Navigatiesysteem voor elektrische scooters

Gerwin Hoogsteen
Jan Oene Krist

Navigatiesysteem voor elektrische scooters

Gerwin Hoogsteen
Jan Oene Krist

Eerste docent: Mw. S. Singh

Opdrachtgever: Dhr. J. Leupen
Coach: Dhr. H. Doornbos

Voorwoord

Voor u ligt het eindverslag van het afstudeerproject van Gerwin Hoogsteen en Jan Oene Krist. Het afgelopen jaar hebben we met veel plezier aan ons afstudeerproject gewerkt. We zijn begonnen met drie studenten. Later, vanaf februari, met twee. De opdracht die we uitgevoerd hebben is afkomstig van het Energie kenniscentrum.

Voor ons is er met dit verslag is er een einde gekomen aan onze studie aan de Hanzehogeschool. We hebben vier jaar hard gewerkt en veel plezier gehad. We kijken terug op vier fantastische jaren en we kunnen wel zeggen dat we het niet hadden willen missen. Ook het afstuderen hebben we heel leuk gevonden. De vrijheid die we hadden om onze ideeën te realiseren was erg prettig.

Graag willen we een aantal mensen bedanken die meegewerkt hebben aan onze opdracht. Als eerste Benny Mintjes. Hij heeft van september tot februari fantastisch werk geleverd en het is verschrikkelijk jammer dat hij nog niet mocht afstuderen. Ook Jan Leupen willen we graag bedanken voor de begeleiding. We hebben de samenwerking als erg prettig ervaren. Ook Jeroen van den Berg en Wim van Gemert willen we bedanken voor de support vanuit het EKC. Tot slot willen we ook graag Henk Doornbos bedanken voor de inhoudelijke begeleiding.

Groningen, juni 2010

Inhoudsopgave

Samenvatting

Tijdens ons afstuderen hebben we ons bezig gehouden met de ontwikkeling van een navigatiesysteem voor elektrische scooters. We willen graag het gebruik van duurzame mobiliteit stimuleren door te laten zien dat het gebruik ervan best hip is. Het navigatiesysteem dat wij gemaakt hebben heeft een aantal functies die een normaal navigatiesysteem niet heeft.

We zijn begonnen met een onderzoek om goede hard- en software uit te zoeken. We maken gebruik van het open source navigatiepakket Navit. Dit maakt het gemakkelijk om speciale functies voor de scooters te ontwikkelen. Het navigatiesysteem draait op de SmartQ5. Dit is een MID (Mobile Internet Device), een soort compacte computer.

Een van de belangrijkste nieuwe mogelijkheden is de integratie met de accu's. In de scooters zit een systeem wat constant de actieradius berekent aan de hand van de energiestromen in de scooter. De actieradius wordt vervolgens op het display van het navigatiesysteem weergegeven. Ook krijgt de gebruiker een melding, als blijkt dat de actieradius te klein dreigt te raken. Er wordt dan voorgesteld om naar het dichtstbijzijnde oplaadpunt te navigeren.

Een andere mogelijkheid is, dat de gebruiker bijzondere locaties toe kan voegen aan het navigatiesysteem. De database met oplaadpalen en andere bestemmingen, zijn door de gebruiker zelf te beheren. Hiervoor hebben we een website gemaakt: www.greenav.nl. Als de gebruiker een account aangemaakt heeft op de site, is het mogelijk om bestemmingen toe te voegen. Door op een gemakkelijke manier deze bestemmingen in te delen in locaties en categorieën blijft het geheel overzichtelijk.

Als extra feature kan de gebruiker de bestemmingen koppelen om zo een route samen te stellen. We zien hierin veel mogelijkheden. Zo zou een stad bijvoorbeeld tochten kunnen aanmaken om de stad beter te leren kennen. De deelnemer aan de tocht kan met behulp van het navigatiesysteem alle interessante punten in de stad verkennen.

Het downloaden van de website naar het navigatiesysteem gaat erg eenvoudig. We hebben een speciale downloadtool gemaakt dat automatisch opstart op het moment dat de gebruiker de gegevens vanaf de website wil downloaden. Deze tool configureert een SD-kaart die vervolgens in het navigatiesysteem gaat.

Al met al kunnen we concluderen dat we een sterke basis gemaakt hebben om het gebruik van elektrisch vervoer te promoten. Het systeem is gemakkelijk in het gebruik en klaar voor de toekomst. De samenwerking met het accumeetsysteem en de oplaadpalen vinden we een hele mooie optie. Er is ook zeker nog ruimte tot verbetering. Bij het accumeetsysteem valt nog wel wat te winnen op nauwkeurigheid. Ook zien we mogelijkheden om de software op smartphones te laten draaien. Dit maak het gebruik van ons systeem een stuk goedkoper.

Summary

We have been developing a navigation system for electric scooters during our final project. The main target was to promote the usage of sustainable mobility by showing its full potential. We started with developing a new navigation system to reach this goal. This navigation system should integrate with the scooters and all other aspects within sustainable mobility, including recharge points.

First of all, we did research on current navigation systems. Using open source navigation software Navit, makes it easy to develop a navigation system for this project. The SmartQ5 mobile internet device was chosen to run the software after research.

The navigation system we developed has some new unique features compared to other navigation systems. One of those unique functions is integrating the information about the batteries. There is an embedded system that measures the energy flow through the scooter. These measurements are used to calculate the distance that can be travelled with batteries before they are empty. This number is being sent to the navigation system, where it will be displayed on the screen. The data is also used to warn the end user before the batteries are empty. The system will recommend to drive to the nearest recharge point.

Another feature of the navigation system is to add personal locations. The database with these locations and recharge points can be managed by the end user on our website, www.greenav.nl. The user can register an account at the website to add destinations and images. The website also allows to share these points of interest with the rest of the world. A couple of destinations can be bundled to create a route. This yields options to create a tour around a city. Tourists could hire an electric scooter to see all touristic places in a city.

Downloading the database with all destinations is very easy with the developed download tool. The end user can make a selection of these points at the website. A configuration file will be downloaded to open the download tool. This application will download all necessary files and images from the website and writes them to the SD card. This SD card can be inserted in the navigation system.

We have to conclude that we created a strong basis to promote the usage of electric mobility. The complete system is easy to use and has lots of potential in the future. Especially the integration of the battery management system and recharge points yields a new unique feature. Improvements can also be made on the complete system. The battery management system can be improved on several points. We also see options to port the software to smartphones and usage with other transportation methods like bicycles.

Verklarende woordenlijst

Woord	Betekenis
Actieradius	De afstand die een voertuig nog kan rijden tot de accu's leeg zijn
API	Application Programming Interface, om een applicatie aan te sturen
CAN	Bussysteem dat veel in voertuigen wordt gebruikt voor communicatie
Compileren	Code omzetten in machine taal die de computer kan uitvoeren
Cross-compileren	Compileren voor een andere processorarchitectuur
CSS	Cascade Style Sheet, om de website op te maken
Emulator	Bootst een ander systeem na
Framework	Indeling waarbinnen wordt geprogrammeerd om overzicht te houden
Geocoding	Omzetten van een adres in coördinaten
HTML	Hypertext Markup Language. De taal waarin websites worden gemaakt
Linux	Open source besturingssysteem, concurrent van Windows en dergelijke
Markup language	Opmaaktaal om pagina's in te ontwerpen
MID	Mobile Internet Device, compacte energiezuinige computer
Node	Een apparaat dat met een netwerk is verbonden
Open source	Broncode die vrij inzichtelijk is
OSD	On Screen Display. Informatie die op het scherm wordt getoond
PHP	Taal waarin dynamische websites kunnen worden gemaakt
Render engine	Code die data op het scherm neerzet
RS232	Serieel communicatieprotocol
RS485	Professionelere variant van RS232
SD-kaart	Secure Digital geheugenkaart
SMD	Surface Mounted Device. Component dat op het oppervlak van de printplaat wordt gemonteerd
Statemachine	Code dat bepaalt in welke stap of modus de software zich bevindt
Struct	Een stuk werkgeheugen waarin verschillende soorten data overzichtelijk kan worden opgeslagen
Template	Lay-out van een website waarbinnen de data wordt weergegeven
Userinterface	De lay-out van een programma dat de eindgebruiker krijgt te zien
Wiki	Online encyclopedie en vraagbaak
XML	Extensible Markup Language waarin configuratie gegevens worden opgeslagen

1. Inleiding

De gemeente Groningen wil in 2025 volledig energie neutraal zijn. Dit is een ambitieus plan. Om dit te bereiken hebben een aantal bedrijven, kenniscentra en de gemeente de handen ineengeslagen. Ook het Energie kenniscentrum (EKC) van de Hanzehogeschool is één van de betrokken partijen. Één van de projecten die het EKC uitvoert is het groene scooter project. Dit project is in juni 2008 gestart met als doel om het gebruik van duurzame mobiliteit te promoten. Onze opdracht is erop gericht om het gebruikersgemak van elektrische scooter te verhogen.

Meer informatie over de doelstellingen van de gemeente Groningen en het EKC zijn te vinden op het internet. De adressen staan vermeld in de literatuurlijst.

Om tot een goed einde van ons project te komen hebben we een aantal stappen doorlopen. Doordat we van te voren niet goed konden overzien wat de mogelijkheden van het navigatiesysteem zouden zijn, hebben we de opdracht verdeeld in kleine stukjes waar we onderzoek naar hebben gedaan. Uiteindelijk zijn we op de volgende systeemeisen uitgekomen:

Ons navigatiesysteem moet een aantal functies krijgen die bijvoorbeeld een TomTom niet heeft. Het EKC wil graag dat de actieradius van de scooters constant in beeld staat. Ook moet het systeem in de gaten houden hoe ver het dichtstbijzijnde oplaadpunt is. Als de actieradius te klein dreigt te worden waarschuwt het navigatiesysteem de bestuurder. Dit zal het gebruikersgemak en betrouwbaarheid van elektrische scooters verhogen.

Ook moet het mogelijk worden om naar bepaalde bijzondere bestemmingen te navigeren. Deze bestemmingen moeten door de gebruiker zelf te beheren zijn. Met behulp van een softwarepakket is er dus een soort database samen te stellen. Ook willen we het mogelijk maken om de bestemmingen aan elkaar te koppelen en zo een route te maken. De routes kunnen bijvoorbeeld ingezet worden om de stad Groningen te verkennen op de scooter.

Om de deelsystemen goed te begrijpen is het belangrijk om eerst een goed beeld te krijgen van ons gehele systeem. Daarom geven we daarvan een kort overzicht in hoofdstuk twee. Hierna zullen we verder gaan met het onderzoek dat we gedaan hebben naar de hard- en software van ons navigatiesysteem. Hoofdstuk vier gaat over de navigatiesoftware. Hierna gaan we het hebben over de integratie met de oplaadpalen. Vervolgens gaan we verder met het beschrijven van de methode om de software te laten draaien op de hardware. Hierop aansluitend beschrijven we de hardware die de actieradius berekend. Dan resten ons nog twee softwareonderdelen. Namelijk het softwarepakket dat we gemaakt hebben en de bijbehorende downloadtool. Tot slot trekken we conclusies en doen we nog enkele aanbevelingen.

2. Systeemoverzicht

Voor we dieper in de materie duiken, geven we eerst een overzicht van alle systeemonderdelen. Dit geeft een goed beeld van ons systeem, zodat de rest van de hoofdstukken goed te begrijpen is. Het navigatiesysteem dat wij ontwikkelen bestaat uit meer dan enkel een navigatiesysteem. Het zou namelijk als boordcomputer gezien kunnen worden.

