige manier datasets te hervormen. Zo is het bijvoorbeeld mogelijk om meerdere kolommen samen te voegen.

Bijlage 4: Opgestelde R-scripts

4.1 Luftdaten API

In dit script wordt sensordata uit de Luftdaten API ingeroepen en bewerkt. De API bevat de actuele meetgegevens (pm10 & pm2.5) van alle Luftdaten sensoren in Europa. Het script bestaat uit een aantal belangrijke stappen die in dit document verder worden toegelicht.

Het eindresultaat: Een tabel, waarin de meetwaardes pm10 & pm2.5 van alle Luftdaten sensoren in Nederland overzichtelijk in staan weergegeven.

1. Benodigde pakketten installeren

Voordat er binnen RStudio gewerkt kan worden met de API's is het van belang dat alle benodigde pakketen zijn geïnstalleerd (functie: 'install.packages') en geactiveerd (functie: 'library').

```
library("jsonlite")
library("httr")
library("dplyr")
library("tidyr")
library("leaflet")
```

2. Inlezen API URL

Met behulp van een URL kan de API worden ingelezen. Binnen de link kan er worden aangegeven hoe en welke data ingelezen dient te worden.

```
Linkluftdaten <- "http://api.luftdaten.info/static/v1/data.json"
```

3. URL plakken in de workspace

Met de functie paste wordt de URL in de workspace geplakt.

```
pastelink <- paste(Linkluftdaten)
```

4. Data binnenhalen uit de URL

Met de functie GET geef je aan dat je de data uit de Luftdaten API URL binnengehaald dient te worden. De Content type dient application/json te zijn, anders kan de data binnen RStudio niet worden bewerkt. Wanneer de status 200 is betekend het dat de data correct is ingeladen.

```
getdataluftd <- GET(pastelink)
```

5. Data omzetten naar tekstbestand

Met de functie content -> text geef je aan dat de data omgezet moet worden naar tekstbestand.

```
get_data_luftd_text <- content(getdataluftd,"text")
```

6. JSON bestand omzetten naar R bestand

Met de functie fromJSON wordt een JSON bestand omgezet naar een R bestand. Door deze omzetting kan de data in RStudio bewerkt worden.

```
get_data_luftd_json <- fromJSON(get_data_luftd_text, flatten = TRUE)
```

7.Data.frame aanmaken

Door de omzetting kan er van de data een overzichtelijke tabel worden gemaakt.
De tabel wordt gemaakt met behulp van de functie as.data.frame. De overzichtelijke tabel maakt het eenvoudiger om de data te bewerken.

```
get_data_luftd_df <- as.data.frame(get_data_luftd_json)
```

8.Sensoren filteren binnen Nederland

Bij de filter functie worden de sensoren binnen Nederland uit de dataset gefilterd.

```
sensoren_Nederland_filter <- filter(get_data_luftd_df,location.country == "NL")
```

9.Omzetten data type coördinaten

Bij de transform functie worden de coordinaten omgezet van tekstwaardes naar numerieke waardes.

```
omzetting_coordinaten <- transform(sensoren_Nederland_filter, location.longitude = as.numeric(locat
ion.longitude), location.latitude = as.numeric(location.latitude))
```

10.Data.frame met meetwaardes omzetten naar een list

Met behulp van de onderstaande code wordt de data.frame met meetwaardes in de huidige tabel gezet. Eerst wordt de data.frame omgezet naar een matrix. Vervolgens worden de kolommen (id, value, value_type) uitgepakt (extract) en omgezet naar een list dat in de tabel te vinden is. Nu staat alle data van de API in één tabel weergegeven.

```
tabel_API_overzicht<- omzetting_coordinaten %>%
 dplyr::mutate(sensordatavalues = matrix(sensordatavalues),
          id = sapply(sensordatavalues, function(x) magrittr::extract2(x, 1)),
          value = sapply(sensordatavalues, function(x) magrittr::extract2(x, 2)),
          value_type = sapply(sensordatavalues, function(x) magrittr::extract2(x, 3)),
          sensordatavalues = NULL)
```

11.Verwijderen onnodige kolommen

In de onderstaande code worden onnodige kolommen uit de dataset gefilterd.

```
Kolommen_filter = subset(tabel_API_overzicht, select = -c(sampling_rate,location.exact_location, id, location.altitude, location.indoor, sensor.pin, sensor.sensor_type.id))
```

12. Veranderen kolomnamen

Met de functie colnames worden de kolomnamen verandert naar de namen die in de R code van het Hollandse luchten platform worden gebruikt.

```
colnames(Kolommen_filter)[2] <- "kit_id"
colnames(Kolommen_filter)[3] <- "lat"
colnames(Kolommen_filter)[4] <- "lon"
colnames(Kolommen_filter)[10] <- "Var"
colnames(Kolommen_filter)[1] <- "date"
```

13. Verwijderen onnodige meetwaardes

Bij de onderstaande code worden de meetwaardes die pm10 en pm2.5 bevatten uit de dataset gefilterd. Dit wordt gedaan door de rijen op te geven die de values temperature en humidity bevatten. Deze rijen worden uit de dataset verwijderd.

```
meetwaardes_filter <- dplyr::filter(Kolommen_filter, !grepl('temperature|humidity', Var))
```

14. Omzetting één kolom met pm10 en pm2.5 waardes naar twee apparte kolommen

Bij de onderstaande code worden de values van P1 (pm10) en p2 (pm2.5) uit één kolom omgezet naar twee kolommen met de naam P1 en P2. De scheiding hierbij vindt plaats bij de , .

```
kolommen_p1_p2 <- separate(meetwaardes_filter, col=value, into = c("P1","P2"), sep = ",")
```

15. Verwijderen onnodige tekens kolom P1

Bij de onderstaande code worden de tekens c(uit de kolom P1 verwijderd. Dit wordt gerealiseerd door het gedeelte rechts van het teken (te splitsen van de betreffende waardes. Kolom P1 wordt vervolgens weggeschreven als een aparte tabel.

```
P1 <- sapply(strsplit(kolommen_p1_p2$P1, split='(', fixed=TRUE), function(x) (x[2]))
```

Bij de onderstaande code worden de tekens "…" uit de kolom P1 verwijderd. Dit wordt gerealiseerd door de tekens "…" uit de dataset te splitsen (filteren) van de betreffende waardes. Het verwijderen van deze tekens is nodig om de factor waardes om te kunnen zetten naar numerieke waardes.

```
P1 <- sapply(strsplit(P1, split='"', fixed=TRUE), function(x) (x[2]))
```

16. Omzetting kolom P1

Bij de onderstaande code worden de factor waardes omgezet naar numerieke waardes.

```
P1 <- as.numeric(P1)
```

Met de functie cbind worden de twee data.frames aan elkaar gekoppeld. Het gaat om de data.frames P1 en kolommen_p1_p2.

```
Luftdaten_P1 <- cbind(kolommen_p1_p2, P1)
```

17. Verwijderen onnodige tekens kolom P2

Bij de onderstaande code worden de tekens) uit de kolom P2 verwijderd. Dit wordt gerealiseerd door het gedeelte links van het teken) te splitsen van de betreffende waardes. Kolom P2 wordt vervolgens weggeschreven als een aparte tabel.

```
P2 <- sapply(strsplit(kolommen_p1_p2$P2, split=')', fixed=TRUE), function(x) (x[1]))
```

Bij de onderstaande code worden de tekens "…" uit de kolom P2 verwijderd. Dit wordt gerealiseerd door de tekens "…" uit de dataset te splitsen (filteren) van de betreffende waardes. Het verwijderen van deze tekens is nodig om de factor waardes om te kunnen zetten naar numerieke waardes.

```
P2 <- sapply(strsplit(P2, split='"', fixed=TRUE), function(x) (x[2]))
```

18. Omzetting kolom P2

Bij de onderstaande code worden de factor waardes omgezet naar numerieke waardes

```
P2 <- as.numeric(P2)
```

Met de functie cbind worden de twee data.frames aan elkaar gekoppeld Het gaat om de data.frames P2 en Luftdaten_P1

```
Luftdaten_P1_P2 <- cbind(Luftdaten_P1, P2)
```

19. Verwijderen onnodige kolommen

In de onderstaande code wordt de onnodige kolom P1, P2 en Var uit de dataset verwijderd. Dit gebeurd op basis van de positie binnen de tabel.

```
Luftdaten_API <- Luftdaten_P1_P2[,- (9:11)]
```

20. Verwijderen foutieve data

In de onderstaande code worden alle coordinaten met 0.0000 uit de dataset verwijderd.

```
Luftdaten_API <- Luftdaten_API[grep('^[1-9]', Luftdaten_API$lat),]
```

In de onderstaande code worden alle NA waardes uit de dataset verwijderd

```
Luftdaten_API <- Luftdaten_API[grep('^[0-9]', Luftdaten_API$P1),]
```

In de onderstaande code worden alle dubbele rijen op basis van sensor.id uit de dataset verwijderd. Hierdoor blijft van elke sensor.id alleeen de meeste actuele meetwaarde over.

```
Luftdaten_API <- distinct(Luftdaten_API, sensor.id, .keep_all = TRUE)
```

21. Data opslaan als RDS bestand

Met behulp van de onderstaande code wordt de data opgeslagen als rds bestand (Rstudio).

```
saveRDS(Luftdaten_API, "API_Luftdaten.rds")
```

22. Opstellen interactieve kaart

Met behulp van de functie leaflet wordt er snel een interactieve kaart gemaakt. Door middel van deze kaart kan er snel worden bekeken of de data op de juiste manier wordt gevisualiseerd en of er fouten inzitten.

```
mymap <- leaflet(data = Luftdaten_API) %>% addTiles() %>%
addMarkers(lat = ~lat, lng = ~lon,
popup = paste("ID", Luftdaten_API$P2))
mymap
```

4.2 Samen Meten API

In dit script wordt sensordata uit de Samen Meten API ingeroepen, samengevoegd en bewerkt. Er wordt één sensor ingeroepen, waarvan de desbetreffende PM10, PM2,5 en gekalibreerde waardes mee zijn genomen. Het script bestaat uit een aantal belangrijke stappen, deze zijn als volgt:

1. Benodigde pakketten installeren
2. Inlezen van de API URL's
3. URL's in verschillende workspaces plakken
4. Data uit de URL's binnenhalen
5. De databestanden omzetten naar tekstbestanden
6. De tekstbestanden omzetten naar RStudio bestanden
7. De RStudio bestanden omzetten naar bewerkbare tabellen
8. De desbetreffende tabellen bijwerken
9. Samenvoegen van de tabellen
10. Laatste bewerkingen kolommen / bijhorende waarden in de tabellen
11. Opslaan van de dataset

Het eindresultaat: Een overzichtelijke tabel voor één sensor waarvan de geregistreerde fijnstof waardes zijn meegenomen. Om meer sensoren te gebruiken kan dit script handmatig aangepast worden. Hoe dit gedaan wordt is gedocumenteerd in de desbetreffende einddocumentatie.