We hebben dus te maken met een aantal componenten. De eerste is natuurlijk het navigatiesysteem zelf. Het tweede onderdeel is de communicatie met de scooter om de gegevens op te halen. Ook moet er communicatie met de computer opgezet worden. Het laatste onderdeel is computersoftware om alle locaties aan te maken. In figuur 1 staat een totaaloverzicht van alle componenten.

Figuur 1 Totaaloverzicht van ons systeem

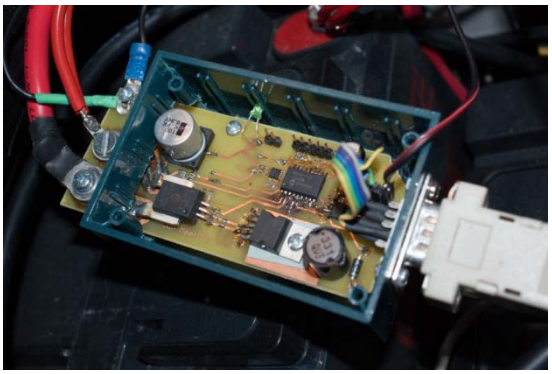
2.1 Navigatiesysteem

Het navigatiesysteem is het component waarop alle data binnen komt. Omdat er niet een kant-en-klaar pakket op de markt is om te testen met communicatie met hardware, moesten we dit zelf maken. Het gebruik van bijvoorbeeld een TomTom valt dus af vanwege de gesloten structuur. Uiteindelijk hebben we gekozen voor een opensource navigatiesysteem en een zogenaamde MID (Mobile Internet Device) waarop het navigatiesysteem zal draaien. We hebben gekozen voor een SmartQ5, dit is een kleine compacte computer. Daarnaast bevat de SmartQ5 een touchscreen. Het onderzoek naar de keuze van het navigatiesysteem en de hardware staat uitgebreid beschreven in hoofdstuk 3.

Het opensource systeem dat we gekozen hebben heet Navit. We hebben Navit zo aangepast dat alle oplaadpalen en interessante locaties in het systeem worden weergegeven. Ook de communicatie met het accumeetsysteem is tot stand gebracht. Uiteindelijk hebben we ons navigatiesysteem een nieuwe naam gegeven, namelijk GreeNav. Dit staat voor Green Navigation.

2.2 Accumeetsysteem

Het accumeetsysteem kan een schatting maken van het resterend aantal kilometers dat nog gereden kan worden met de scooter voordat deze opgeladen moet worden. Dit systeem, zichtbaar in figuur 2, is ontwikkeld door onze voorgangers Patrick de Vries en Stefan Huisman. We hebben in dit systeem een paar aanpassingen gedaan. Dit om de communicatie met GreeNav tot stand te kunnen brengen.



Figuur 2 Het accumeetsysteem

Het resterende aantal kilometers zal, samen met de snelheid, naar het navigatiesysteem op de SmartQ5 worden gestuurd over een RS232-verbinding. Verdere details over de hardware staan beschreven in hoofdstuk 7.

2.3 Website

Om de locaties aan te kunnen maken is een website opgezet. Een website heeft namelijk een aantal voordelen ten opzichte van offline software. Onder andere kunnen alle punten worden gedeeld met de rest van de wereld. De keuze voor een website werd versterkt door de integratie met Google Maps. Door Google Maps te gebruiken, kunnen gebruikers eenvoudig punten aanmaken door de locatie aan te klikken op de kaart. Naast het aanmaken van punten, is het ook mogelijk om deze te bundelen en zo een route te vormen. In figuur 3 is te zien hoe de website er uit ziet.

De gebruiker kan met een eigen account zijn persoonlijke bestemmingen toevoegen. Al deze bestemmingen worden opgenomen in de database. Gebruikers kunnen de bestemmingen en routes vervolgens toevoegen aan een persoonlijke selectie. Deze is vervolgens te downloaden. De website wordt besproken in hoofdstuk 8.

Figuur 3 www.greenav.nl

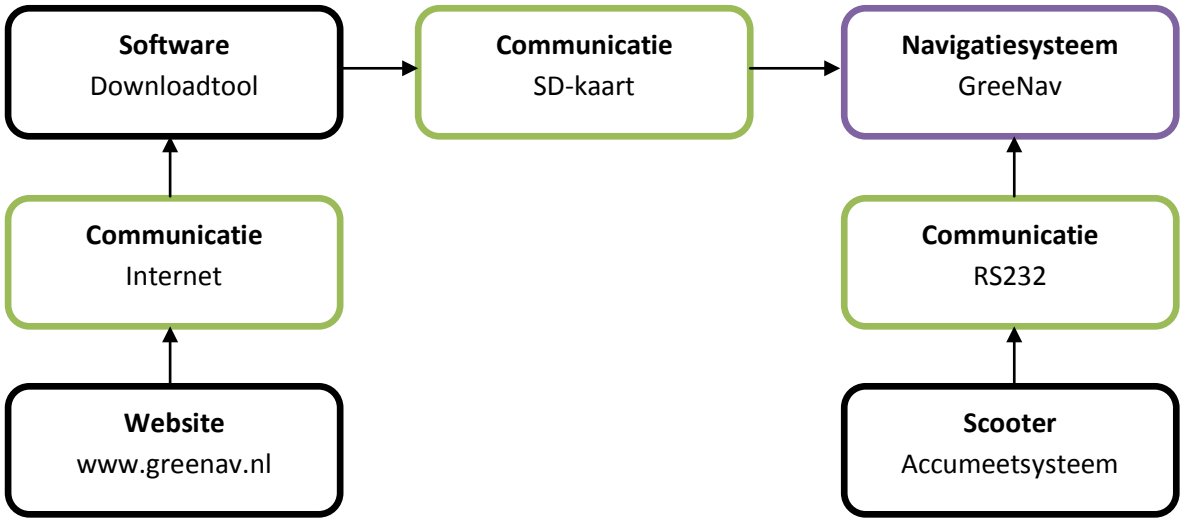
2.4 Downloadtool

Om de gegevens van de website naar het navigatiesysteem te krijgen is een applicatie geschreven. Nadat de gebruiker een selectie heeft gemaakt van alle gegevens die naar het navigatiesysteem moeten, kan er een configuratiebestand gedownload worden. Het gedownloade bestand zorgt ervoor dat de applicatie wordt gestart. Nadat enkele stappen zijn doorlopen, zullen de afbeeldingen van de website worden gedownload, zoals in figuur 4 is te zien. Vervolgens wordt deze data naar een SD-kaart weggeschreven. Als deze SD-kaart in de SmartQ5 gestopt wordt, zijn de gegevens die op de website geselecteerd zijn beschikbaar in GreeNav.

Figuur 4 De downloadtool

2.5 Totaaloverzicht

Wanneer we alle onderdelen samenvoegen, dan zien we het volgende schema met de communicatielijnen:



Schema 1 Schematisch overzicht van het systeem

3. Analyse navigatiesysteem

Het onderzoek begon met het uitzoeken van een geschikt navigatiesysteem om aanpassingen in te maken. Er moest ruimte zijn om een eigen interface te maken welke geschikt was om aan de hand van afbeeldingen te navigeren. Ook moest er de optie zijn om te communiceren met het accumeetsysteem in de scooter.

In eerste instantie betekent dit dat de code vrij inzichtelijk moet zijn, of op zijn minst op deze punten. Hierdoor vallen standaard navigatiesystemen af. Enkel TomTom kent een project om bepaalde delen open te stellen, maar dit was voor ons project te beperkt. De keuze is dan ook gemaakt om een open source navigatiesysteem te nemen.

Ook daar komt weer een probleem naar boven. Deze navigatiesystemen bestaan enkel uit software. Er moet dus ook een apparaat worden gekozen om deze software op te kunnen draaien. Dat geeft ons echter wel de vrijheid om de juiste hardware te kiezen die voor het project volstaat en waar we de mogelijkheid hebben om nog uit te breiden.

3.1 Software

Er zijn een aantal open source navigatieapplicaties te vinden op het internet. Drie van die applicaties bevonden zich in een vergevorderd stadium. Dit zijn Navit, RoadNav en We-travel.

3.1.1 Navit

Navit is de grootste van de genoemde systemen. Het kent een uitgebreide community van ontwikkelaars die aan het project werken. Ook is er veel algemene project documentatie te vinden in de wiki van dit project. Tevens is het systeem zeer uitgebreid en geschikt voor diverse systemen.

Figuur 5 Het logo van Navit

3.1.2 RoadNav

Ook RoadNav heeft potentie. De software ziet er simpel uit, maar is wel uitbreidbaar. Ook hier vinden we een wiki terug, maar deze is wel minder uitgebreid. Ook is de compatibiliteit met andere systemen minder ver gevorderd als bij Navit.

3.1.3 We-travel

We-travel is ook een navigatiesysteem. Dit systeem is echter in JAVA geschreven. Ook is het veel minder geavanceerd en ziet er zowaar spartaans uit.

Uiteindelijk hebben we voor Navit gekozen. We-travel was door de beperkte mogelijkheden toch al geen serieuze kandidaat. Tussen Navit en RoadNav viel de keuze op Navit voornamelijk door de grootte van het project. Navit heeft meer algemene documentatie en opties waardoor het eenvoudiger wordt om het systeem naar onze hand te zetten.

3.2 Hardware

Voor de hardware was er maar een beperkt budget beschikbaar. Daarnaast moest het apparaat op Linux draaien om de beste compatibiliteit met Navit te verkrijgen. Ook moest het apparaat een GPS-module en een communicatiemogelijkheid met het accumeetsysteem bevatten. Een touchscreen zou tevens veel toegevoegde waarde bevatten voor de bediening.

We kwamen twee geschikte kandidaten tegen om te gebruiken. De eerste was de OpenMoko Freerunner, de ander was de SmartQ5.

3.2.1 OpenMoko Freerunner

Dit is een open smartphone. Hiervan zijn alle schema's beschikbaar op het internet. Verder bevat de telefoon veel componenten zoals een GPS-module en bluetooth. Ook bevat dit apparaat een 2,8" touchscreen en USB-aansluiting. Het apparaatje draait op Linux en bevat een ARM-processor.

3.2.2 Smartdevices SmartQ5

De SmartQ5, te zien in figuur 6, is een zogenaamde MID (Mobile Internet Device). Dit is een kleine computer welke op Linux draait. Verder bevat ook dit apparaat een ARM-processor en bluetooth module. Het touchscreen van dit apparaat is met 4,3" een stuk groter. Een GPS-module ontbreekt echter.

We hebben gekozen voor de SmartQ5. Het open idee achter de OpenMoko sprak ons erg aan. Echter is een 2,8" scherm aan de kleine kant voor een navigatiesysteem. Uiteindelijk is gebruikersgemak één van de belangrijkste onderdelen van het navigatiesysteem.

Het ontbreken van een GPS-module hebben we opgelost door een aparte USB-ontvanger aan te sluiten. We hebben gekozen voor de Navilock NL-402U. Dit is een goedkope GPS-module die ook geschikt is voor Linux. Daarnaast hebben we nog een USB-hub toegevoegd om ook de communicatie met de hardware te kunnen maken.

Figuur 6 De SmartQ5

4. Navigatiesoftware GreeNav

Het navigatiesysteem dat op de SmartQ5 komt, is het open source navigatiesysteem Navit. In deze software moeten de oplaadpunten en afbeeldingen komen. Ook zal hierin de communicatie met het accumeetsysteem moeten worden geschreven. De aangepaste software die we ontwikkeld hebben voor de scooters hebben we uiteindelijk GreeNav genoemd.

4.1 Ontwerp

Voor het navigatiesysteem hebben we een simpel ontwerp gemaakt. De bediening van het apparaat moet gelijk duidelijk zijn voor de gebruikers, waarbij een handleiding niet nodig zou moeten zijn. Steeds terugkomende elementen zijn de terug-knop en de home-knop. Ook is duidelijk aangegeven waar de gebruiker zich bevindt. Verder maken we gebruik van grote knoppen zodat het systeem gemakkelijk te bedienen is met de vinger.

Figuur 7 Een overzicht van categorieën in GreeNav

Verder maken we optimaal gebruik van de 800x480 pixels die de SmartQ5 ons biedt. Om de leesbaarheid te vergroten hebben we gebruik gemaakt van witte letters op een zwarte achtergrond. Daarmee is de tekst beter te lezen bij fel licht. Tijdens het navigeren maken we gebruik van een aantal indicatoren om toch veel informatie te tonen zonder dat dit ten koste gaat van de oppervlakte van de kaart. In figuur 7 en 8 worden twee schermen van het navigatiesysteem getoond.

Figuur 8 Bestemmingsinformatie in GreeNav

4.2 Code binnen de bestaande functies in Navit

De gebruikte navigatiesoftware, Navit, bestaat uit veel code. Ondanks de vele algemene documentatie is de code minder goed gedocumenteerd. Om de eigen toegevoegde code overzichtelijk te houden, proberen we deze zo veel mogelijk los te houden van de standaard code. Daardoor wordt ook een update naar een nieuwere versie in de toekomst gemakkelijker.

4.2.1 Inlezen van het XML-configuratiebestand

Een belangrijk bestand in Navit is het XML-configuratiebestand. In dit bestand kunnen vele onderdelen van Navit geconfigureerd worden, zoals bijvoorbeeld de indeling van het scherm. Om de configuratie in te lezen gebruiken we de functies van osd_core.c. Dit is een van de belangrijkste bronbestanden van de oorspronkelijke Navit. We hebben hierin een nieuw component aangemaakt. Deze is in de XML terug te vinden onder de naam "gsn_xml". Wanneer deze bij het inladen van de XML-bestand wordt aangetroffen, zal een functie in osd_code.c worden aangeroepen. Over de exacte implementatie van het XML-bestand is verderop in het verslag meer te lezen.

4.2.2 Gebruik van labels

Om tekst op het display te zetten maken we gebruik van het 'text'-item dat in Navit te vinden is. De functie die voor alles zorgt, is ook in osd_core.c terug te vinden. In deze functie is code toegevoegd om de label aan te passen. Deze code geeft de inhoud van het label door naar onze eigen code middels een functie. Daar wordt, aan de hand van de statemachine, bepaald wat de huidige tekst moet zijn van dat label. Deze wordt door de functie geretourneerd, waarop deze waarde zal worden weergegeven.

4.2.3 Gebruik van afbeeldingen en knoppen

Ook bij de afbeeldingen hebben we een dergelijke constructie aangemaakt. We hebben in dit geval de complete functie gekopieerd en aangepast. Opvallend binnen Navit is dat zowel een knopje als afbeelding hetzelfde onderdeel is. Een knop bevat een afbeelding, terwijl een afbeelding eerder een knop is zonder toegewezen functie.

De weer te geven afbeelding wordt opgehaald middels een functie. Deze heeft, net als bij de label, een parameter om aan te geven om welke knop het gaat. De functie zal daarop een string terugsturen met de afbeeldinglocatie.

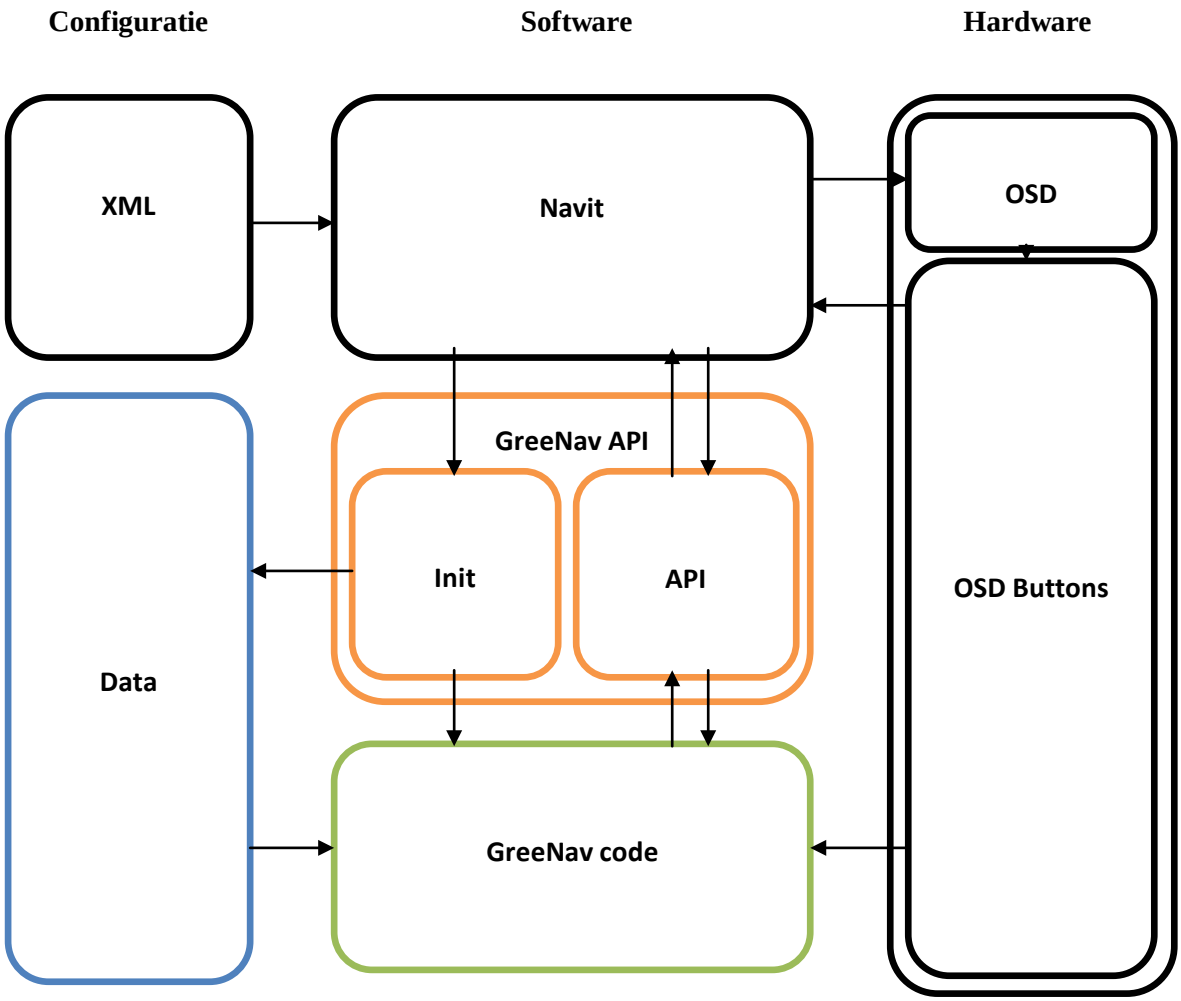
Het aanroepen van de juiste functie bij het drukken op de knop gaat wel middels de normale manier die Navit kent. Binnen navit.c kan worden aangegeven welke functie moet worden aangeroepen wanneer er op de knop wordt gedrukt.

4.3 Organisatie code

Aangezien het navigatieprogramma behoorlijk complex is, hebben we met elkaar duidelijke afspraken gemaakt over de indeling. Navit zelf heeft een behoorlijke duidelijke indeling welke we deels hebben overgenomen. Ook hebben we een laag geschreven welke zorgt voor een gemakkelijkere aansturing van de originele code vanuit de eigen code. Op deze manier maken we de software niet moeilijker dan deze al is.

4.3.1 Organisatie

Om beter te begrijpen hoe onze code in elkaar steekt, is het belangrijk het totale plaatje van de software eerst te begrijpen. In het onderstaande schema is te zien hoe de diverse onderdelen zich tot elkaar verhouden. Ook valt af te leiden hoe de communicatie tussen de diverse onderdelen verloopt. Weggelaten is echter de communicatie met andere onderdelen zoals GPS. Deze onderdelen zijn al aanwezig in Navit, en daar hoeven we niets aan te veranderen.



Schema 2 Schematische opbouw van GreeNav

XML:

Het XML-bestand bevat alle configuratiegegevens voor Navit. Hierin staan ook de configuratiegegevens voor alle interessante punten met de afbeeldingslocaties. Het XML-bestand kan enkel door Navit worden ingelezen. Echter zijn enkele onderdelen hiervan gekoppeld aan de GreeNav API.

Navit:

Navit is de gewone, originele, code van de navigatiesoftware. Deze leest het XML-bestand uit en stuurt ook de hardware aan, Zo wordt onder andere de GPS uitgelezen, maar wordt ook het beeld opgebouwd.

OSD:

Het On Screen Display (OSD) wordt door Navit aangestuurd en opgebouwd. De configuratie hiervoor staat in de XML.

OSD Buttons:

OSD Buttons hebben een speciale functie. Ze worden, net als de rest van het OSD, door Navit op het scherm geplaatst. Wanneer er echter op gedrukt wordt zal een commando worden gegeven. Deze is gelijk te koppelen aan een functie in de C-code.

GreeNav API - Init:

Het initialisatie deel van de GreeNav API zorgt er voor dat de software in de goede toestand komt te staan. Daarnaast heeft het ook de functie om koppelingen vast te leggen. Het belangrijkste onderdeel is wellicht het opvangen van eigen configuratiegegevens vanuit de XML.

GreeNav API - API:

De API zelf is het belangrijke communicatiedeel tussen onze code, en die van Navit. Vanuit de code kunnen functies van de API worden aangeroepen om zo functies van Navit op een eenvoudige manier te kunnen gebruiken. Denk aan het instellen van een nieuwe bestemming, of het opnieuw opbouwen van het scherm.

Data:

Voor de opslag van configuratiegegevens gebruiken we een struct. Dit is een stuk werkgeheugen waarin verschillende soorten data overzichtelijk kan worden opgeslagen. Op deze manier bewaren we de gegevens uit de XML, zoals de locaties, categorieën en uiteraard de bestemmingen. Deze data wordt door de code gebruikt om het navigatiesysteem aan de hand van deze locaties te laten navigeren.

GreeNav code:

De GreeNav code is de uiteindelijke code die alle handelingen vertaald in API functieaanroepen en bepaald wat er op het scherm moet komen. De data hiervoor wordt uit de struct gehaald. De handelingen zelf worden gestart door commando's vanuit de knoppen op het scherm of gebeurtenissen die door Navit worden gestart en door de API worden doorgestuurd.