1. Benodigde pakketten installeren

Voordat er binnen RStudio gewerkt kan worden met de API's is het van belang dat alle benodigde pakketen zijn ge?nstalleerd (functie: 'install.packages') en geactiveerd (functie: 'library').

```
library(httr)
library(jsonlite)
library(tidyr)
library(plyr)
library(dplyr)
```

2. Inlezen van de API URL's

Om de benodigde data te verzamelen die verwerkt moet worden in de applicatie zullen er een aantal API URL's opgeroepen moeten worden (functie: 'value' <- "URL"). Afhankelijk van het soort API moeten er meerdere calls uitgevoerd worden, wat in dit geval van toepassing is.

```
Infor1 <- "https://api-samenmeten.rivm.nl/v1.0/Things(30017)"
Infor2 <- "https://api-samenmeten.rivm.nl/v1.0/Things(30017)/Locations"

# pm25 data
Infor3 <- "https://api-samenmeten.rivm.nl/v1.0/Datastreams(104771)/Observations"
Infor4 <- "https://api-samenmeten.rivm.nl/v1.0/Datastreams(104771)/Observations?$top=20&$skip=
```

```
20"
Infor5 <- "https://api-samenmeten.rivm.nl/v1.0/Datastreams(104771)/Observations?$top=20&$skip=
40"
Infor6 <- "https://api-samenmeten.rivm.nl/v1.0/Datastreams(104771)/Observations?$top=20&$skip=
60"
Infor7 <- "https://api-samenmeten.rivm.nl/v1.0/Datastreams(104771)/ObservedProperty"

# pm10 data
Infor8 <- "https://api-samenmeten.rivm.nl/v1.0/Datastreams(104773)/Observations"
Infor9 <- "https://api-samenmeten.rivm.nl/v1.0/Datastreams(104773)/Observations?$top=20&$skip=
20"
Infor10 <- "https://api-samenmeten.rivm.nl/v1.0/Datastreams(104773)/Observations?$top=20&$skip
=40"
Infor11 <- "https://api-samenmeten.rivm.nl/v1.0/Datastreams(104773)/Observations?$top=20&$skip
=60"
Infor12 <- "https://api-samenmeten.rivm.nl/v1.0/Datastreams(104773)/ObservedProperty"

# pm25 kal data
Infor13 <- "https://api-samenmeten.rivm.nl/v1.0/Datastreams(104772)/Observations"
Infor14 <- "https://api-samenmeten.rivm.nl/v1.0/Datastreams(104772)/Observations?$top=20&$skip
=20"
Infor15 <- "https://api-samenmeten.rivm.nl/v1.0/Datastreams(104772)/Observations?$top=20&$skip
=40"
Infor16 <- "https://api-samenmeten.rivm.nl/v1.0/Datastreams(104772)/Observations?$top=20&$skip
=60"
Infor17 <- "https://api-samenmeten.rivm.nl/v1.0/Datastreams(104772)/ObservedProperty"

# pm10 kal data
Infor18 <- "https://api-samenmeten.rivm.nl/v1.0/Datastreams(104774)/Observations"
Infor19 <- "https://api-samenmeten.rivm.nl/v1.0/Datastreams(104774)/Observations?$top=20&$skip
=20"
Infor20 <- "https://api-samenmeten.rivm.nl/v1.0/Datastreams(104774)/Observations?$top=20&$skip
=40"
Infor21 <- "https://api-samenmeten.rivm.nl/v1.0/Datastreams(104774)/Observations?$top=20&$skip
=60"
Infor22 <- "https://api-samenmeten.rivm.nl/v1.0/Datastreams(104774)/ObservedProperty"
```

3. URL's in verschillende workspaces plakken

Vervolgens is het de bedoeling dat alle API URL links worden verplaatst in een verschillende workspaces. In dit geval is 'pastelink' een toepasselijke benaming voor het plakken van de diverse API calls.

```
pastelink1 <- paste(Infor1)
pastelink2 <- paste(Infor2)

# pm25 data
pastelink3 <- paste(Infor3)
pastelink4 <- paste(Infor4)
pastelink5 <- paste(Infor5)
pastelink6 <- paste(Infor6)
pastelink7 <- paste(Infor7)
```

```
# pm10 data
pastelink8 <- paste(Infor8)
pastelink9 <- paste(Infor9)
pastelink10 <- paste(Infor10)
pastelink11 <- paste(Infor11)
pastelink12 <- paste(Infor12)

# pm25 kal data
pastelink13 <- paste(Infor13)
pastelink14 <- paste(Infor14)
pastelink15 <- paste(Infor15)
pastelink16 <- paste(Infor16)
pastelink17 <- paste(Infor17)

# pm10 kal data
pastelink18 <- paste(Infor18)
pastelink19 <- paste(Infor19)
pastelink20 <- paste(Infor20)
pastelink21 <- paste(Infor21)
pastelink22 <- paste(Infor22)
```

4. Data uit de URL's binnenhalen

Vervolgens is het de bedoeling dat de benodigde data uit de API calls worden gehaald. Om dit uit te voeren wordt de functie 'GET' gebruikt. Met de functie GET geef je aan wele dat je uit de API call1 (link) wilt binnenhalen. Het content type dient application/json te zijn, anders kan er niet mee worden gewerkt. Bij status: '200' is de data correct ingeladen.

```
thing <- GET(pastelink1)
location <- GET(pastelink2)

# pm25 data
obser_pm25 <- GET(pastelink3)
obser2_pm25 <- GET(pastelink4)
obser3_pm25 <- GET(pastelink5)
obser4_pm25 <- GET(pastelink6)
observ_prop_pm25 <- GET(pastelink7)

# pm10 data
obser_pm10 <- GET(pastelink8)
obser2_pm10 <- GET(pastelink9)
obser3_pm10 <- GET(pastelink10)
obser4_pm10 <- GET(pastelink11)
observ_prop_pm10 <- GET(pastelink12)

# pm25 kal data
obser_pm25_kal <- GET(pastelink13)
obser2_pm25_kal <- GET(pastelink14)
obser3_pm25_kal <- GET(pastelink15)
obser4_pm25_kal <- GET(pastelink16)
observ_prop_pm25_kal <- GET(pastelink17)
```

```
# pm10 kal data
obser_pm10_kal <- GET(pastelink18)
obser2_pm10_kal <- GET(pastelink19)
obser3_pm10_kal <- GET(pastelink20)
obser4_pm10_kal <- GET(pastelink21)
observ_prop_pm10_kal <- GET(pastelink22)
```

5. De databestanden omzetten naar tekstbestanden

Met de functie 'content(dataset, "text")' geef je aan dat de data omgezet moet worden naar een tekstbestand.

```
thing_sensor <- content(thing,"text")
location_sensor <- content(location,"text")

# pm25 data
obser_pm25_sensor <- content(obser_pm25, "text")
obser2_pm25_sensor <- content(obser2_pm25, "text")
obser3_pm25_sensor <- content(obser3_pm25, "text")
obser4_pm25_sensor <- content(obser4_pm25, "text")
observprop_pm25_sensor <- content(observ_prop_pm25, "text")

# pm10 data
obser_pm10_sensor <- content(obser_pm10, "text")
obser2_pm10_sensor <- content(obser2_pm10, "text")
obser3_pm10_sensor <- content(obser3_pm10, "text")
obser4_pm10_sensor <- content(obser4_pm10, "text")
observprop_pm10_sensor <- content(observ_prop_pm10, "text")

# pm25 kal data
obser_pm25_kal_sensor <- content(obser_pm25_kal, "text")
obser2_pm25_kal_sensor <- content(obser2_pm25_kal, "text")
obser3_pm25_kal_sensor <- content(obser3_pm25_kal, "text")
obser4_pm25_kal_sensor <- content(obser4_pm25_kal, "text")
observprop_pm25_kal_sensor <- content(observ_prop_pm25_kal, "text")

# pm10 kal data
obser_pm10_kal_sensor <- content(obser_pm10_kal, "text")
obser2_pm10_kal_sensor <- content(obser2_pm10_kal, "text")
obser3_pm10_kal_sensor <- content(obser3_pm10_kal, "text")
obser4_pm10_kal_sensor <- content(obser4_pm10_kal, "text")
observprop_pm10_kal_sensor <- content(observ_prop_pm10_kal, "text")
```

6. De tekstbestanden omzetten naar RStudio bestanden

Om verder te kunnen werken met de data is het van belang dat de JSON bestanden worden omgezet naar R bestanden. Met de functie 'fromJSON' wordt dit gewaarborgd.

```
thing_json <- fromJSON(thing_sensor, flatten = TRUE)
location_json <- fromJSON(location_sensor, flatten = TRUE)

# pm25 data
```

```
obser_pm25_json <- fromJSON(obser_pm25_sensor, flatten = TRUE)
obser2_pm25_json <- fromJSON(obser2_pm25_sensor, flatten = TRUE)
obser3_pm25_json <- fromJSON(obser3_pm25_sensor, flatten = TRUE)
obser4_pm25_json <- fromJSON(obser4_pm25_sensor, flatten = TRUE)
observprop_pm25_json <- fromJSON(observprop_pm25_sensor, flatten = TRUE)

# pm10 data
obser_pm10_json <- fromJSON(obser_pm10_sensor, flatten = TRUE)
obser2_pm10_json <- fromJSON(obser2_pm10_sensor, flatten = TRUE)
obser3_pm10_json <- fromJSON(obser3_pm10_sensor, flatten = TRUE)
obser4_pm10_json <- fromJSON(obser4_pm10_sensor, flatten = TRUE)
observprop_pm10_json <- fromJSON(observprop_pm10_sensor, flatten = TRUE)

# pm25 kal data
obser_pm25_kal_json <- fromJSON(obser_pm25_kal_sensor, flatten = TRUE)
obser2_pm25_kal_json <- fromJSON(obser2_pm25_kal_sensor, flatten = TRUE)
obser3_pm25_kal_json <- fromJSON(obser3_pm25_kal_sensor, flatten = TRUE)
obser4_pm25_kal_json <- fromJSON(obser4_pm25_kal_sensor, flatten = TRUE)
observprop_pm25_kal_json <- fromJSON(observprop_pm25_kal_sensor, flatten = TRUE)

# pm10 kal data
obser_pm10_kal_json <- fromJSON(obser_pm10_kal_sensor, flatten = TRUE)
obser2_pm10_kal_json <- fromJSON(obser2_pm10_kal_sensor, flatten = TRUE)
obser3_pm10_kal_json <- fromJSON(obser3_pm10_kal_sensor, flatten = TRUE)
obser4_pm10_kal_json <- fromJSON(obser4_pm10_kal_sensor, flatten = TRUE)
observprop_pm10_kal_json <- fromJSON(observprop_pm10_kal_sensor, flatten = TRUE)
```

7. De RStudio bestanden omzetten naar bewerkbare tabellen

Door de omzetting kan er een tabel worden gemaakt van de data. De tabel wordt gemaakt met behulp van de functie ‘as.data.frame’. Er moet eerst een dataframe worden gemaakt voordat de data verder bewerkt (gefilterd kan worden).

```
thing_data <- as.data.frame(thing_json)
location_data <- as.data.frame(location_json)

# pm25 data
obser_pm25_data <- as.data.frame(obser_pm25_json)
obser2_pm25_data <- as.data.frame(obser2_pm25_json)
obser3_pm25_data <- as.data.frame(obser3_pm25_json)
obser4_pm25_data <- as.data.frame(obser4_pm25_json)
observprop_pm25_data <- as.data.frame(observprop_pm25_json)

# pm10 data
obser_pm10_data <- as.data.frame(obser_pm10_json)
obser2_pm10_data <- as.data.frame(obser2_pm10_json)
obser3_pm10_data <- as.data.frame(obser3_pm10_json)
obser4_pm10_data <- as.data.frame(obser4_pm10_json)
observprop_pm10_data <- as.data.frame(observprop_pm10_json)

# pm25 kal data
obser_pm25_kal_data <- as.data.frame(obser_pm25_kal_json)
```

```
obser2_pm25_kal_data <- as.data.frame(obser2_pm25_kal_json)
obser3_pm25_kal_data <- as.data.frame(obser3_pm25_kal_json)
obser4_pm25_kal_data <- as.data.frame(obser4_pm25_kal_json)
observprop_pm25_kal_data <- as.data.frame(observprop_pm25_kal_json)

# pm10 kal data
obser_pm10_kal_data <- as.data.frame(obser_pm10_kal_json)
obser2_pm10_kal_data <- as.data.frame(obser2_pm10_kal_json)
obser3_pm10_kal_data <- as.data.frame(obser3_pm10_kal_json)
obser4_pm10_kal_data <- as.data.frame(obser4_pm10_kal_json)
observprop_pm10_kal_data <- as.data.frame(observprop_pm10_kal_json)
```