4.3.2 Naamgeving

Nieuwe functies in het navigatieprogramma zijn ingedeeld per doel. Daarmee zijn alle functies die een bepaalde taak vervullen netjes bij elkaar gezet. Dit zijn onder andere de API en het aansturen van het OSD. De nieuwe functies en variabelen beginnen met de prefix "gsn_". De letters "gsn" staan hierin voor Groene Scooter Navigatie, de tijdelijke naam die we aan het programma meegaven. Na deze prefix zal worden aangegeven waar de functie bij hoort, zoals het aansturen van het OSD bijvoorbeeld. De functie om het scherm op nieuw op te bouwen heet dan ook "gsn_osd_redraw()". Op deze manier blijft de code overzichtelijk en zijn de nodige functies snel terug te vinden. Ook de eigen variabelen beginnen met de "gsn_"-prefix.

4.3.3 Greenav API

Om gemakkelijk en onafhankelijk van Navit te kunnen programmeren hebben we een API opgezet. API staat voor Application Programming Interface. Deze API vormt een tussenlaag tussen onze code en die van Navit zelf.

Deze API bestaat uit een aantal belangrijke functies om Navit aan te sturen. Voorbeelden zijn het instellen van een nieuwe bestemming, of het opnieuw laten opbouwen van de OSD (On Screen Display). Het voordeel van deze tussenlaag is dat hierdoor geen verdere kennis van Navit vereist is. Het is tevens gemakkelijker om functies te verbeteren, zonder dat de rest van de code moet worden herschreven.

De API bestaat grotendeels uit functieaanroepen die al in Navit aanwezig waren. Door de benodigde functieaanroepen te bundelen in de eigen functies van de API is het eenvoudiger om te kunnen programmeren. Dat komt, naast de documentatie, doordat enkel de benodigde functies in de API bevinden. Daardoor hoef je niet door de complete code heen. Verderop in het verslag zijn de functies die de API bevat uitgelegd.

4.4 XML-configuratie

Het XML-configuratiebestand bevat alle instellingen voor Navit. Hieronder vallen onder andere de opbouw van het scherm en het te gebruiken GPS-ontvanger. Hierin is ook de configuratie met interessante bestemmingen en routes opgenomen.

4.4.1 Navit XML-gegevens

Navit bevat veel opties voor de configuratie van het programma. Dit zijn onder andere de schermopbouw, de gebruikte render engine, de GPS-module en parameters voor het bepalen van de route. Een uitgebreide uitleg van alle bestaande functies binnen deze XML, genaamd navit.xml, is te vinden op: http://wiki.navit-project.org/index.php/Configuring_Navit

4.4.2 Bestandslocatie

Normaalgesproken staat het configuratiebestand navit.xml in de documentenmap of elders op de vaste opslag van de computer. De meest gebruikte locatie is:

```
~/.navit/navit.xml
```

Dit is echter geen handige locatie wanneer het configuratiebestand gemakkelijk aan te passen moet zijn. Daarom is gekozen om de configuratie te verplaatsen naar een locatie waar deze eenvoudig aan te passen is. Voor de SmartQ5 is dit de SD-kaart. Deze vorm van opslag is gemakkelijk uit het apparaat te halen om vervolgens op een vaste computer de configuratie uit te voeren. De nieuwe plaats voor het configuratiebestand is als volgt:

```
/media/SDDATA/Greenav/navit.xml
```

Om deze alternatieve locatie te kunnen gebruiken is de code lichtelijk aangepast. Binnen start.c is de procedure voor het inladen van het XML-bestand dusdanig aangepast, zodat het configuratiebestand vanaf hier wordt ingeladen.

4.4.3 GreeNav configuratie

Voor de configuratie van de GreeNav-onderdelen is een nieuw element aangemaakt binnen het configuratiebestand. Deze heeft de naam "gsn_xml" mee gekregen. Een voorbeeld hiervan in de XML is:

```
<osd type="gsn_xml" label="init|start"/>
```

De waarde staat vervolgens in het label, waarvan de inhoud doorgestuurd wordt naar de GreeNav API.

De commando's voor het configureren worden op type gelabeld. Zo zal een functie die te maken heeft met de initialisatie, met "init" beginnen. Daarna volgt een streep ("|") waarna een parameter wordt opgegeven. Dit kunnen ook meerdere parameters zijn, allen met een streep ("|") gescheiden.

4.5 GreeNav API

De GreeNav API is de tussenliggende laag in de software. API staat voor Application Programming Interface. Het doel er van is dan ook om het programmeren gemakkelijker te maken. De interface zelf is een verzameling van functies, die op hun beurt weer communiceren met de applicatie. Je zou kunnen zeggen dat je hiermee een applicatie binnen een applicatie kunt bouwen.

Dat is ook wat in GreeNav gebeurt. De code die voor het navigeren op basis van afbeeldingen en bestemming zorgt, staat los van de originele Navit code. Middels deze API kunnen gemakkelijke de belangrijkste functies worden aangeroepen. Daarbij is geen kennis van Navit vereist en is voor het ontwikkelen van nieuwe functies dus ook niet meer nodig om in de vele regels code daarvan te duiken.

De API bestaat uit twee onderdelen. De eerste is de initialisering van de software terwijl de tweede zorgt voor het aansturen van Navit wanneer het programma draait.

4.5.1 Init

Het initialisatie deel, genaamd init, bevat de functies die nodig zijn om alle data op zijn plaats te zetten. Deze wordt geheel gestart aan de hand van de commando's in de XML. Het grootste onderdeel is dan ook de functie die de invoer vanuit de XML omzet.

Andere belangrijke onderdelen zijn het reserveren van ruimte voor de data-struct. Ook zorgt de initialisatie API er voor dat de belangrijke variabelen hun juiste initiële toestand krijgen. Daarnaast worden belangrijkste koppelingen, met de rest van de Navit software, vastgelegd. De belangrijkste onderdelen van de initialisatie:

- Initialiseren waarden
- Ruimte reserveren voor de arrays
- Waarden uit de XML in de data-struct zetten

Functies zijn te herkennen aan "gsn_init_..."

4.5.2 API

Het andere deel van de API is belangrijk voor het aansturen van Navit tijdens het navigeren. Deze API kan ook vanuit de Navit code worden aangeroepen en daardoor GreeNav beïnvloeden als het nodig is, bijvoorbeeld als de bestemming bereikt is.

De API wordt echter voornamelijk gebruikt voor de aansturing van Navit aan de hand van user input. Deze is gelijk op te vangen vanuit het drukken op een knop. Daarna kan, na het uitvoeren van de code, de uitvoer via de API aan Navit worden doorgegeven. Dit zijn onder andere het instellen van een nieuwe bestemming, of het opnieuw opbouwen van het display. De belangrijkste onderdelen van de API:

- Bestemming opgeven
- Display opnieuw opbouwen

Functies zijn te herkennen aan "gsn_api_..."

4.6 GreeNav code

De GreeNav code is de daadwerkelijke code die zorgt voor het navigeren op basis van afbeeldingen en bestemmingen. De opbouw van de code gaat per onderdeel. De functies die met dat onderdeel, bijvoorbeeld de statemachine, te maken hebben beginnen ook met "gsn_" gevolgd door deze naam. Een functie die begint met "gsn_osd_" zal dan ook te maken hebben met het aansturen van het on screen display.

De code bepaalt wat de software moet doen en weergeven. Zo zijn er onder andere variabelen om de gekozen locatie, categorie en bestemming te onthouden. Samen met de statemachine bepalen deze de inhoud van het scherm. Ook acties zoals het instellen van een nieuwe bestemming, worden door de statmachine bepaald. Verder worden ook gegevens als de actieradius en snelheid daar bijgehouden.

4.6.1 Menu statemachine

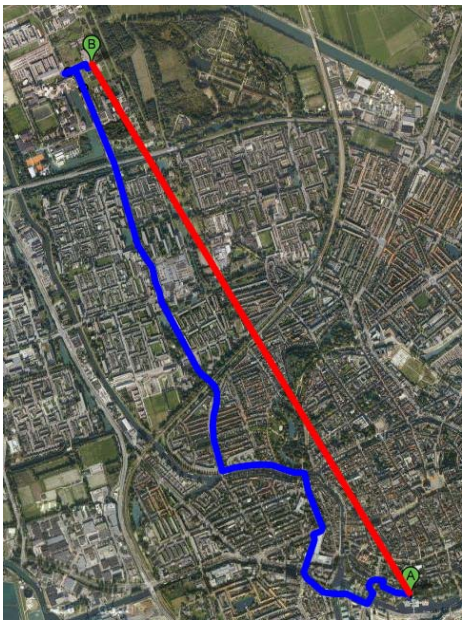
De menu statemachine bepaalt in welk stadium de software is. Deze kan voor een groot deel gelijk worden getrokken met de schermen die worden getoond. Ook worden de gekozen locatie, categorie en bestemming onthouden. Aan de hand van deze data kan de aansturing van het OSD bepalen welke knoppen, labels en plaatjes moeten worden weergegeven op het scherm.

5. Afstand tot oplaadpalen berekenen

Een belangrijk onderdeel van het navigatiesysteem is de integratie met de oplaadpalen en de gegevens vanuit de scooters. Hiermee kan het gebruikersgemak van een elektrische scooter fors worden verbeterd. Één van de belangrijkste onderdelen is dan ook om de data te combineren om de gebruiker op tijd naar een oplaadpaal te kunnen leiden.

Er is echter één probleem op dit vlak: rekenkracht. Als het navigatiesysteem een paar keer per minuut de afstand naar een aantal palen moet bepalen, dan kost dat veel rekenkracht om de kortste route te bepalen. Daarvoor moesten we dus een oplossing zoeken.

Figuur 9 Het oplaadpunt op het Zernike



Figuur 10 Rood: hemelsbrede afstand
Blauw: werkelijke route

5.1 Hemelsbreed de afstand bepalen

Een minder zware berekening is de directe lijn tussen 2 coördinaten bepalen. In dat geval heb je slechts één berekening per oplaadpaal in plaats van een zware routeberekening. Daarmee belasten we de SmartQ5 zo min mogelijk met deze taak. De code om de afstand te bepalen is als volgt:

```
theta = lon1 - lon2;
dist = sin(deg2rad(lat1)) * sin(deg2rad(lat2)) + cos(deg2rad(lat1)) *
cos(deg2rad(lat2)) * cos(deg2rad(theta));
dist = acos(dist);
dist = rad2deg(dist);
dist = dist * 60 * 1.1515 * 1.609344;
```

Het hemelsbreed bepalen van de afstand kent echter ook een probleem: nauwkeurigheid. Een weg loopt nooit direct naar de bestemming toe, zoals te zien is in figuur 10. Dit heeft als gevolg dat de daadwerkelijke route in de meeste gevallen langer is dat de hemelsbrede afstand. Om het geheel nauwkeuriger te maken hebben we een aantal punten bepaald. Tussen die punten is de afstand bepaald middels de hemelsbrede berekening en de route die Google Maps ons voorstelde.

5.2 Analyse

In totaal hebben we 50 meetgegevens genomen voor dit onderzoek. De meeste gegevens zijn genomen binnen de steden Groningen en Drachten. We hebben echter ook punten uit de omgeving meegenomen. De maximale afstand bedraagt daarbij 60km, maar de meeste meetgegevens vallen binnen een afstand van 30km, een reële afstand voor de scooters. Ook zijn er nog een aantal verre punten mee genomen om te onderzoeken wat het effect is voor auto's.

Deze meetgegevens hebben we eerst gegroepeerd in verschillende groepen. De eerste groep is de afstand binnen de steden. Hier blijkt de gemiddelde afwijking rond de 40% te liggen. Net buiten de stad neemt de afwijking iets af tot 37%. Op langere afstanden van 10 tot 30km zien we het verschil nog verder afnemen tot gemiddeld 24%.