8. De desbetreffende tabellen bijwerken

Om vervolgens onnodige informatie weg te laten en alleen de noodzakelijke elementen te behouden, wordt het makkelijker om met de data te werken. Het werkt als volgt: ’ df = subset(mydata, select = -c(‘.’ , ‘.’)) ’ Binnen ‘-c()’ wordt aangegeven welke kolommen weggelaten moeten worden. Daarnaast moet de betreffende dataset ingevuld worden bij ‘mydata’.

```
df  = subset(thing_data, select = -c(description, properties.project, X.iot.id, X.iot.selfLink, Locations.iot.navigationLink, Datastreams.iot.navigationLink, HistoricalLocations.iot.navigationLink))
df2 = subset(location_data, select = -c(value..iot.id, value.name, value.location.type, value..iot.selfLink, value.description, value.Things.iot.navigationLink, value.HistoricalLocations.iot.navigationLink))

# pm25 data
df3 = subset(obser_pm25_data, select = -c(X.iot.nextLink, value..iot.id, value..iot.selfLink, value.Datastream.iot.navigationLink, value.FeatureOfInterest.iot.navigationLink, value.resultTime))
df4 = subset(obser2_pm25_data, select = -c(X.iot.nextLink, value..iot.id, value..iot.selfLink, value.Datastream.iot.navigationLink, value.FeatureOfInterest.iot.navigationLink, value.resultTime))
df5 = subset(obser3_pm25_data, select = -c(X.iot.nextLink, value..iot.id, value..iot.selfLink, value.Datastream.iot.navigationLink, value.FeatureOfInterest.iot.navigationLink, value.resultTime))
df6 = subset(obser4_pm25_data, select = -c(X.iot.nextLink, value..iot.id, value..iot.selfLink, value.Datastream.iot.navigationLink, value.FeatureOfInterest.iot.navigationLink, value.resultTime))
df7 = subset(observprop_pm25_data, select = -c(X.iot.id, X.iot.selfLink, name, definition, Datastreams.iot.navigationLink))

# pm10 data
df8 = subset(obser_pm10_data, select = -c(X.iot.nextLink, value..iot.id, value..iot.selfLink, value.Datastream.iot.navigationLink, value.FeatureOfInterest.iot.navigationLink, value.resultTime))
df9 = subset(obser2_pm10_data, select = -c(X.iot.nextLink, value..iot.id, value..iot.selfLink, value.Datastream.iot.navigationLink, value.FeatureOfInterest.iot.navigationLink, value.resultTime))
df10 = subset(obser3_pm10_data, select = -c(X.iot.nextLink, value..iot.id, value..iot.selfLink, value.Datastream.iot.navigationLink, value.FeatureOfInterest.iot.navigationLink, value.resultTime))
df11 = subset(obser4_pm10_data, select = -c(X.iot.nextLink, value..iot.id, value..iot.selfLink, value.Datastream.iot.navigationLink, value.FeatureOfInterest.iot.navigationLink, value.resultTime))
df12 = subset(observprop_pm10_data, select = -c(X.iot.id, X.iot.selfLink, name, definition, Datastreams.iot.navigationLink))

# pm25 kal data
df13 = subset(obser_pm25_kal_data, select = -c(X.iot.nextLink, value..iot.id, value..iot.selfLink, value.Datastream.iot.navigationLink, value.FeatureOfInterest.iot.navigationLink, value.resultTime))
```

```
df14 = subset(obser2_pm25_kal_data, select = -c(X.iot.nextLink, value..iot.id, value..iot.selfLink, value.
Datastream.iot.navigationLink, value.FeatureOfInterest.iot.navigationLink, value.resultTime))
df15 = subset(obser3_pm25_kal_data, select = -c(X.iot.nextLink, value..iot.id, value..iot.selfLink, value.
Datastream.iot.navigationLink, value.FeatureOfInterest.iot.navigationLink, value.resultTime))
df16 = subset(obser4_pm25_kal_data, select = -c(X.iot.nextLink, value..iot.id, value..iot.selfLink, value.
Datastream.iot.navigationLink, value.FeatureOfInterest.iot.navigationLink, value.resultTime))
df17 = subset(observprop_pm25_kal_data, select = -c(X.iot.id, X.iot.selfLink, name, definition, Datastre
ams.iot.navigationLink))
# pm10 kal data
df18 = subset(obser_pm10_kal_data, select = -c(X.iot.nextLink, value..iot.id, value..iot.selfLink, value.D
atastream.iot.navigationLink, value.FeatureOfInterest.iot.navigationLink, value.resultTime))
df19 = subset(obser2_pm10_kal_data, select = -c(X.iot.nextLink, value..iot.id, value..iot.selfLink, value.
Datastream.iot.navigationLink, value.FeatureOfInterest.iot.navigationLink, value.resultTime))
df20 = subset(obser3_pm10_kal_data, select = -c(X.iot.nextLink, value..iot.id, value..iot.selfLink, value.
Datastream.iot.navigationLink, value.FeatureOfInterest.iot.navigationLink, value.resultTime))
df21 = subset(obser4_pm10_kal_data, select = -c(X.iot.nextLink, value..iot.id, value..iot.selfLink, value.
Datastream.iot.navigationLink, value.FeatureOfInterest.iot.navigationLink, value.resultTime))
df22 = subset(observprop_pm10_kal_data, select = -c(X.iot.id, X.iot.selfLink, name, definition, Datastre
ams.iot.navigationLink))
```

Vervolgens moeten er in de observatie bestanden de kolomnamen van de waardes aangepast worden. Dit zal in een overzichtelijke dataframe resulteren wanneer alles is samengevoegd.

```
names(df3)[2] <- "value.pm25"
names(df4)[2] <- "value.pm25"
names(df5)[2] <- "value.pm25"
names(df6)[2] <- "value.pm25"
names(df7)[1] <- "description.pm25"

names(df8)[2] <- "value.pm10"
names(df9)[2] <- "value.pm10"
names(df10)[2] <- "value.pm10"
names(df11)[2] <- "value.pm10"
names(df12)[1] <- "description.pm10"

names(df13)[2] <- "value.pm25.kal"
names(df14)[2] <- "value.pm25.kal"
names(df15)[2] <- "value.pm25.kal"
names(df16)[2] <- "value.pm25.kal"
names(df17)[1] <- "description.pm25.kal"

names(df18)[2] <- "value.pm10.kal"
names(df19)[2] <- "value.pm10.kal"
names(df20)[2] <- "value.pm10.kal"
names(df21)[2] <- "value.pm10.kal"
names(df22)[1] <- "description.pm10.kal"
```

Om in de volgende stappen alle dataframes samen te voegen, wordt er in elke dataframe de kolom ‘naam’ toegevoegd. Hierdoor kunnen alle losse dataframes uiteindelijk door een overeenkomend kolom samengevoegd worden.

```
df2$name <- df$name

# pm25 data
df3$name <- df$name
df4$name <- df$name
df5$name <- df$name
df6$name <- df$name
df7$name <- df$name

# pm10 data
df8$name <- df$name
df9$name <- df$name
df10$name <- df$name
df11$name <- df$name
df12$name <- df$name

# pm25 kal data
df13$name <- df$name
df14$name <- df$name
df15$name <- df$name
df16$name <- df$name
df17$name <- df$name

# pm10 kal data
df18$name <- df$name
df19$name <- df$name
df20$name <- df$name
df21$name <- df$name
df22$name <- df$name
```

9. Samenvoegen van de tabellen

Er zijn twee mogelijkheden om datasets samen te voegen: horizontaal en verticaal. Horizontaal: toevoegen kolommen door de functie 'merge' te gebruiken: total <- merge(data frameA, data frameB,by"ID"). Verticaal: toevoegen rijen door de functie 'rbind' te gebruiken: total <- rbind(data frameA, data frameB).

```
total <- merge(df, df2,by="name")

# pm25 data
data_pm25 <- rbind(df3, df4)
data2_pm25 <- rbind(df5, df6)
totaldata_pm25 <- rbind.fill(data_pm25,data2_pm25)
total_pm25 <- merge(totaldata_pm25, df7, by="name")
table_pm25 <- merge(total, total_pm25,by="name")

# pm10 data
data_pm10 <- rbind(df8, df9)
data2_pm10 <- rbind(df10, df11)
totaldata_pm10 <- rbind.fill(data_pm10,data2_pm10)
total_pm10 <- merge(totaldata_pm10, df12,by="name")
table_pm10 <- merge(total, total_pm10,by="name")
```

```
# pm25 kal data
data_pm25_kal <- rbind(df13, df14)
data2_pm25_kal <- rbind(df15, df16)
totaldata_pm25_kal <- rbind.fill(data_pm25_kal,data2_pm25_kal)
table_pm25_kal <- merge(totaldata_pm25_kal, df17, by="name")
#table_pm25_kal <- merge(total, total_pm25_kal,by="name")

# pm10 kal data
data_pm10_kal <- rbind(df18, df19)
data2_pm10_kal <- rbind(df20, df21)
totaldata_pm10_kal <- rbind.fill(data_pm10_kal,data2_pm10_kal)
table_pm10_kal <- merge(totaldata_pm10_kal, df22,by="name")
#table_pm10_kal <- merge(total, total_pm10_kal,by="name")

# tabel met alle data
table_total <- merge(table_pm25, total_pm10,by="value.phenomenonTime")
table_total2 <- merge(table_pm25_kal, table_pm10_kal,by="value.phenomenonTime")
table_total3 <- merge(table_total, table_total2,by="value.phenomenonTime")
SamenMeten_final = subset(table_total3, select = -c(name.y.x, name.y.y, name.x.y))
```

10. Laatste bewerkingen kolommen / bijhorende waarden in de tabellen

Om in een latere fase de tabel te kunnen koppelen aan het Hollandse Luchten platform van het RIVM, is het noodzakelijk dat alle kolommen en rijen in de nieuwe dataset overeenkomen met die van de oude dataset. Dit betekent dat alle kolommen en de desbetreffende waarden in de kolommen van de nieuwe dataset, overeenkomen met die van de oude dataset.

Als eerste zijn noodzakelijke kolommen verwerkt en aangepast. Dit is het geval bij de locatiegegevens. de lat en lon, die beide in apparte kolommen geplaats moeten worden. Vervolgens is de waarde aanduiding in de kolom met de tijdgegevens geplaatst.

```
SamenMeten_final <- separate(SamenMeten_final, col=value.location.coordinates, into = c("lat","lon")
, sep = ",")
SamenMeten_final <- separate(SamenMeten_final, col=value.phenomenonTime, into = c("date","time
"), sep = "T")
View(SamenMeten_final)
```

Bij de onderstaande code worden onnodige tekens, in dit geval ‘(’, ‘)’ en ‘.000Z’, uit de kolommen verwijderd. Dit wordt gerealiseerd door een gedeelte binnen de kolomnaam te splitsen en verwijderen van de desbetreffende waardes.

```
SamenMeten_final$lat <- sapply(strsplit(SamenMeten_final$lat, split='(', fixed=TRUE), function(x) (x[2]
))
SamenMeten_final$lon <- sapply(strsplit(SamenMeten_final$lon, split=')', fixed=TRUE), function(x) (x[
1]))
SamenMeten_final$time <- sapply(strsplit(SamenMeten_final$time, split='.000Z', fixed=TRUE), functio
n(x) (x[1]))

SamenMeten_final$date <- as.POSIXct(paste(SamenMeten_final$date, SamenMeten_final$time), for
mat="%Y-%m-%d %H:%M:%S")
SamenMeten_final$time <- NULL
```

Vervolgens is het van belang om te kijken hoe de class van de kolom wordt gedefinieerd. Om de data uiteindelijk te kunnen verwerken in het Hollandse Luchten platform. De functie ‘class’ wordt gebruikt om erachter te komen wat voor soort class kolommen in de tabel hebben.

```
class(SamenMeten_final$date)

class(SamenMeten_final$pm25)

class(SamenMeten_final$pm10)

class(SamenMeten_final$lat)

class(SamenMeten_final$lon)
```

In dit geval zijn er twee kolommen in de nieuwe dataset waarbij de class niet overeenkomt met de oude dataset. Om de juiste class aan deze kolommen te hangen, wordt de functie ‘as.numeric’ gebruikt. Deze functie zorgt ervoor dat de kolommen de class ‘numeric’ krijgen.

```
SamenMeten_final$lat <- as.numeric(as.character(SamenMeten_final$lat))
SamenMeten_final$lon <- as.numeric(as.character(SamenMeten_final$lon))
```

Vervolgens is het de bedoeling dat alle kolomnamen overeenkomen met die van de oude dataset Met de functie colnames worden de kolomnamen verandert naar de kolomnamen die in de dataset van het Hollandse luchten platform wordt gebruikt.

```
colnames(SamenMeten_final)[2] <- "kit_id"
colnames(SamenMeten_final)[7] <- "pm25"
colnames(SamenMeten_final)[9] <- "pm10"
colnames(SamenMeten_final)[11] <- "pm25_kal"
colnames(SamenMeten_final)[13] <- "pm10_kal"
```

11. Opslaan van de dataset

Met behulp van de onderstaande code wordt de data opgeslagen als rds bestand (Speciaal bestandsformaat RStudio) Door het als rds bestand op te slaan, kan het r-script makkelijk in de global r-script van het Hollandse Luchten platform worden toegevoegd.

```
saveRDS(SamenMeten_final, file = "LTD_22481.rds")
```

4.3 Interactieve kaart

In dit script wordt de interactieve kaart vormgegeven. Voor deze interactieve kaart wordt de bewerkte data uit de Luftdaten API gebruikt. Het script bestaat uit een aantal belangrijke stappen die in dit document verder worden toegelicht.