Deze eerste resultaten komen overeen met de verwachting die we hadden. Naarmate de afstand groter wordt, zal ook de kans op een directere route toenemen. Opvallend is echter dat op nog langere afstanden de afwijking weer groter wordt. Dit geldt dan voor auto's. De verklaring hiervoor is dat men in dat geval de snelste route neemt: die van de snelweg. Deze wegen lopen echter minder direct naar de eindbestemming.

5.3 Formule opstellen

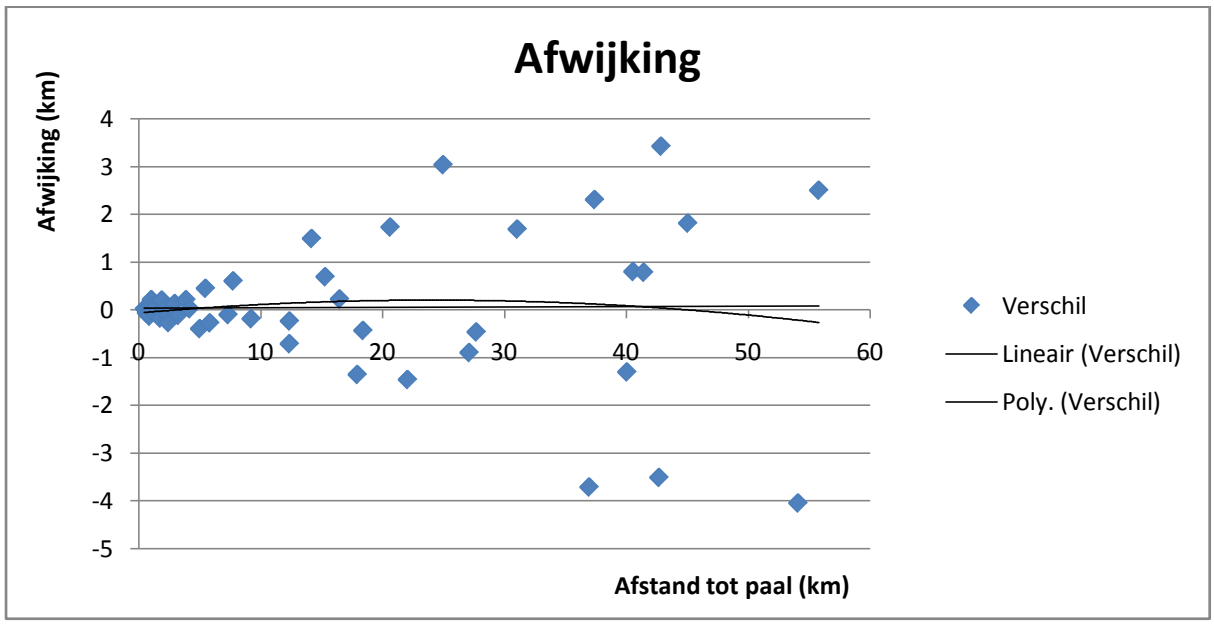
Uit deze gegevens moest een formule worden opgesteld. Om de nauwkeurigheid te verbeteren hebben we de lange afstanden weggelaten en is de focus gelegd op scooters en fietsen. Door alle gegevens in een grafiek te zetten kunnen

we in Excel een trendlijn genereren met bijbehorende formule, waarin s staat voor de afstand in meters. Deze formule is nog iets aangepast en is uiteindelijk als volgt geïmplementeerd:

$$s_{benadering} = (1{,}3 * 10^{-11} * s_{hemelsbreed}{}^{2} - 0{,}6 * 10^{-6} * s_{hemelsbreed} + 1{,}4) * s_{hemelsbreed}$$

Formule 1

Als we de afstand met deze formule berekenen, dan komt deze al aardig in de buurt van de afstand die Google Maps aangeeft. Ook een grafiek in Excel laat dit zien:



Grafiek 1 Afwijking tussen de berekende afstand en de daadwerkelijke afstand

In deze grafiek valt goed te zien wat de relatie is tussen de echte afstand en de afwijking van de formule. De afwijking neemt toe naarmate de afstand tot de paal groter wordt. Deze punten zijn goed verdeeld aan de positieve en negatieve kant. Dat zien we ook terug in de trendlijnen. De lineaire lijn ligt bijna exact op de nul-lijn. De polynome trendlijn laat slechts een lichte buiging zien.

Met deze formule zijn we er echter nog niet. De afwijking kan nog altijd negatief uitpakken. Juist het met een lege accu komen te staan willen we voorkomen. Daarom gaan we uit van het "worst case"-scenario. Dat betekent dat we nog een berekening moeten toevoegen om altijd positief uit te komen.

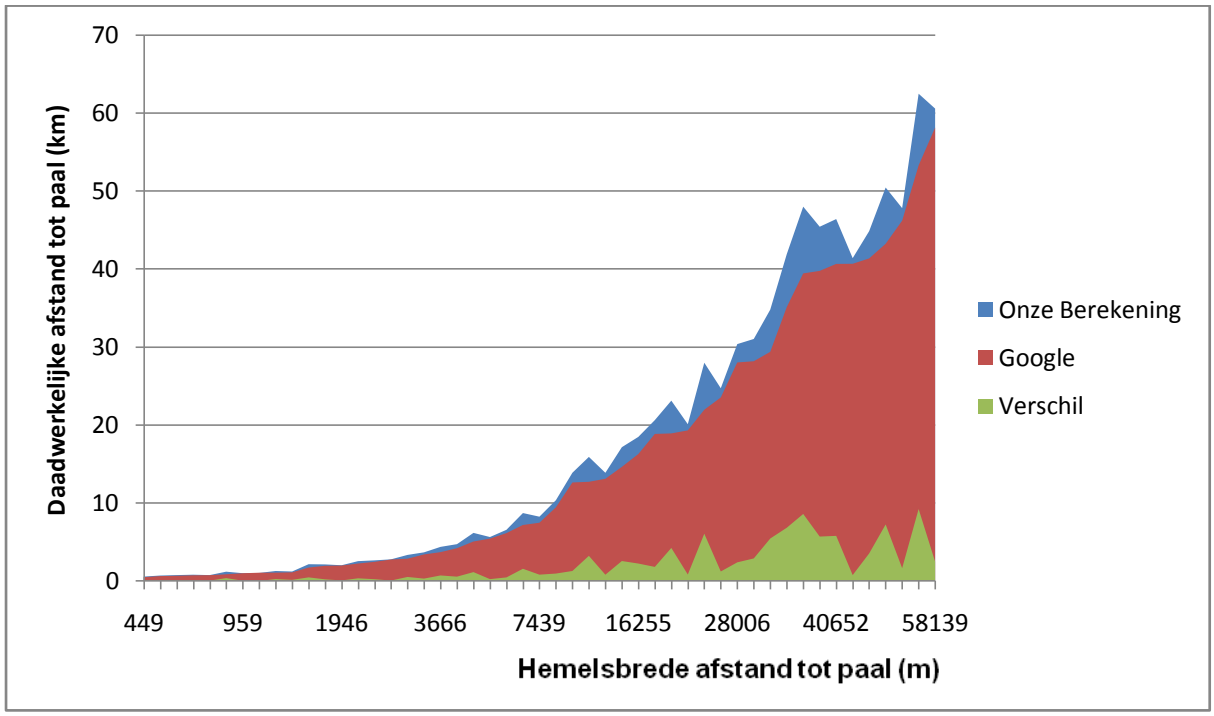
De afwijking in de grafiek loopt vrijwel lineair, ook als we op kleinere delen inzoomen. Daarmee kwamen we uit op een correctie van 120 meter per kilometer. Dat betekent dat het eindantwoord van de formule nog eens met 1,12 moet worden vermenigvuldigd. De uiteindelijke formule is dan:

$$s_{benadering} =$$

$$(1{,}3 * 10^{-11} * s_{hemelsbreed}{}^{2} - 0{,}6 * 10^{-6} * s_{hemelsbreed} + 1{,}4) * s_{hemelsbreed} * 1{,}12$$

Formule 2

Als we de afstand met deze formule berekenen, dan zien we dat we vrijwel altijd positief uitkomen. In slechts 1 geval blijkt dat we 31 meter te kort komen. Dit is ook te zien in de volgende grafiek:



Grafiek 2 Onze berekening ten opzicht van Google

In de grafiek zien we terug dat onze berekening er altijd boven zit. Naarmate de afstand groter wordt zal ook de mogelijke afwijking toenemen. Als we echter kijken naar de belangrijkere korte afstanden, dan zien we een zeer betrouwbaar resultaat. Tot 4km noteren we een maximale afwijking van 500 meter, terwijl de afwijking niet boven de 2km komt bij afstanden tot 10km.

Op deze manier kunnen we een vrij betrouwbare berekening uitvoeren zonder dat deze veel rekenkracht kost. Wel moeten we er rekening mee houden dat in uitzonderlijke gevallen, bijvoorbeeld bij het oversteken van een kanaal, de berekening nog steeds niet nauwkeurig is. Voor de afstanden die bij een scooter van belang zijn, blijft de afwijking binnen de perken en ligt in dezelfde orde van grote als de afwijking die het accumeetsysteem zou kunnen vertonen.

5.4 Implementatie

De formule die we aan de hand van de analyse gevonden hebben is geïmplementeerd in de code. We hebben echter nog meer optimalisatie toegepast op dit systeem. Dit omdat het aantal oplaadpunten in de database kan toenemen. Niet alle oplaadpunten zijn echter altijd van belang. In de meeste gevallen zullen veel oplaadpunten buiten het rijbereik van de scooter liggen. Deze punten hoeven we dus niet mee te nemen in onze berekening. We hebben drie categorieën gemaakt:

- Uitgeschakeld
- Actief
- Prioriteit

Bij de initialisatie zullen de oplaadpalen die zich verder dan 50km van de huidige locatie bevinden worden uitgeschakeld. Deze hebben geen nut voor de scooter gezien de actieradius. De actieve punten bevinden zich binnen deze 50km afstand. De afstand tot deze punten wordt iedere 2 minuten bepaald. Dan zijn er nog punten met een hoge prioriteit. Dit zijn de punten die zich binnen een straal van 10km bevinden. Van deze punten zal elke 10 seconden de afstand tot de paal worden bepaald.

Met deze methode sparen we nog meer rekenkracht. Wanneer de gebruiker buiten bereik van een paal dreigt te komen zal een waarschuwing worden gegeven. Deze kan ook worden gegeven op het moment dat de gebruiker slechts 10km aan acculading over heeft. Bij het bepalen nemen we nog een extra marge van 3km mee om zo een eventuele meetfout aan de accu op te vangen. Op deze manier kan de eindgebruiker altijd zijn bestemming bereiken.

6. GreeNav op de SmartQ5

Om GreeNav op de SmartQ5 te krijgen hebben we behoorlijk wat moeite gedaan. In eerste instantie lijkt het verhaal hetzelfde als op de computer: Installeer de pakketten en compileer GreeNav. Zo gemakkelijk gaat dat echter niet door het gebrek aan opslagcapaciteit.

6.1 Onderzoek voor het compileren

De eerste poging die we hebben uitgezocht was dan ook het verwijderen van ongebruikte pakketten. Aan software, zoals een PDF-reader en mailclient, hebben we niets bij het navigeren. Op deze manier hebben we getracht genoeg ruimte vrij te maken op de SmartQ5 voor de benodigde bestanden. Dit bleek echter niet het geval.

We moesten op zoek naar meer ruimte. Het opslagmedium van de SmartQ5 bestaat uit een stuk flashgeheugen met een capaciteit van 1GB. Daarvan is zo'n 660MB beschikbaar voor het systeem en applicaties. De overige ruimte is in tweeën gedeeld, waarbij de ene helft swap is, terwijl de andere partitie gebruikt wordt voor persoonlijke documenten. Die laatste is zo'n 130MB groot. Ruimte zat voor alle applicaties, maar daarvoor moest deze wel worden toegevoegd aan het systeemgeheugen. Helaas wil het samenvoegen van deze partities niet lukken. Dat komt doordat de schijf in gebruik is, immers is Linux daar geïnstalleerd en geladen.

Figuur 11 Greenav op een denkbeeldig apparaat

Op een normale computer zou je een live cd kunnen laden en met een Linux-installatie op die schijf de partities opnieuw in kunnen delen. Op de SmartQ5 is dat niet mogelijk. Pogingen om de installatie directory van pakketbeheerprogramma Aptitude te wijzigen gaven ook geen resultaat.

Na dit onderzoek moesten we concluderen dat compileren op de SmartQ5 zelf niet ging lukken. Daarom zijn we gaan kijken naar oplossingen om dit op de computer te doen. Normaalgesproken geeft dit ook geen problemen met het compileren van exe-bestanden voor Windows. Deze draaien immers op elke computer.

Het probleem is echter dat de SmartQ5 een andere processor heeft als de computers. Dat is in beginsel al de reden dat we Linux gebruiken, aangezien de standaard versies van Windows er niet op werken. De processor in de SmartQ5 kent een ARM11-instructieset. De processors in de computer zelf kennen een x86 (i386) of x86-64 (AMD64) instructieset.

Om de code op onze computers te kunnen compileren voor de ARM11-instructieset zijn er twee opties. De eerste is cross-compileren. Deze optie is echter complex om aan de gang te krijgen. Binnen Linux moet van alles worden ingesteld. Daarbij werken niet alle instructies op het internet, wat tot gevolg kan hebben dat de complete installatie van Linux om zeep wordt geholpen.

De tweede optie is het compileren via een virtual machine. Met Qemu kunnen we een ARM11 processor emuleren. Daarmee is het net alsof we een ARM11 processor in onze computer hebben. Daardoor is het platform gelijk aan dat van de SmartQ5. Het nadeel is echter dat alle code moet worden vertaald. Dat laatste heeft als gevolg dat het proces zeer traag verloopt. Het compileren op zich kan dan ook zo een uur in beslag nemen. Ter vergelijking, op de directe x86-architectuur duurt dit ongeveer een minuut.

6.2 Compileren middels emulator

Middels Qemu kunnen we dus een ARM11 processor emuleren. Hiervoor zijn zogenaamde images beschikbaar. Dat zijn kopieën van harde schijven, waar het besturingssysteem en de emulatie-instellingen al klaar staan. De image die wij gebruiken bevat een installatie van de Linux distributie Debian. Dat is de basisdistributie waar ook Ubuntu (wat we op de computers gebruiken) en het besturingssysteem van de SmartQ5 op zijn gebaseerd. Daardoor is het geheel goed compatibel met elkaar.

Nadat de emulator is ingesteld, kan een map worden gekoppeld aan de emulator. Deze is dan vanuit de emulator te benaderen en op die manier kan de broncode worden overgezet naar de emulator.

Daarna kunnen de voor GreeNav benodigde pakketten worden geïnstalleerd middels Aptitude. Vervolgens kan GreeNav normaal worden gecompileerd in de emulator. Op dat moment krijgen we een uitvoerbaar bestand voor ARM11 processors terug. Deze kunnen we vervolgens inpakken met Tar, een soort Winzip voor Linux.

6.3 GreeNav overzetten naar SmartQ5

Het ingepakte bestand kunnen we vervolgens op een SD-kaart of USB-stick zetten. Waarna we deze in de SmartQ5 steken. Op dat moment is het slechts een kwestie van uitpakken en uitvoeren. Dat laatste kan middels de console.

In de bijlagen is een uitgebreide handleiding te vinden voor het installeren van Qemu, het compileren en de vereiste handelingen op de SmartQ5.

6.4 GreeNav op de SmartQ5

Nu GreeNav voor de SmartQ5 geschikt is, kunnen we deze testen op het apparaat zelf. Het starten kan middels de console, maar mooier is om dit door het apparaat zelf uit te voeren zodra deze is opgestart. In dat geval kan ook de USB-host functie worden geactiveerd om verbinding met de hardware en GPS-module te maken. Op deze manier hoeft de eindgebruiker het apparaat enkel te starten met de powerknop, waarop het navigatiesysteem vanzelf zal starten en klaar is voor gebruik.

Tijdens het draaien van eerdere testversies liep het systeem nog redelijk vlot door de aangemaakte database van locaties. Hierbij was echter nog geen GPS ontvangst aanwezig en ook de hardware was nog niet aangesloten. Bij de uiteindelijke versie van GreeNav, te zien in figuur 13, moeten we helaas vaststellen dat de hardware niet krachtig genoeg is.

Zolang we geen ontvangst hebben en ook de hardware in de scooter niet gebruiken loopt het navigatiesysteem op de SmartQ5 nog prima. Wanneer we echter gaan navigeren of de hardware uitlezen lijkt de software te crashen. Dit blijkt echter niet het geval als we naar de log kijken. Hierin blijkt dat de software nog gewoon loopt en de afstand naar de dichtstbijzijnde berekent.

Hieruit kunnen we opmaken dat de processor van de SmartQ5 niet krachtig genoeg is om alle taken uit te voeren. De software is constant bezig om alle binnenkomende data vanaf de GPS-module en de hardware te verwerken. Zodra de processor klaar is met het verwerken van deze data, is er weer nieuwe beschikbaar die verwerkt moet worden. Hierdoor is er geen rekenkracht beschikbaar voor de verwerking van gebruikersinvoer, zoals het maken van een keuze.

Op de laptop doet dit probleem zich dan, door de beschikbare rekenkracht, ook niet voor en loopt de navigatiesoftware naar behoren. Het beschikbare werkgeheugen in de SmartQ5 is wel ruim voldoende voor de software. Op filmpjes op het internet blijkt ook dat smartphones als de HTC Diamond II geen moeite hebben met het draaien van Navit. Opvallend is dat deze smartphones niet veel krachtigere hardware hebben dan de SmartQ5. Het probleem zou dan ook kunnen worden veroorzaakt door het besturingssysteem.

Wat betreft het besturingssysteem zijn er nog een aantal opties. Allereerst is er een versie van de firmware die de processor op 800MHz laat draaien in plaats van 667MHz. Ook zijn er andere besturingssystemen te vinden voor de SmartQ5. Dit zijn Android en MER, welke beide op Linux zijn gebaseerd. Als er een processor moet komen met meer rekenkracht, dan zal deze een betere instructieset moeten bevatten. De Cortex A8-core van ARM zet betere prestaties neer die in de buurt van de Intel Atom processor komen. Deze processor zet tevens 4 tot 5 maal zoveel prestaties neer ten opzichte van de ARM11-core die de SmartQ5 bevat.

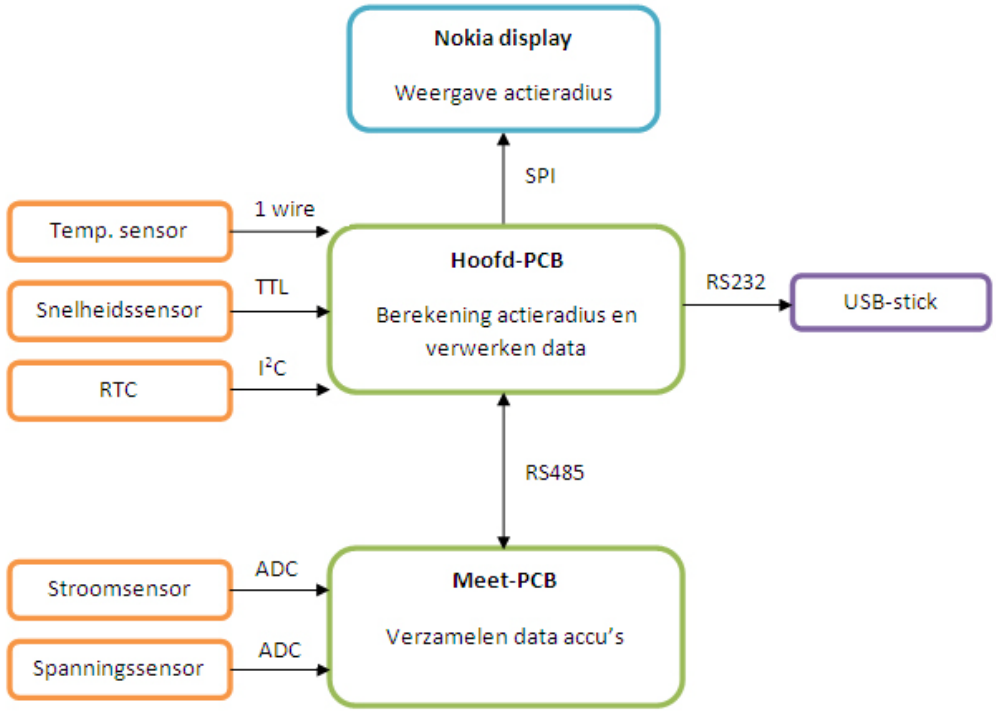
Figuur 12 GreeNav op de SmartQ5

7. Hardware

Voor het berekenen van de actieradius hebben Patrick de Vries en Stefan Huisman, onze voorgangers, een tweetal printplaaten ontworpen. Het systeem bevat een meet-unit en een hoofd-unit. Tijdens ons project willen we hier nog een extra unit aan toevoegen, namelijk het navigatiesysteem. Het doel is om de actieradius constant in beeld weer te geven. Als we de actieradius beschikbaar hebben in het navigatiesysteem kunnen we hierop andere functies baseren. Een voorbeeld hiervan is dat de gebruiker een melding krijgt als de actieradius te klein dreigt te raken.

7.1 Oude situatie

Hieronder staat een overzicht van de hardware zoals die gemaakt is door Patrick en Stefan.



Schema 3 De oude hardware situatie

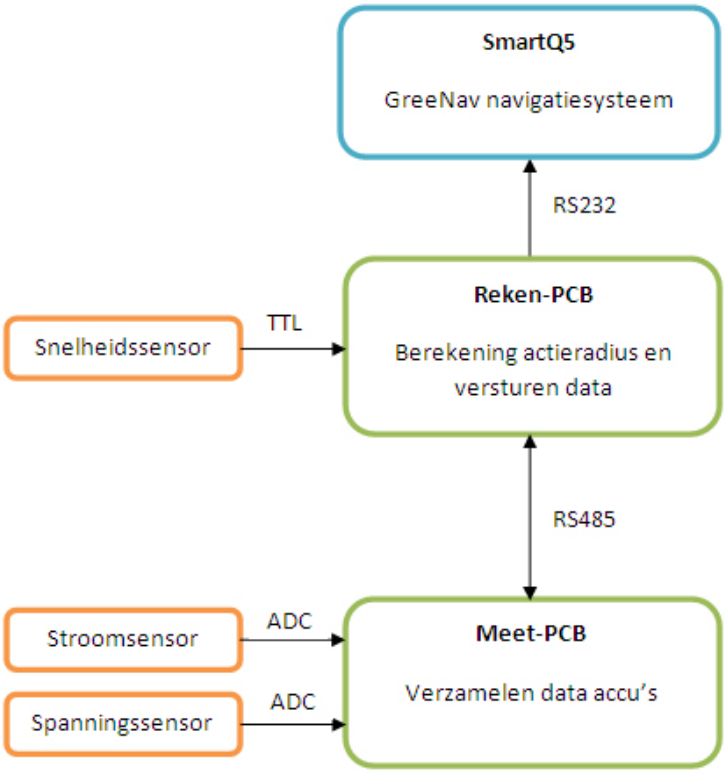
Bovenaan staat het Nokia-schermpje dat nu op de scooter gemonteerd is. Op het scherm staat de actieradius.