Eindresultaat: Een interactieve kaart dat de actuele pm10 en pm2.5 waardes uit de Luftdaten API weergeeft.

1. Benodigde pakketten installeren

Voordat er binnen RStudio gewerkt kan worden met de API's is het van belang dat alle benodigde pakketen zijn geïnstalleerd (functie: 'install.packages') en geactiveerd (functie: 'library').

```
library("shiny")
library("leaflet")
library("htmltools")
library("htmlwidgets")
library("leaflet.extras")
library("shinyWidgets ")
```

2. Binnenhalen data

Bij de onderstaande code wordt de bewerkte data van de Luftdaten API binnengehaald.

```
API_Luftdaten <- readRDS("API_Luftdaten ")
```

User Interface

In dit kopje wordt de indeling van de pagina aangemaakt.

3. Aanmaken UI

Met behulp van de waarde (value) ui wordt er aangegeven dat de betreffende functies de User Interface van de webpagina vormen.

```
ui <- bootstrapPage(
```

5. Grootte webpagina

Met behulp van de onderstaande code wordt aangegeven wat de grootte van de webpagina is.

```
tags$style(type = "text/css", "html, body {width:100%;height:100%}"),
```

5. Verwijzing naar de interactieve kaart

In de onderste code wordt er verwezen naar de interactieve kaart die in het kopje server is aangemaakt. Hiernaast wordt er aangegeven dat de kaart de gehele webpagina moet invullen.

```
leafletOutput("MyMap", width = "100%", height = "100%"),
```

6. Aanmaken slider en verwijzing naar de betreffende data

In de onderste code wordt een slider aangemaakt en verwezen naar de data die hieraan gekoppeld is. Ook wordt de layout van de slider aangepast (kleur en transparantie). Met behulp van de slider heeft de gebruiker de mogelijkheid om de data in de interactieve kaart te filteren.

```
absolutePanel(bottom = 0, left = 200, fixed = TRUE, class = "panel panel-default",  width = 320, style
=          "opacity: 0.8;",  chooseSliderSkin("Flat", color = "#112446"),

                                    sliderInput("range", "Selecteer Meetwaardes", min(0), max(60),
                                                value = range(API_Luftdaten$P1), step = 1
                                    ))
```

Server

In dit kopje wordt de (input & output) data die gelinkt is aan de functies in de User Interface uitgewerkt.

7. Server aanmaken

Met behulp van de waarde (value) server wordt er aangegeven dat de betreffende functies die de data vormgeven de server van de webpagina vormen.

```
server <- function(input, output) {
```

8. Uitwerken datastromen slider

Met de onderstaande codes worden de twee datastromen (pm10/pm2.5) voor de slider uitgewerkt. Met de functie reactive wordt er aangegeven dat de data moet mee veranderen met de instellingen van de slider die de gebruiker hanteert. Bij de filter functie worden de sensoren binnen Nederland uit de dataset gefilterd.

```
sliderData1 <- reactive({
                                    API_Luftdaten[API_Luftdaten$P1 >= input$range[1] &
API_Luftdaten$P1 <=                 input$range[2] ,]   })

                                                sliderData2 <- reactive({

        API_Luftdaten[API_Luftdaten$P2 >= input$range[1] & API_Luftdaten$P2 <=
                input$range[2] ,]    })
```

9. Aanmaken kleurenschema

Met de onderstaande code wordt een kleurenschema aangemaakt.

```
bins <- c(0, 5, 10, 15, 20, 25, 30, 35, 40, Inf) color_gradient <- colorBin("YlOrRd", domain =
        API_Luftdaten$P1, bins = bins)
```

10. Aanmaken titel

In de onderstaande code wordt de opmaak van de titel vormgegeven.

```
tag.map.title <- tags$style(HTML("
 .leaflet-control.map-title {
   transform: translate(-50%,20%);
   position: fixed !important;
   left: 50%;
   text-align: center;
   padding-left: 15px;
   padding-right: 15px;
   background: rgba(255,255,255,0.75);
   font-weight: bold;
   font-size: 28px;
 } "   ))
```

In de onderstaande code krijgt de titel een naam.

```
title <- tags$div(
                tag.map.title, HTML("Realtime DataViewer (Luftdaten API)")  )
```

11. Vormgeving interactieve kaart

Met behulp van de onderstaande code wordt de kaart vormgegeven (renderLeaflet/leaflet). Deze kaart bevat geen (dynamische) data uit de API. Deze data wordt in de functie

ProxyLeafet binnengehaald en gelinkt aan deze kaart. Binnen deze code worden alle functionaliteiten die betrekking hebben tot de kaart uitgewerkt.

```
output$MyMap <- renderLeaflet({

 leaflet(API_Luftdaten) %>%
```

11.1 Toevoegen titel

In de onderstaande code wordt de titel aan de kaart toegevoegd.

```
addControl(title, position = "topleft", className="map-title") %>%
```

11.2 Toevoegen basemaps

Met de onderstaande code worden verschillende basemaps ingeladen.

```
addProviderTiles(providers$OpenStreetMap.Mapnik, group = "OpenStreetMap") %>%

addProviderTiles(providers$Esri.WorldImagery, group = "Satelliet") %>%

addProviderTiles(providers$Esri.WorldGrayCanvas, group = "Gray Canvas” %>%
```

Met de onderstaande code wordt de standard basemap ingesteld (OpenStreetMap)

```
addTiles(group = "OpenStreetMap") %>%
```

11.3 Aanmaken standaard kaartscherm

Met deze code wordt het standaard kaartscherm ingesteld op de gemeente Arnhem.

```
setView(5.906991, 51.981582, zoom = 12) %>%
```

11.4 Toevoegen selectievakje

Met de onderstaande codes wordt een selectievakje aangemaakt. Met behulp van dit selectievakje is het mogelijk om een meetwaarde (pm10 & pm25) en een basemap te selecteren.

```
  addLayersControl(

    baseGroups = c("OpenStreetMap", "Satelliet", "Gray Canvas"),

    overlayGroups = c("pm10", "pm2.5"),

    options = layersControlOptions(collapsed = FALSE)) %>% setView (5.906991, 51.981582, zoom =
12) %>%
```

Met de onderstaande code wordt het selectievakje pm2.5 standaard uitgezet.

```
  hideGroup("pm2.5") %>%
```

11.5 Toevoegen minimap

Met de onderstaande codes wordt een minimap aangemaakt.

```
  addMiniMap(toggleDisplay = TRUE) %>%
```

11.6 Toevoegen legenda

Met de onderstaande code wordt een legenda toegevoegd.

```
  addLegend("bottomleft", pal =  color_gradient, values = API_Luftdaten$P1,
  title = "Meetwaardes (µg/m3)") %>%
```

11.7 Toevoegen zoomfunctie Nederland

Met de onderstaande code wordt een knop toegevoegd (Javascript). Hiermee is het mogelijk om automatisch in/uit te zoomen naar Nederland.

```
addEasyButton(easyButton(
    icon= "fa-globe", title= "Zoom naar Nederland",
    onClick=JS("function(btn, map){ map.setZoom(7); }"))) %>%
```

11.8 Toevoegen zoomfunctie huidige locatie

Met de onderstaande code wordt een knop toegevoegd (Javascript). Hiermee is het mogelijk om in te zoomen naar de huidge locatie van de gebruiker.

```
addEasyButton(easyButton(
        icon= "fa-crosshairs" , title= "Mijn Locatie",
        onClick=JS("function(btn, map){ map.locate({setView: true, enableHighAccuracy: true}); }" )))
%>%
```

11.9 Toevoegen schaalbalk

Met de onderstaande code wordt een knop toegevoegd (Javascript). Hiermee is het mogelijk om in te zoomen naar de huidge locatie van de gebruiker.

```
addScaleBar(position = "bottomleft", options = scaleBarOptions(metric = TRUE,
  imperial = FALSE, maxWidth = 150))  %>%
```

11.10 Reset kaartscherm

Met de onderstaande code wordt een reset knop toegevoegd. Hiermee wordt het mogelijk te maken om het kaartscherm te resetten naar het startscherm (gemeente Arnhem).

```
addResetMapButton() %>%
```

11.11 Toevoegen zoekfunctie

Met de onderstaande code wordt een zoekfunctie toegevoegd. Hiermee is het mogelijk om naar adressen te zoeken.

```
addSearchOSM(options = searchOptions(moveToLocation = TRUE,
        collapsed = TRUE, autoCollapse = FALSE, textCancel = "Verwijderen",
```

```
        zoom = 15, autoType = TRUE,  autoResize = TRUE,

        hideMarkerOnCollapse = TRUE,  textErr = "Locatie niet gevonden",

        textPlaceholder = "Zoeken..."))                })
```

12. Vormgeving dynamische data

De data in de kaart moet mee veranderen met de instellingen van de slider. Hierdoor wordt er gebruikt gemaakt van de functie leafletProxy. Deze functie maakt het mogelijk om (dynamische) data uit de Luftdaten API vorm te geven en te linken aan de leaflet kaart (en functies) die hierboven is gemaakt. Binnen de leaflet functie is het niet mogelijk om met dynamische data te werken. De Observe functie geeft aan dat de data moet kunnen reageren op veranderingen (slider).

```
observe({

  leafletProxy( "MyMap", data = sliderData1()) %>%

  clearMarkers() %>%
```

12.1 Toevoegen zoekfunctie

Met de onderstaande codes worden cirkels in de kaart getekend voor de waarde P1 (pm10). Deze waardes worden gelinkt aan de slider, zodat de cirkels mee kunnen veranderen met de betreffende instellingen. Voor het kleurgebruik wordt er verwezen naar het gemaakte kleurenschema (color_gradient). Tot slot worden aan de cirkels labels gekoppeld en wordt de data gelinkt aan het selectievakje.

```
addCircleMarkers(data = sliderData1(), ~lon, ~lat, layerId = ~sensor.id,

                 label = lapply(sliderData1()$P1, HTML), radius = 8,

                 fillColor = color_gradient(sliderData1()$P1), fillOpacity = 1,

                 group = "pm10", stroke = TRUE, color = "black", weight = 2,
```

Met de onderstaande code wordt het pop up scherm vormgegeven.

```
popup= ~paste("Sensor ID:", sliderData1()$sensor.id, "<br>",

                      "Meetwaarde:", "pm10", "<br>",

                      "Sensorwaarde:", sliderData1()$P1, "<br>",

                      "Coordinaten:", sliderData1()$lat, ",  ",   sliderData1()$lon)) %>%
```

12.2 Toevoegen zoekfunctie

Met de onderstaande codes worden cirkels in de kaart getekend voor de waarde P2 (pm2.5). Deze waardes worden gelinkt aan de slider, zodat de cirkels mee kunnen veranderen met de betreffende instellingen. Voor het kleurgebruik wordt er verwezen naar het gemaakte kleurenschema (color_gradient). Tot slot worden aan de cirkels labels gekoppeld en wordt de data gelinkt aan het selectievakje.

```
addCircleMarkers(data = sliderData2(), ~lon, ~lat, layerId = ~kit_id,
                 label = lapply(sliderData2()$P2, HTML), radius = 8,
                 fillColor = color_gradient(sliderData2()$P2), fillOpacity = 1,
                 group = "pm2.5", stroke = TRUE, color = "black", weight = 2,
```

Met de onderstaande code wordt het pop up scherm vormgegeven.

```
popup= ~paste("Sensor ID:", sliderData2()$sensor.id, "<br>",
                        "Meetwaarde:", "pm2.5", "<br>",
                        "Sensorwaarde:", sliderData2()$P2, "<br>",
                        "Coordinaten:", sliderData2()$lat, ",  ",   sliderData2()$lon)) %>%
```

13. Applicatie runnen

Door de onderstaande code kan de applicatie gerund worden.

```
shinyApp(ui = ui, server = server)
```

4.4 Downloadfunctie

In dit script wordt de downloadfunctie opgesteld. Voor deze downloadfunctie kaart wordt de bewerkte data uit de Samen Meten API gebruikt. Het script bestaat uit een aantal belangrijke stappen die in dit document verder worden toegelicht.