Het middelste gedeelte is de hoofd-PCB. Er komen drie sensoren binnen. Namelijk een temperatuursensor, een snelheidssensor en een realtime-clock. De temperatuursensor wordt momenteel niet gebruikt. De hoofd-PCB zorgt er ook voor dat de meetwaarden weggeschreven worden naar een USB-stick.

Het onderste gedeelte is de meet-PCB. Hier wordt de stroom en spanning gemeten om hiermee de actieradius te bepalen.

7.2 Nieuwe situatie

Er is een groot verschil tussen de oude en de nieuwe situatie. Dit is namelijk de SmartQ5. Door het gebruik van een soort mini-computer zijn er een aantal functies niet meer nodig. De realtime-clock bijvoorbeeld is overbodig geworden. De tijd kan ook uit de SmartQ5 gehaald worden. Het wegschrijven naar USB-stick is eveneens overbodig geworden. Tenslotte halen we de schakelaars, die niet in het schema staan, ook weg. Uiteindelijk moet dit de nieuwe situatie worden:



Schema 4 De nieuwe hardware situatie

De bovengenoemde veranderingen betekenen dat de hoofd-PCB een stuk simpeler zal worden. Eigenlijk is de enige functie nog het berekenen van de actieradius. Vanaf nu noemen we het dus ook liever de reken-unit.

7.3 RS232

Voor de communicatie van de reken-unit naar de SmartQ5 is gekozen voor RS232. Deze keuze hebben we in thema 14 gemaakt. Er waren voor ons twee andere opties. Namelijk CAN en RS485.

RS485

RS485 is de professionelere variant van RS232. Er zijn hogere snelheden over langere afstanden te halen. Ook is RS485 veel minder gevoelig voor storingen. Nog een groot verschil met RS232 is dat RS485 meerdere (maximaal 32) nodes kan hebben. Het zou dus mogelijk zijn om ook een PC aan te sluiten op de communicatielijnen.

Een nadeel van RS485 is dat het wat meer tijd kost om het te implementeren in een microcontroller. We hebben ook nog nooit met RS485 gewerkt, waardoor ervaring ontbreekt

CAN

Ook CAN zou nog een optie zijn. Eigenlijk zijn de verschillen met RS485 relatief klein. In ons geval valt CAN snel af door de kosten. Het opzetten van een CAN-netwerk met een PC daarin is een stuk duurder dan RS485.

RS232

Uiteindelijk hebben we dus voor RS232 gekozen. Het voordeel van RS232 is dat het heel gangbaar is voor communicatie van en naar een microcontroller. De SmartQ5 heeft een USB poort, dus we kunnen een USB-serieel converter gebruiken. Dit zorgt ervoor dat we tijd kunnen besparen ten opzichte van CAN en RS485.

De nadelen die RS232 heeft zijn bij ons eigenlijk niet echt van toepassing. Omdat de hardware en de SmartQ5 in het stuur zitten, is de USB kabel maar zeer kort waardoor de kans op storingen klein is. Ook het feit dat er geen motorkabels in de buurt zijn, zorgt voor reductie van storingen. Het feit dat RS232 maar geschikt is voor 2 nodes vinden we ook niet een groot issue. Voorlopig denken we niet dat er nog meerdere nodes nodig zijn. Het was leuk geweest dat de mogelijkheid er zou zijn, maar het is niet noodzakelijk.

7.4 De reken-unit

Figuur 13 De reken-unit

In bijlage 1 staat het schema dat we ontworpen hebben voor de reken-unit. Eigenlijk is er veel hetzelfde gebleven ten opzichte van het schema van Patrick en Stefan.

Het enige nieuwe is eigenlijk het gedeelte rond de FT232RL. Dit is een chip van FTDI die zorgt voor een USB-serieel conversie. Er is op deze manier alleen maar een USB kabel nodig om de hardware aan de computer, of in dit geval de SmartQ5, aan te sluiten.

De print hebben we laten maken bij Eurocircuits. Hierdoor weten we zeker dat de onderlinge verbinden zo optimaal mogelijk zijn. Dit komt ten goede aan de stabiliteit van het systeem. Qua componenten hebben we zoveel mogelijk voor SMD gekozen. Dit is over het algemeen wat goedkoper.

7.4.1 Het communicatieprotocol

Het communicatieprotocol is erg simpel opgezet. Er wordt door de reken-unit eens in de 5 seconden een string naar de SmartQ5 gestuurd. Deze string bevat twee gegevens. Namelijk de snelheid waarmee gereden wordt en de actieradius. De string die verzonden wordt kan er zo uit zien:

```
*;0203;0370#
```

De asterisk (*) geeft aan dat we een nieuwe string starten en het hekje geeft het einde aan. In GreeNav wordt gekeken of deze tekens op de goede plek staan. Als dit het geval is dan wordt de string verder uitgelezen en opgesplitst. Het eerste getal, in het voorbeeld 0203, staat voor de snelheid. het getal 0203 staat gelijk aan een snelheid van 20,3km/h. Na de “;” staat de actieradius. Het getal 0370 staat voor een actieradius van 37,0km. De puntkomma's worden door GreeNav gebruikt om de gegevens te kunnen scheiden.

8. Website

Om aan te geven waar de locaties zich bevinden moet er een systeem zijn wat gebruiksvriendelijk is. Je kan niet verwachten dat de gebruiker de coördinaten uit zijn hoofd kent. Het invoeren van een adres is in dat geval voor de gebruiker het gemakkelijkst. Hiervoor moet je echter aan 'geocoding' doen, het omzetten van een adres naar coördinaten.

Nog gemakkelijker is echter een kaart waarop je de bestemming kan aangeven. Helaas heeft Openstreetmap, die we ook voor het navigatiesysteem gebruiken, geen gemakkelijke optie. Er is echter nog een grote speler die bij velen bekend is: Google Maps. De functies die we met de gratis versie van Google Maps kunnen gebruiken zijn ruim voldoende.

Figuur 14 www.greenav.nl

Het maken van een gebruiksvriendelijke website is eigenlijk meer iets voor een student van Human Technology. Wij hebben echter geen tijd gehad om een student om raad te vragen. Dit zou in de toekomst zeker aan te bevelen zijn. Tijdens het maken van de website hebben we wel zoveel mogelijk rekening gehouden met de gebruiksvriendelijkheid. Een pagina van de website is in figuur 15 te zien.

8.1 Google maps

Google Maps is enkel geschikt voor online gebruik. Een optie om de kaarten samen te laten werken met een applicatie is niet zomaar klaar. Dit bracht ons op het idee om een website te ontwikkelen voor het beheer. Het grote voordeel daarvan is ook dat de data tussen verschillende gebruikers kan worden gedeeld. Door de aanwezige kennis over websites was het ook geen probleem om de beheerapplicatie in een website te bouwen.

8.2 Programmeertaal en database

De website voor GreeNav kent meerdere programmeertalen. Twee programmeertalen zorgen voor de inhoud van de website. Dit zijn PHP en Javascript. Verder zijn er ook nog twee zogenaamde 'markup languages'. Dit zijn HTML en CSS. Deze twee zorgen voor de opmaak van de website. Daarnaast hebben we nog een MySQL-database. Daarin wordt alle data opgeslagen. Hieronder wat meer informatie over de verschillende talen.

8.2.1 PHP (versie 5)

PHP Hypertext Preprocessor is één van de meest gebruikte programmeertalen voor websites. De taal lijkt sprekend op C, waardoor deze snel is aan te leren. Verder is er veel documentatie terug te vinden over de taal op het internet. Daarnaast bieden de meeste webhostingpakketten PHP-ondersteuning aan. De code wordt op de server zelf gecompileerd en uitgevoerd.

8.2.2 Javascript

Javascript is een taal die op de computer zelf wordt uitgevoerd. Deze taal is nodig voor de aansturing van Google Maps. Dat de code op de computer van de gebruiker wordt uitgevoerd geeft ook een voordeel. Op deze manier kan een pagina namelijk worden gewijzigd zonder dat de pagina opnieuw moet worden geladen. Hierdoor kunnen bepaalde functies sneller worden uitgevoerd. Denk hierbij aan het veranderen van de volgorde van een route.

8.2.3 HTML

De HTML markup language is wellicht het meest bekend. Deze taal wordt doorgaans gebruikt voor het maken van een statische website. Hierin kunnen onder andere tabellen worden gemaakt. Door HTML samen met PHP te gebruiken kan een dynamische website worden gemaakt. De PHP-code wordt dan eerst uitgevoerd, waarna de uitkomsten van dat script binnen HTML-code worden geplaatst. Dit is een statische pagina die wordt gegenereerd aan de hand van de invoer en data in de database. Deze pagina kan de browser vervolgens laten zien.

8.2.4 CSS

Een Cascading Style Sheet is een configuratiebestand dat zorgt voor de opmaak van de website. Hierin kan bijvoorbeeld worden aangegeven wat de kleur, afstand en het lettertype moet zijn van een bepaald HTML-element. Deze wordt op elke pagina over genomen. Hierdoor ziet de hele website er op iedere pagina hetzelfde uit. Het heeft ook als voordeel dat het uiterlijk gemakkelijk en snel te veranderen is.

8.2.5 MySQL

Net als PHP is ook MySQL een gratis product dat door veel webhosters wordt aangeboden. Daardoor is voor deze database veel documentatie te vinden. MySQL is niet de meest uitgebreide optie en kent in de geavanceerde functies ook nog wel eens bugs. Wij hebben echter de basis nodig die probleemloos werkt. Voor ons is de documentatie en ondersteuning bij webhosters dan ook belangrijker. De database wordt gebruikt om alle data op te slaan die de gebruikers genereren.

8.3 Framework

Om de achterliggende code van de website overzichtelijk te houden is een framework opgezet. Binnen dit framework zijn de meeste onderdelen compleet modulair opgezet. Dit houdt in dat nieuwe componenten gemakkelijk kunnen worden toegevoegd aan de website. Dat maakt het tevens ook gemakkelijk om onderdelen tegelijkertijd te ontwikkelen. Hieronder beschrijven we de onderdelen die terug te vinden zijn in het framework. In de bijlagen is uitgebreidere documentatie terug te vinden van het framework.

8.3.1 Alt

Binnen de Alt-map (alternatief) komen de scripts die niets te maken hebben met de website zelf. Dit zijn voornamelijk onderdelen die door andere applicaties kunnen worden gebruikt. Een bekend voorbeeld is een RSS-feed, waarmee je automatisch updates krijg van bijvoorbeeld nieuwsberichten. Andere opties zijn o.a. een sitemap in XML-formaat, welke door onder andere Google kan worden gebruikt om de website te indexeren.

8.3.2 Config

Deze map bevat de configuratie van de website. De centrale configuratie zorgt er voor dat instellingen gemakkelijk te wijzigen zijn. Hieronder vallen onder andere het menu en de naam van de website. Belangrijker is echter de serverconfiguratie. Hierdoor kunnen we de website lokaal testen met de instellingen die hiervoor gelden. Voor de webhosting hoeven alleen een aantal instellingen gewijzigd te worden.