Eindresultaat: Een downloadfunctie, waarbij de gebruiker historische meetwaardes (pm10 en pm2.5) uit de Samen Meten API kan downloaden.

1. Benodigde pakketten installeren

Voordat er binnen RStudio gewerkt kan worden met de API's is het van belang dat alle benodigde pakketen zijn geïnstalleerd (functie: 'install.packages') en geactiveerd (functie: 'library').

```
library("shiny")
```

User Interface

In dit kopje wordt de indeling van de downloadfunctie aangemaakt.

2. Aanmaken UI

Met behulp van de waarde (value) tp wordt er aangegeven dat de betreffende functies de User Interface van de downloadfunctie vormen. Zo worden onder andere tekstvensters, selectievensters en verschillende (interactieve) knoppen vormgegeven.

```
tp <-  tabPanel( "Downloaden",
              sidebarLayout(
               sidebarPanel(id = "sidebar",
                        h4("Toelichting"),
                        p("De sensoren in de toolbox kunnen gedownload worden.
                          In het drop-down menu kan een specifieke sensor-ID worden geselecteerd.
                          Vervolgens kan er een download formaattype geselecteerd worden.
                          Wanneer beide elementen zijn ingevuld en je op de download knop klikt,
                          wordt het desbetreffende bestand lokaal opgeslagen op je apparaat.",
                          style = "font-size:12px"),
                        width = 3),

               mainPanel(
                selectInput("dataset", "Selecteer de sensor",
                         choices = c("LTD_22481", "LTD_24283", "LTD_24322", "LTD_24801",
"LTD_25494",                                          "LTD_27239", "LTD_27720", "LTD_31298")),
                br(),
                helpText("Selecteer het downloadformaat"),
                radioButtons ("type", "Formaat type:",
                         choices = c("Excel (CSV)", "Text (TSV)", "Text (Space Separated)", "Doc")),
                br(),
                helpText("Klik op de download knop om de geselecteerde sensor te downloaden"),
                downloadButton('downloadData', 'Download'),
                width = 9),
              position = "right"))

 return(tp)
}
```

Server

In dit kopje wordt de (input & output) data die gelinkt is aan de functies in de User Interface uitgewerkt.

3. Aanmaken Server

Met behulp van de waarde (value) server wordt er aangegeven dat de betreffende functies die de data vormgeven de server van de downloadfunctie vormen.

```
server <- function(input, output) {
```

4. Downloadfunctie creëren

Met behulp van de onderstaande code wordt de downloadfunctie voor de sensoren van de Samen Meten API uitgewerkt.

```
 datasetInput <- reactive({
```

4.1 Dataset selecteren

Met behulp van de onderstaande code wordt het mogelijk gemaakt dat de gebruiker de dataset van een sensor kan selecteren. Dit wordt gedaan op basis van de gewenste sensor id.

```
switch(input$dataset,
       "LTD_22481" = input_df1,
       "LTD_24283" = input_df2,
       "LTD_24322" = input_df3,
       "LTD_24801" = input_df4,
       "LTD_25494" = input_df5,
       "LTD_27239" = input_df6,
       "LTD_27720" = input_df7,
       "LTD_31298" = input_df8)
 })                    }
```

4.2 Bestandsformaat selecteren

Met behulp van de onderstaande code heeft de gebruiker de mogelijkheid om het gewenste bestandsformaat te selecteren.

```
fileext <- reactive({  switch(input$type,
       "Excel (CSV)" = "csv", "Text (TSV)" = "txt", "Text (Space Seperated)" = "txt", "Doc" = "doc")
 })
```

4.3 Dataset koppelen aan bestandsformaat

Met behulp van de onderstaande code wordt de geselecteerde dataset (sensor.id) gekoppeld aan het geselecteerde bestandsformaat.

```
output$downloadData <- downloadHandler(
  filename = function(){
    paste(input$dataset, fileext(), sep= ".")
  },

  content = function(file) {
    sep <- switch(input$type, "Excel (CSV)" = ",", "Text (TSV)" = "\t", "Text (Space Separated)"
          = " ", " Doc" = " ")

    write.table(datasetInput(), file, sep = sep,
              row.names = FALSE)                                    } )
```

4.5 Interactieve grafieken

In dit script worden twee interactieve grafieken opgesteld. Voor deze interactieve grafieken wordt de bewerkte data uit de Samen Meten API gebruikt. Het script bestaat uit een aantal belangrijke stappen die in dit document verder worden toegelicht.

Eindresultaat: Twee interactieve grafieken, waar de historische meetwaardes (pm10 en pm2.5) uit de Samen Meten API in gevisualiseerd worden.

1. Benodigde pakketten installeren

Voordat er binnen RStudio gewerkt kan worden met de API's is het van belang dat alle benodigde pakketen zijn geïnstalleerd (functie: 'install.packages') en geactiveerd (functie: 'library').

```
library("shiny")
library("plotly")
library("ggplot")
```

User Interface

In dit kopje wordt de indeling voor de interactieve grafieken aangemaakt.

2. Aanmaken UI

Met behulp van de waarde (value) tp wordt er aangegeven dat de betreffende functies de User Interface van de interactieve grafieken vormen. Zo worden tekstvenster opgesteld en wordt de opmaak samengesteld.

3. Verwijzing naar de interactieve grafiek voor pm10 waardes

In de onderste code wordt er verwezen naar de interactieve grafiek (pm10) die in het kopje server is aangemaakt.

```
tp <-  tabPanel( "Grafiek PM 10",
              sidebarLayout(
               sidebarPanel(id = "sidebar",
                        h4( "Toelichting"),
                        p( "Bij selectie van een sensor wordt de PM 10 waarde over een bepaald
                       tijdsbestek getoond. Hierbij is selectie van meerdere sensoren mogelijk.
                      De grafiek bevat een aantal functionaliteiten (legenda en extra functies) om
                      de data beter te bekijken, vergelijken en analyseren. Er is ook de mogelijkheid
                      om de grafiek te downloaden (PNG formaat).",
                           style = "font-size:12px"),
                           width = 3),
               mainPanel(
                helpText( "Laat een grafiek van de aangeklikte sensoren zien (PM 10)"),
                plotlyOutput( "interplot"),
                width = 9),
               position = "right"))
 return(tp)
}
```

4. Verwijzing naar de interactieve grafiek voor pm2.5 waardes

In de onderste code wordt er verwezen naar de interactieve grafiek (pm2.5) die in het kopje server is aangemaakt.

```
tp <-  tabPanel( "Grafiek PM 2.5",
              sidebarLayout(
               sidebarPanel(id = "sidebar",
                        h4( "Toelichting"),
                        p( "Bij selectie van een sensor wordt de PM 2.5 waarde over een bepaald
                       tijdsbestek getoond. Hierbij is selectie van meerdere sensoren mogelijk.
                      De grafiek bevat een aantal functionaliteiten (legenda en extra functies) om
                      de data beter te bekijken, vergelijken en analyseren. Er is ook de mogelijkheid
                      om de grafiek te downloaden (PNG formaat).",
                           style = "font-size:12px"),
                           width = 3),
               mainPanel(
                helpText( "Laat een grafiek van de aangeklikte sensoren zien (PM 2.5)"),
                plotlyOutput( "interplot"),
                width = 9),
               position = "right"))
 return(tp)       }
```

Server

In dit kopje wordt de (input & output) data die gelinkt is aan de functies in de User Interface uitgewerkt.

5. Aanmaken Server

Met behulp van de waarde (value) server wordt er aangegeven dat de betreffende functies die de data vormgeven de server van de interactieve grafieken vormen.

```
server <- function(input, output) {
```

6. Interactieve grafiek pm10 creëren

Met behulp van de onderstaande code wordt de interactieve grafiek voor pm10 waardes van de Samen Meten API uitgewerkt.

```
output$interplot <- renderPlotly({
  comp <- selectReactiveComponent(input)
  dates <- selectReactiveDates(input)
  selected_id <- values$df[which(values$df$selected & values$df$groep == geen_groep),'kit_id']
  show_input <-input_df[which(input_df$kit_id %in% selected_id),]

  q <- ggplot(show_input, aes(x=date, y=pm10, color=kit_id))
  q <- q + geom_line()
  q <- q + geom_point(size=1)
  q <- q + scale_colour_hue(name= "Legenda", l=20)
  q <- q + xlab("Datum") + ylab("pm 10")
  q <- q + ggtitle("PM 10 waarde in een bepaald tijdsbestek")
  q <- q + theme_bw()  })
```

7. Interactieve grafiek pm2.5 creëren

Met behulp van de onderstaande code wordt de interactieve grafiek voor pm2.5 waardes van de Samen Meten API uitgewerkt.

```
output$interplot <- renderPlotly({
  comp <- selectReactiveComponent(input)
  dates <- selectReactiveDates(input)
  selected_id <- values$df[which(values$df$selected & values$df$groep == geen_groep),'kit_id']
  show_input <-input_df[which(input_df$kit_id %in% selected_id),]

  q <- ggplot(show_input, aes(x=date, y=pm2.5, color=kit_id))
  q <- q + geom_line()
  q <- q + geom_point(size=1)
  q <- q + scale_colour_hue(name= "Legenda", l=20)
  q <- q + xlab("Datum") + ylab("pm 2.5")
  q <- q + ggtitle("PM 2.5 waarde in een bepaald tijdsbestek")
  q <- q + theme_bw()  })
```

4.6 Tabellen (Luftdaten & Samen Meten)

In dit script worden twee tabellen opgesteld. Voor deze tabellen wordt de bewerkte data uit de Samen Meten- en Luftdaten API gebruikt. Het script bestaat uit een aantal belangrijke stappen die in dit document verder worden toegelicht.

Eindresultaat: Twee tabellen, waar de bewerkte data uit de Samen Meten- en Luftdaten API in worden weergegeven.

1. Benodigde pakketten installeren

Voordat er binnen RStudio gewerkt kan worden met de API's is het van belang dat alle benodigde pakketen zijn geïnstalleerd (functie: 'install.packages') en geactiveerd (functie: 'library').

```
library("shiny")
library("DT")
```

User Interface

In dit kopje wordt de indeling voor de twee tabellen aangemaakt.

2. Aanmaken UI

Met behulp van de onderstaande code worden de functionaliteiten die de User Interface vormen uitgewerkt.

3. Verwijzing naar de tabel van Samen Meten API

In de onderste code wordt er verwezen naar de tabel (Samen Meten API) die in het kopje server is aangemaakt.

```
tabPanel( "Tabel (API Samen Meten)",
        fluidRow(
          column(10,
              DTOutput( 'table_overviewSam')
          ))),
```

4. Verwijzing naar de tabel van Luftdaten API

In de onderste code wordt er verwezen naar de tabel (Luftdaten API) die in het kopje server is aangemaakt.

```
tabPanel( "Tabel (API Luftdaten)",
        fluidRow(
          column(10,
              DTOutput( 'table_overviewLuft')
))),
```

Server

In dit kopje wordt de (input & output) data die gelinkt is aan de functies in de User Interface uitgewerkt.

5. Aanmaken Server

Met behulp van de waarde (value) server wordt er aangegeven dat de betreffende functies die de data vormgeven de server van de tabellen vormen.

```
server <- function(input, output) {
```

6. Tabel Samen Meten API creëren

Met behulp van de onderstaande code wordt een overzichtelijke tabel ontwikkeld, waar de data van de Samen Meten API in staat weergegeven.

```
output$table_overviewSam <- renderDT(input_df,
                    filter = "top",
                    options = list(
                      pageLength = 10
                    ) )
```

7. Tabel Luftdaten API creëren

Met behulp van de onderstaande code wordt een overzichtelijke tabel ontwikkeld, waar de data van de Luftdaten API in staat weergegeven.

```
output$table_overviewLuft <- renderDT(Luftdaten_API,
                    filter = "top",
                    options = list(
                      pageLength = 10
                    ) )
```

Bijlage 5: GIT functie (versies)

Nadat de API's succesvol zijn ingeladen en bewerkt is er succesvol een link gelegd tussen de data uit de API en de codes van het Hollandse Luchten platform (RIVM). Vanaf dit moment wordt er gekeken om het huidige platform uit te breiden met extra functionaliteiten. Voordat hiermee gestart kan worden is het belangrijk om tussentijdse versies op te kunnen slaan. Hiermee wordt er voor gezorgd dat je kan terugvallen op oudere versies, wanneer er in een later stadium fouten worden gemaakt. Op deze voorkom je dat je handmatig de code moet veranderen naar de code uit vorige versies. Het handmatig terugzetten van codes is een complexe werkwijze en neemt in tijd beslag.

Om te kunnen werken met verschillende versies wordt de tool GIT gedownload en gelinkt aan RStudio. GIT is een tool, waarmee je terug kunt bladeren in verschillende versies van je codes. Hiernaast kunnen deze versies (codes) ook gelinkt worden aan je GitHub account. Hieronder wordt een korte toelichting gegeven hoe deze tool in elkaar zit en hoe je dit linkt aan RStudio.

Installatie GIT en connectie met RStudio

Link installatie GIT: https://git-scm.com/downloads

In de bovenstaande link is de website te vinden, waarmee de GIT tool te installeren is. Wanneer de installatie is voltooid kan er worden gestart met de connectie tussen RStudio en de GIT tool. Wanneer RStudio is opgestart ga je naar Global Options (Tools menu) en vervolgens naar Git/SVN (zie figuur 1). Zorg ervoor dat de checkbox is aangevinkt en dat je linkt naar de folder, waar je de GIT tool hebt opgeslagen (GIT/bin/git). Wanneer deze instellingen goed staan sla de wijzigingen op en start RStudio opnieuw op.

Figuur 1: het maken van een connectie tussen RStudio en de tool GIT

Connectie RStudio met GIT account

Nu er succesvol een link is gelegd tussen RStudio en de tool GIT is het belangrijk om RStudio te linken aan je GIT account. Wanneer je nog geen GIT account hebt kan je dit via de onderstaande link aanmaken:

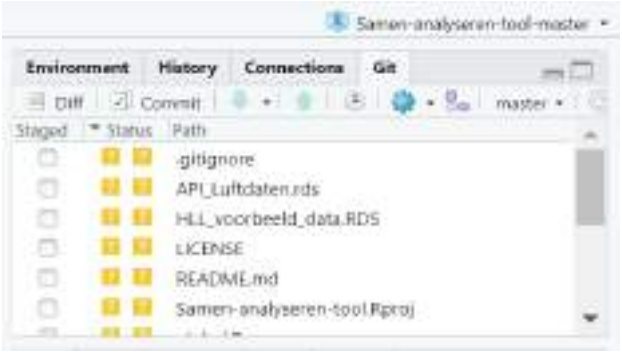
Link aanmaken GIT account:
https://github.com/join?ref_cta=Sign+up&ref_loc=header+logged+out&ref_page=%2F&source=header-home

Wanneer je een account heb aangemaakt ga je naar Shell (Tools menu) en plak je de onderstaande code met jouw gebruikersnaam en je e-mailadres in het script.

```
git config --global user.name 'yourGitHubUsername'
git config --global user.email 'name@provider.com'
```

RStudio project

Nu je succesvol RStudio hebt gelinkt aan je GIT account kan je aan de slag met de tool. Hiervoor dien je een nieuw RStudio project op te starten of een bestaand RStudio project openen. Als het goed is krijg je het scherm te zien dat in figuur 2 staat weergegeven.

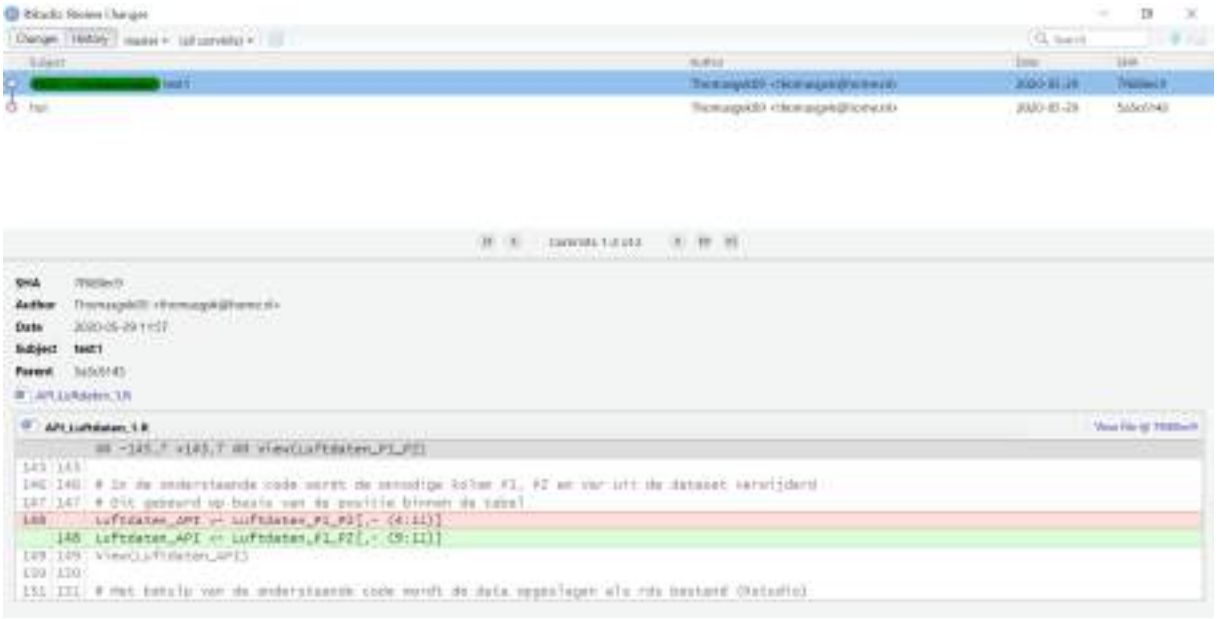


Figuur 2: de GIT tool binnen RStudio

Wanneer het scherm uit figuur 2 niet in beeld komt ga je naar Version Control (Tools menu) en dan naar Project Setup. Klik vervolgens op Git/SVN en zorg ervoor dat de optie Version control system op GIT staat. Start RStudio vervolgens opnieuw op en herhaal deze stap.

De GIT tool

Nu de GIT tool succesvol is ingeladen binnen een RStudio project kan je aan de slag met de GIT tool. Binnen de tool kies je de bestanden, waarbinnen je wilt gaan werken. Hierna ga je naar de optie 'Commit' om een versie van je codes op te slaan. Geef de versie een naam en klik nogmaals op Commit. Lees het bijbehorende pop-up scherm goed door om te kijken of het opslaan goed is verlopen. Verander een stuk code in je script en doe nog een Commit verzoek. Ga vervolgens naar de optie 'Diff' en er opent zich een nieuw scherm. Zodra je op 'History' klikt komt het onderstaande scherm in beeld. Dit scherm geeft de verschillende versies (commits) weer die zijn aangemaakt. Hiernaast kan je de betreffende code met de onderlinge verschillen tussen de versies terugvinden.



Figuur 3 het scherm 'History' waar de verschillende versies (commits) in staan weergegeven

Zodra RStudio niet goed is gelinkt met je GIT account krijg je geen commits te zien in het bovenstaande scherm. Als dit het geval is, probeer dan de stap 'connectie RStudio met GIT account' nogmaals uit te voeren en herhaal de stappen die erna volgen.

Oproepen voorgaande versies
Nu we succesvol verschillende versies van de codes hebben gemaakt is het van belang om deze versies op te roepen. Om dit te doen dien je een nieuwe Terminal aan te maken of een bestaande Terminal te openen. Dit kan je doen door daar Terminal te gaan (Tools menu) en daar een bestaande of een nieuwe terminal aan te maken. Zodra je dit hebt gemaakt krijg je onder in beeld een veld te zien waarin je de onderstaande code moet kopiëren.

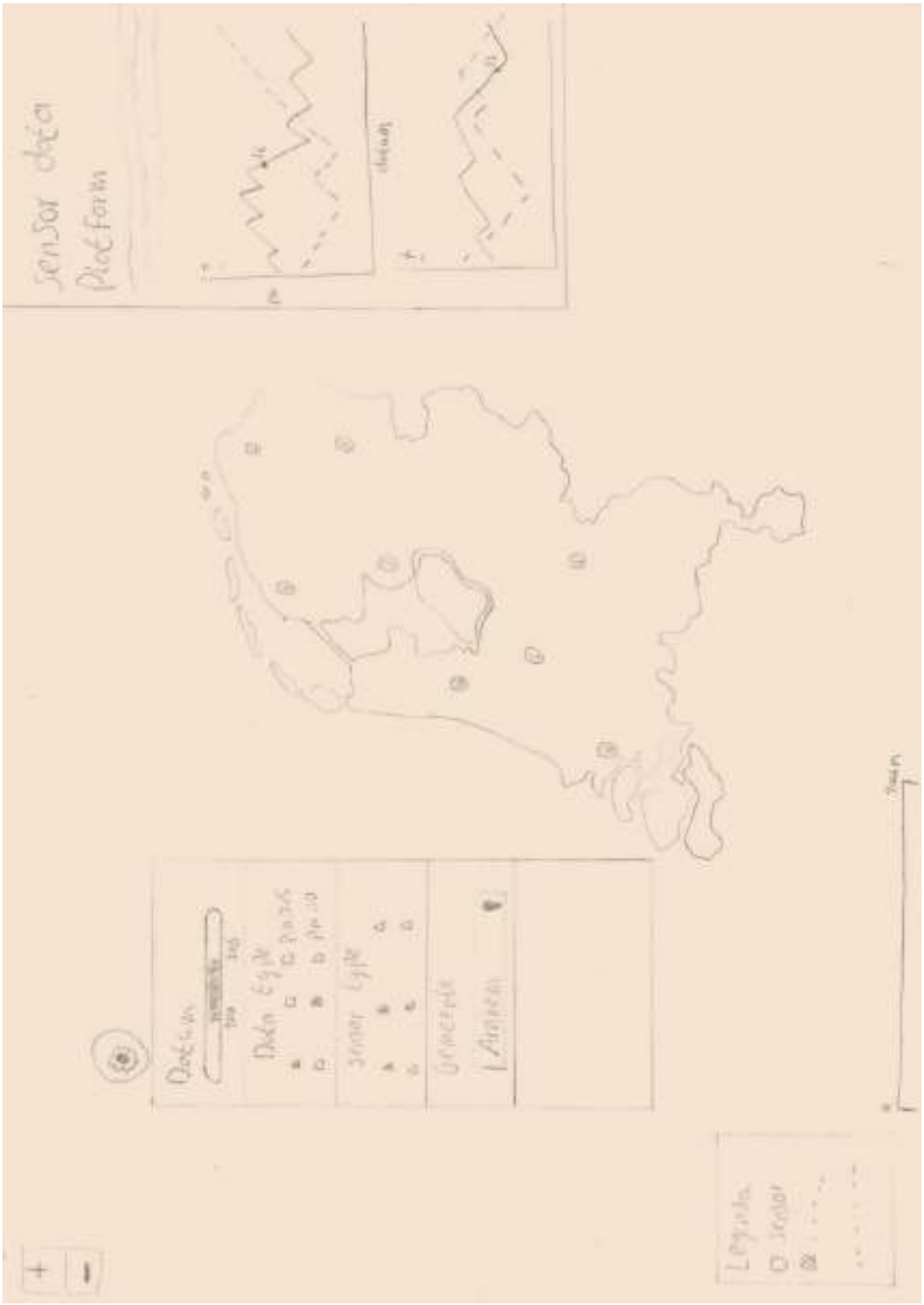
```
git reset --hard SHA
```

Bij het kopje SHA dient het SHA nummer ingevuld te worden. Dit is een identiek nummer dat elke versie (Commit) toegereikt krijgt. Dit nummer is voor elke versie te vinden in het scherm 'history'. Zodra je het SHA nummer hebt gevonden plak het in de bovenstaande code. Klik vervolgens op enter en wanneer je geen foutmeldingen krijgt moet de code in het script aangepast worden naar de code uit de betreffende versie.

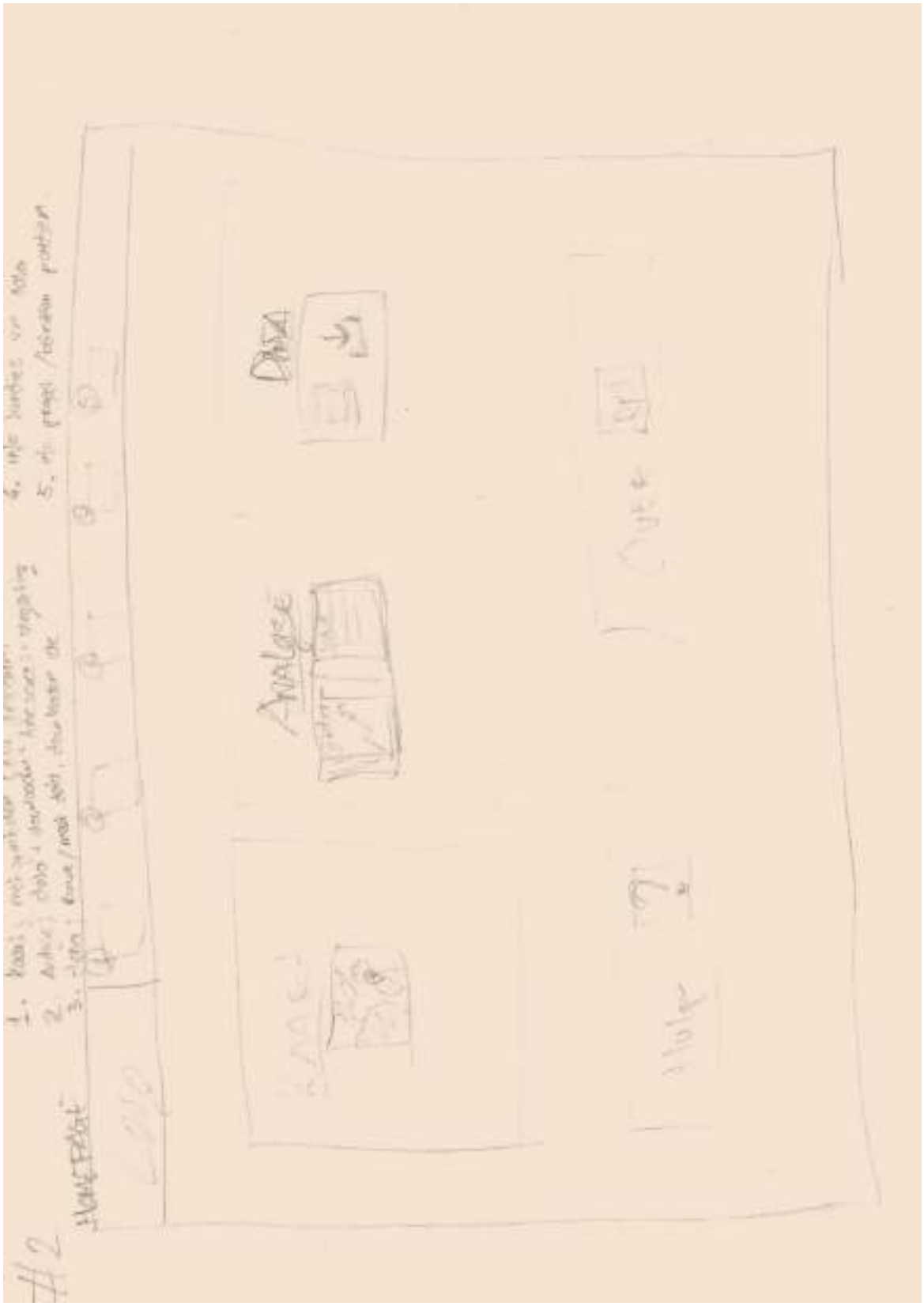
Nu het is gelukt om met behulp van de GIT tool verschillende versies van je code op te slaan en op te roepen kan er worden gestart met het uitbreiden van het Hollandse Luchten platform.

Bijlage 6: Schetsen (mock ups)

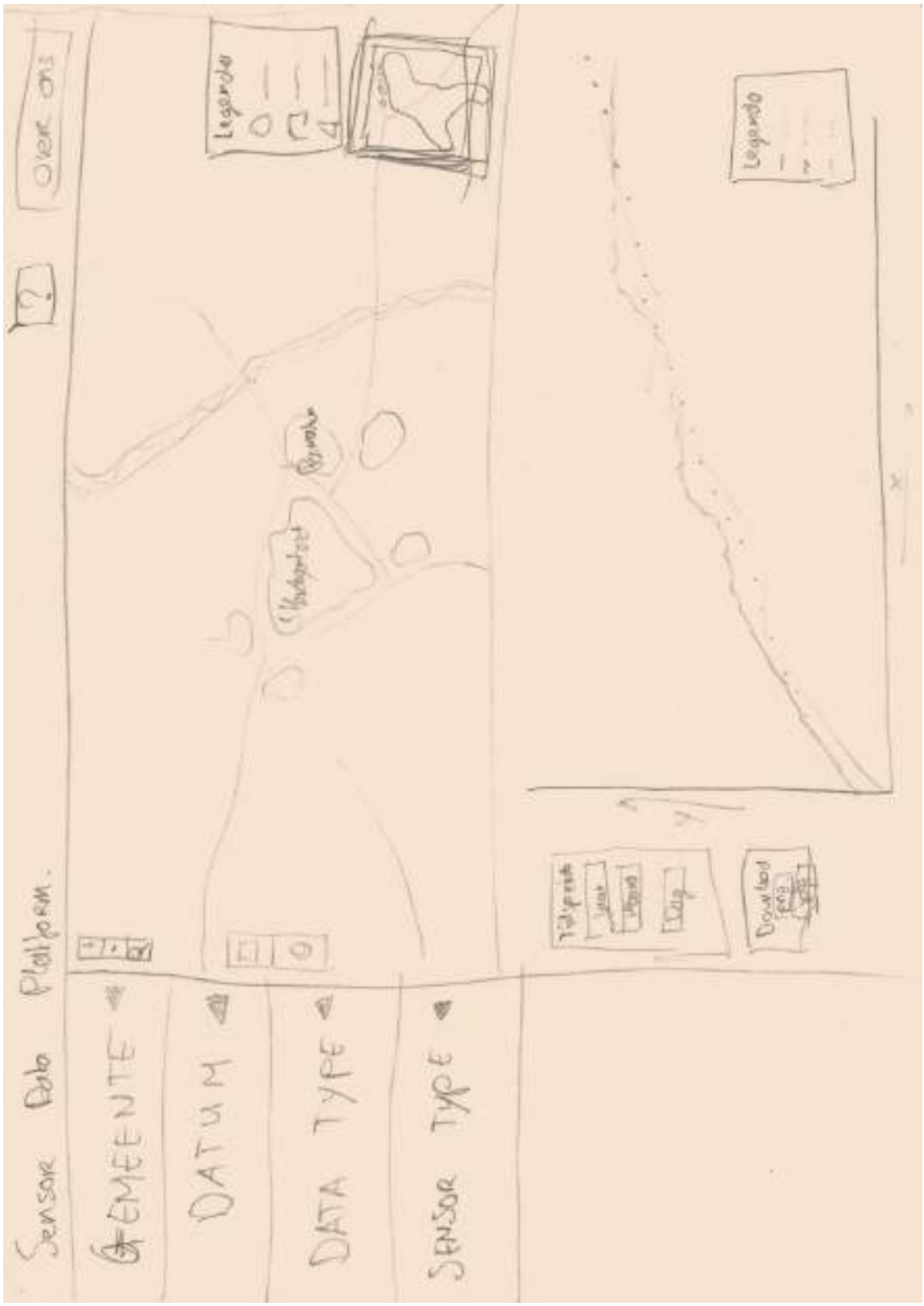
Ontwerpkeuze 1



Ontwerpkeuze 2

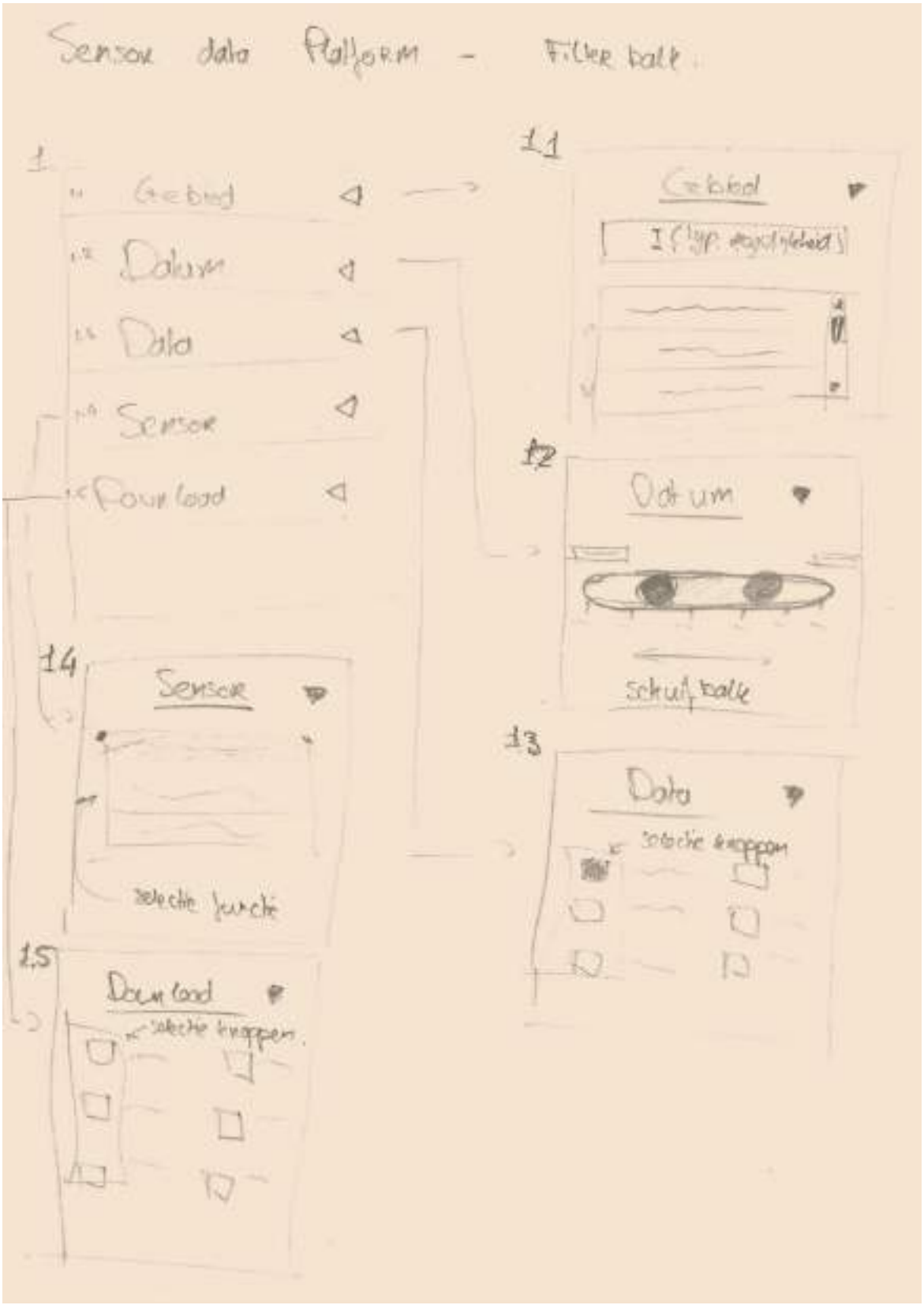


Gekozen ontwerp (1/2)

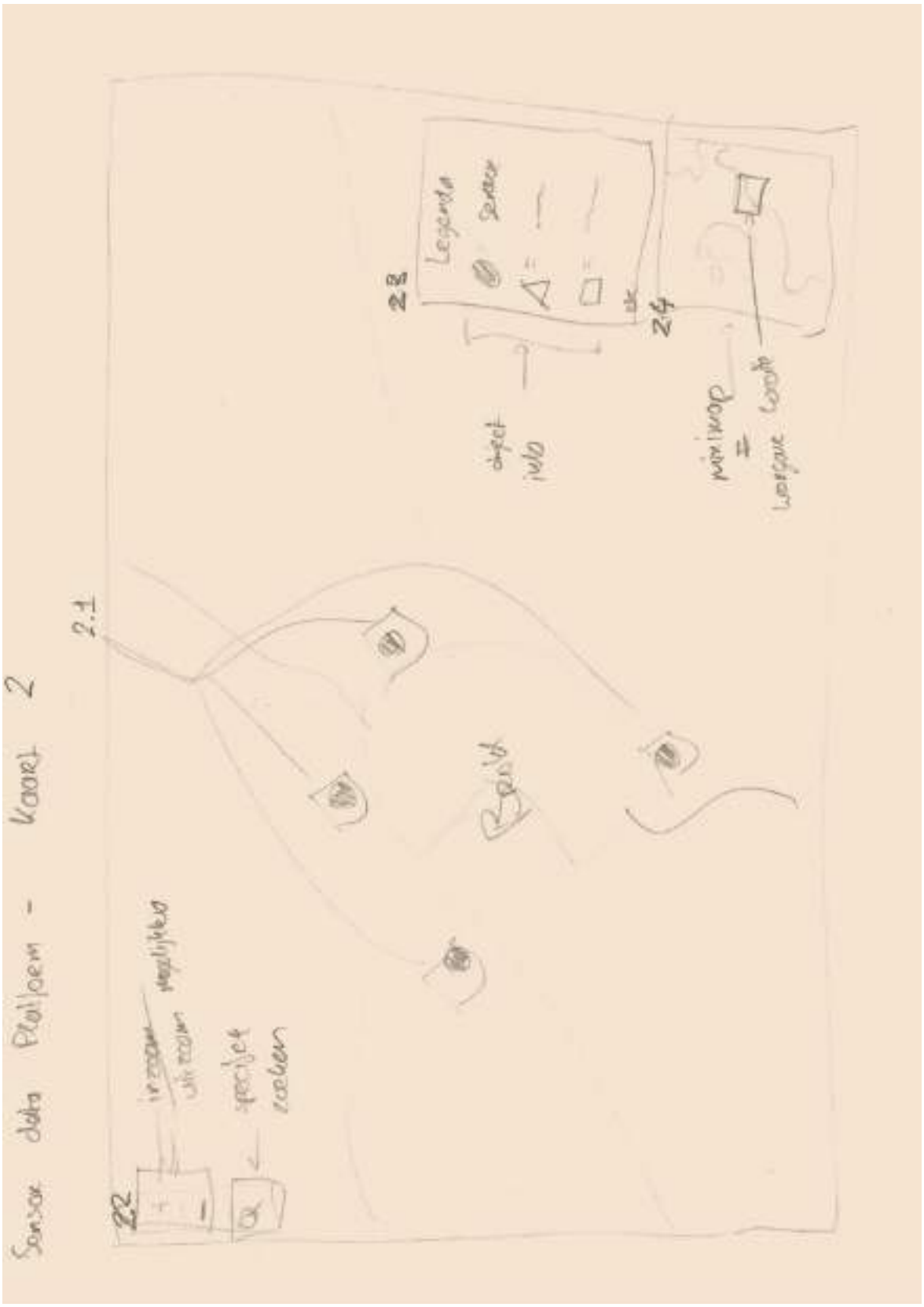


Gekozen ontwerp (2/2)

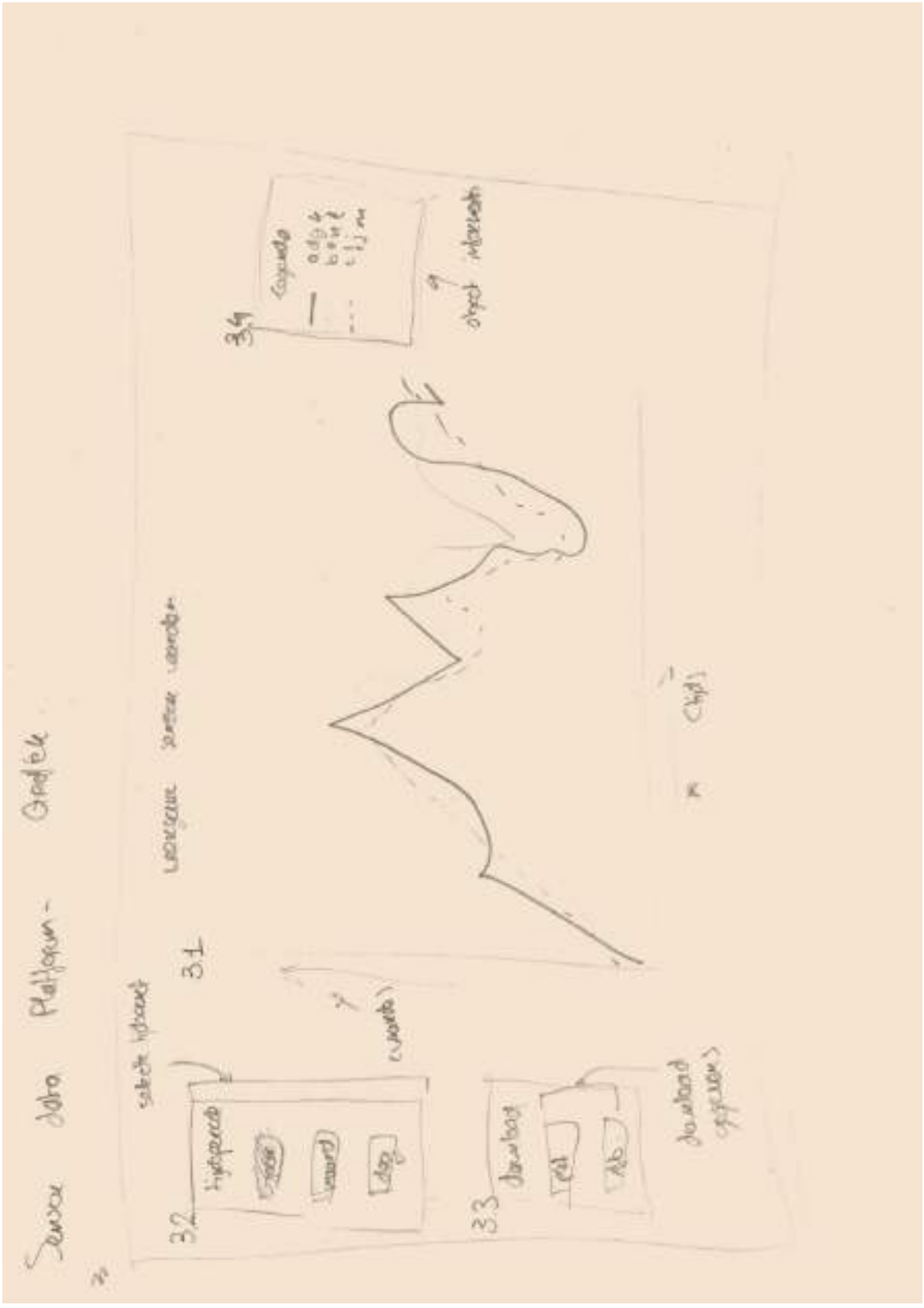
Filter functie



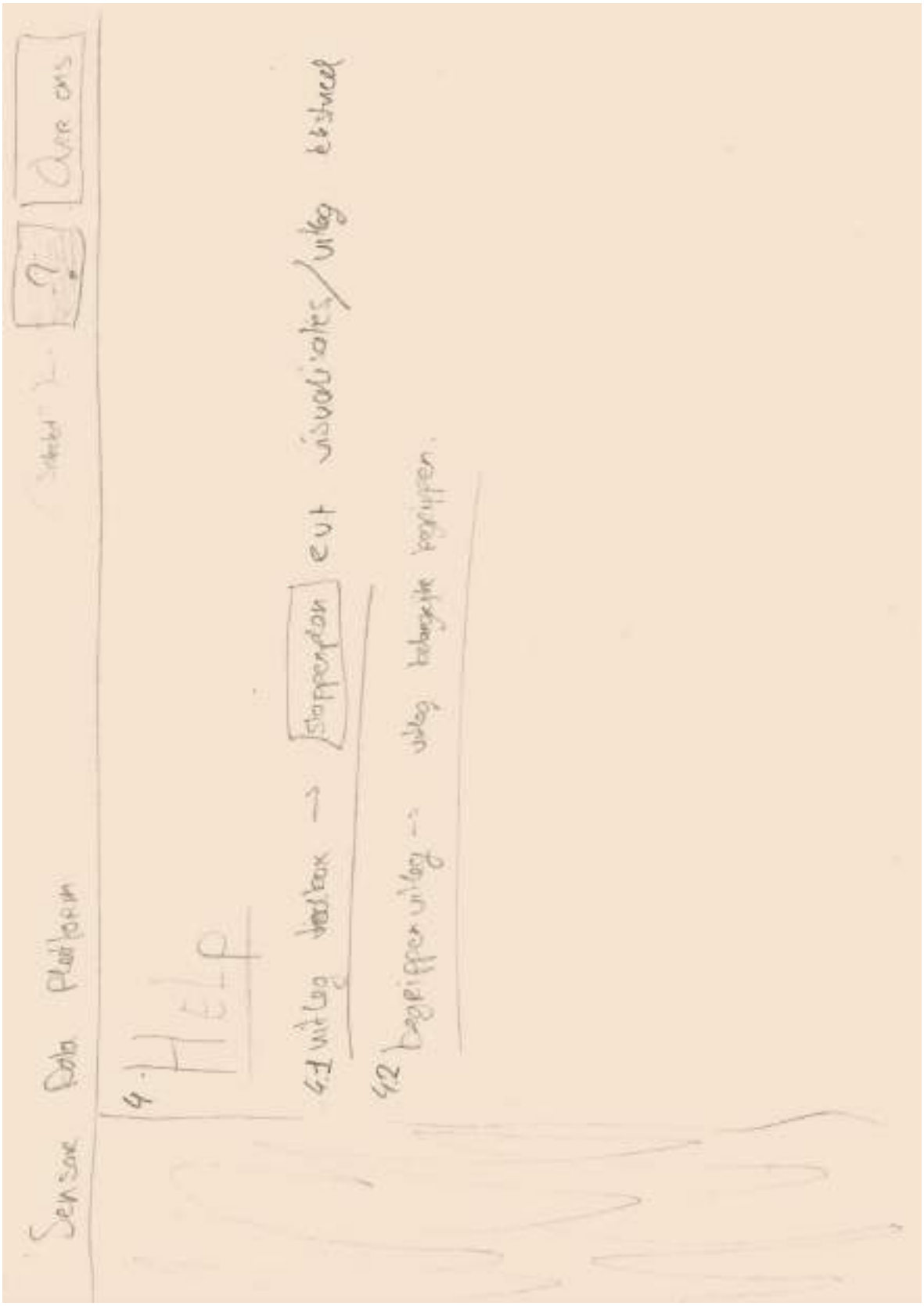
Kaart functie



Grafiek functie



Help pagina


Sensors
Data
Platform
?
Over ons
4. HELP
4.1 uitleg
4.2 Veelgestelde vragen

Over ons pagina