8.3.3 Core

Core bevat de basis van de website. Dit essentiële deel wordt na de configuratie als eerste ingeladen. De core zorgt onder andere voor het maken van verbinding met de database. Verder wordt de juiste template gekozen. Ook zullen de library, scripts en de juiste module worden ingeladen. De core kan het best worden vergeleken met een initialisatie en de main-loop in C.

8.3.4 Extensions

Hierin zijn de extensies te vinden. Dit zijn componenten die los worden ingeladen en vanuit een module of template kunnen worden aangeroepen. Het zijn doorgaans kleine en simpele bouwstenen voor de website. Voorbeelden van extensies zijn een loginbox of hoofdmenu.

8.3.5 Library

De library bevat allemaal functies. Deze bibliotheek kan vanuit de hele code worden gebruikt en maakt het gemakkelijk om functies te hergebruiken die vaker voor komen. De library bevat zo onder andere functies om informatie over de gebruiker op te vragen. Ook zijn er functies voor onder andere Google Maps, de MySQL database en afbeeldingen terug te vinden.

8.3.6 Media

In de map media zijn onder ander afbeeldingen en video's terug te vinden. Ook is er een standaard pakket van icoontjes en smiley's. De map kent verder geen scripts en wordt niet geladen.

8.3.7 Modules

In de map modules komen de echte pagina's te staan. Er kan slechts één module worden geladen per pagina. Deze bevat dan ook de zogenaamde 'content' van een pagina. Voorbeelden hiervan zijn onder andere het overzicht van locaties of het laten zien van een nieuwsbericht.

8.3.8 Scripts

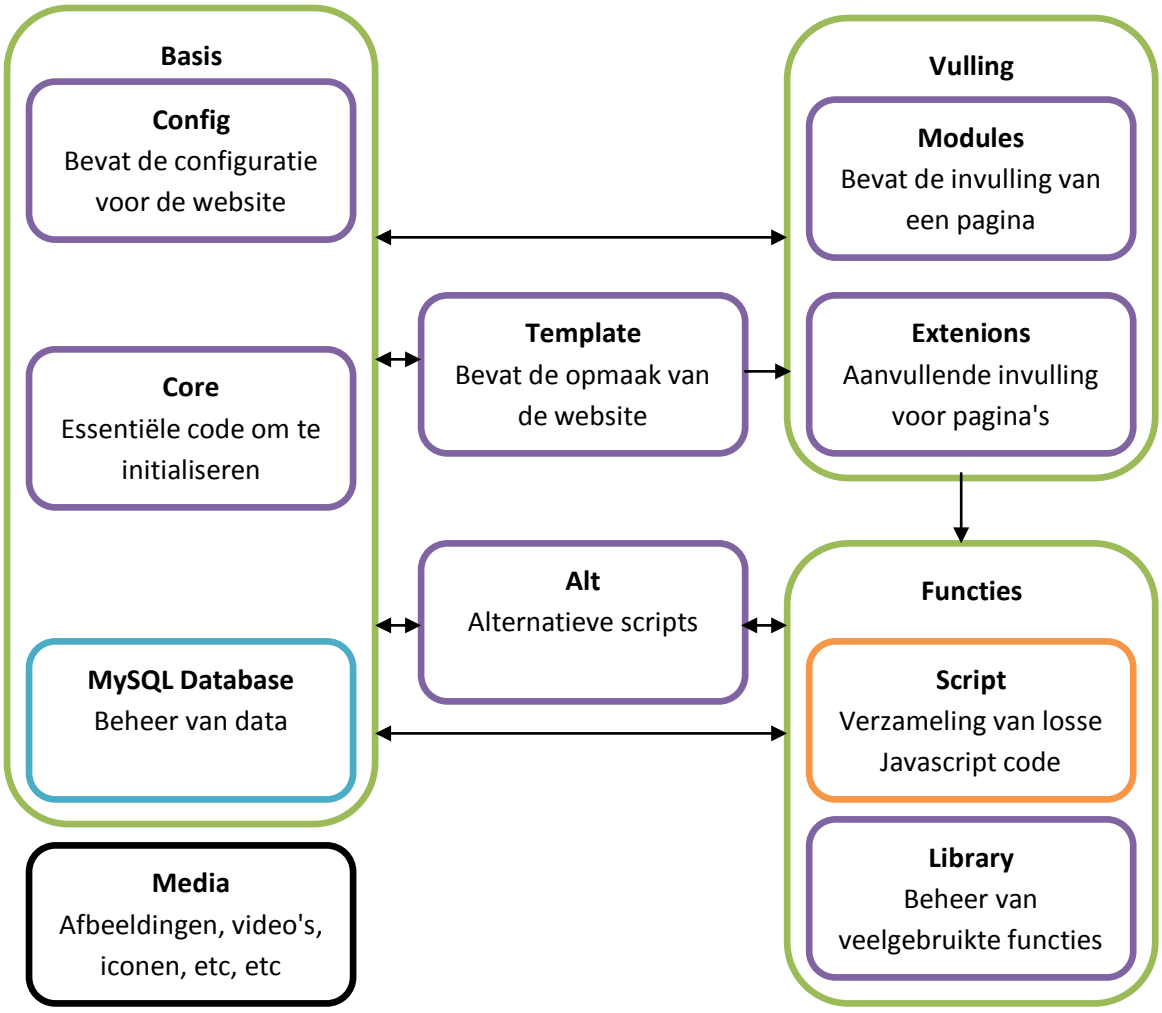
Hierin is alle Javascript terug te vinden. Bij het laden van de pagina zal een stuk PHP-code zorgen dat de links naar de Javascript bestanden zal plaatsvinden binnen de head-tag van de HTML-code. Deze map bevat onder andere het script om de selectie te wijzigen. Ook standaard scripts van het internet, zoals Lytebox voor het weergeven van grote afbeeldingen zijn aanwezig.

8.3.9 Template

Hierin komt het template van de website. Dit is het geraamte waarin de website zal worden weergegeven. Deze zorgt dan ook voor het uiterlijk van de website zelf. Binnen een template vinden we veel HTML-code voor de indeling. Daarbinnen worden op de juiste plaatsten de module en extensies aangeroepen. Ook vinden we hier de CSS-bestanden terug.

8.3.10 Globaal overzicht

Hieronder is een schematisch overzicht dat de diverse onderdelen en verhoudingen weergeeft.



Schema 5 Schematisch overzicht van de opbouw van de website

Stap 1: Hierin wordt dus eerst de basis geladen met behulp van de core. Deze legt de verbinding met de database en maakt gebruik van de configuratie. Vervolgens zal deze de library en scripts toevoegen. Ook zullen enkele functies uit de library worden aangeroepen.

Stap 2: Daarna zal worden gekeken wat exact geladen moet worden. Als dat geen reguliere pagina is zal het juiste onderdeel van alt worden aangeroepen. In het andere geval zullen de modules en extensions worden geladen. Na deze initialisatie zal het juiste template worden geladen.

Stap 3 (alternatief): Wanneer een alternatief onderdeel wordt geladen zal het juiste script worden uitgevoerd. Deze heeft toegang tot de library en de scripts.

Stap 3 (template): In het geval dat een pagina wordt geladen zal eerst het template worden geladen. Vanuit dit template zal de juiste module worden geladen. Datzelfde geldt ook voor de extensions die zijn ingesteld. De modules en extensions maken gebruik van zowel de configuratie als de functies die beschikbaar zijn binnen de library en scripts.

8.4 Ontwerp

Voor de eindgebruiker is het ontwerp ook belangrijk. De website moet niet alleen vlot werken, maar vooral ook gemakkelijk en duidelijk zijn. Daarnaast moet de website mensen ook aantrekken. Een fris design is bepalend voor de eerste indruk die de gebruiker van de website krijgt. Hieronder valt te lezen wat we hebben gedaan om de website zo goed mogelijk op te zetten.

8.4.1 Compatibiliteit

Belangrijk is dat de website overal goed weer te geven is. Niet alle browsers laten iedere website hetzelfde zien. Daarom hebben we de website getest met de meest gebruikte browsers. Deze zijn:

- Internet Explorer 8
- Mozilla Firefox 3.5.x en 3.6.x
- Google Chrome 4.x
- Opera 10.x
- Safari 4

Ook blijkt de website goed te werken op smartphones. De pagina's laden vlot op het mobiele internet. Wel doet Google Maps er een tijdje over voor de kaart wordt weergegeven. De geteste smartphone betreft een Sony Ericsson Xperia X1 met Windows Mobile 6.5.3 en Opera Mobile 10 als browser.

Ondersteuning voor Internet Explorer 6 is niet aanwezig. De website draait wel met deze browser, maar geeft fouten in de opmaak. Internet Explorer 6 heeft nog altijd een aanmerkelijk marktaandeel, maar opvolgers zijn al lang beschikbaar. Het verhelpen van de problemen zou veel tijd kosten. Wij volgen dan ook het voorbeeld van grote websites zoals Tweakers.net en Engadget.com die onlangs de ondersteuning voor Internet Explorer 6 lieten vallen.

8.4.2 Ontwerp van de website

De basis van het ontwerp is zeer gebruikelijk. De banner en het menu zijn bovenaan terug te vinden, zoals te zien is in figuur 16. Deze nemen in de hoogte niet te veel ruimte in. Daarna komt de pagina zelf. Deze is in twee delen opgedeeld met de daadwerkelijke pagina aan de rechterzijde. Aan de linkerzijde is ruimte om andere informatie en opties te bieden. Dit zijn onder andere een loginbox en een overzicht van de gemaakte selectie van bestemmingen en routes.

Figuur 15 De website

Door deze relatief simpele lay-out in acht te nemen is de website overzichtelijker. Doordat veel websites deze indeling hanteren is de structuur ook meteen duidelijk bij de meeste gebruikers. Daardoor zullen veel gebruikers de website sneller begrijpen waardoor deze ook door een breed publiek te gebruiken is.

De breedte van de hele website bedraagt 1000 pixels. Deze breedte is gekozen om ook netbookgebruikers in staat te stellen de website gemakkelijk te gebruiken. De resolutie van veel netbooks is namelijk 1024 pixels breed. Daarmee is bij een fullscreen browser nog genoeg ruimte om de scrollbalk weer te geven. Hierdoor hoeft de gebruiker niet horizontaal over de pagina te scrollen, iets dat niet gebruiksvriendelijk is.

8.4.3 Iconen

Om de website nog verder te verduidelijken is gebruik gemaakt van iconen. De iconenset van famfamfam.com is gratis te gebruiken. Deze bevat zo'n 1000 iconen waardoor voor elke knop of informatie wel een mooi icoontje te vinden is. De hele set is terug te vinden in de 'media'-map van het framework.

8.5 Google Maps

De belangrijkste reden dat we voor het gebruik van een website gekozen hebben, is dat we dan gemakkelijk gebruik kunnen maken van Google Maps. Hiermee zouden we een eenvoudige manier om coördinaten te verkrijgen kunnen aanbieden. Daarnaast zou een kaart ook mogelijkheden geven om de gebruiker meer informatie over de locatie aan te bieden. De keuze voor Google Maps was vlot gemaakt. Het heeft als voordelen dat het gratis te gebruiken is, veel functies en mogelijkheden bevat en ook nog eens een goede documentatie kent.

8.5.1 Testen met Google Maps

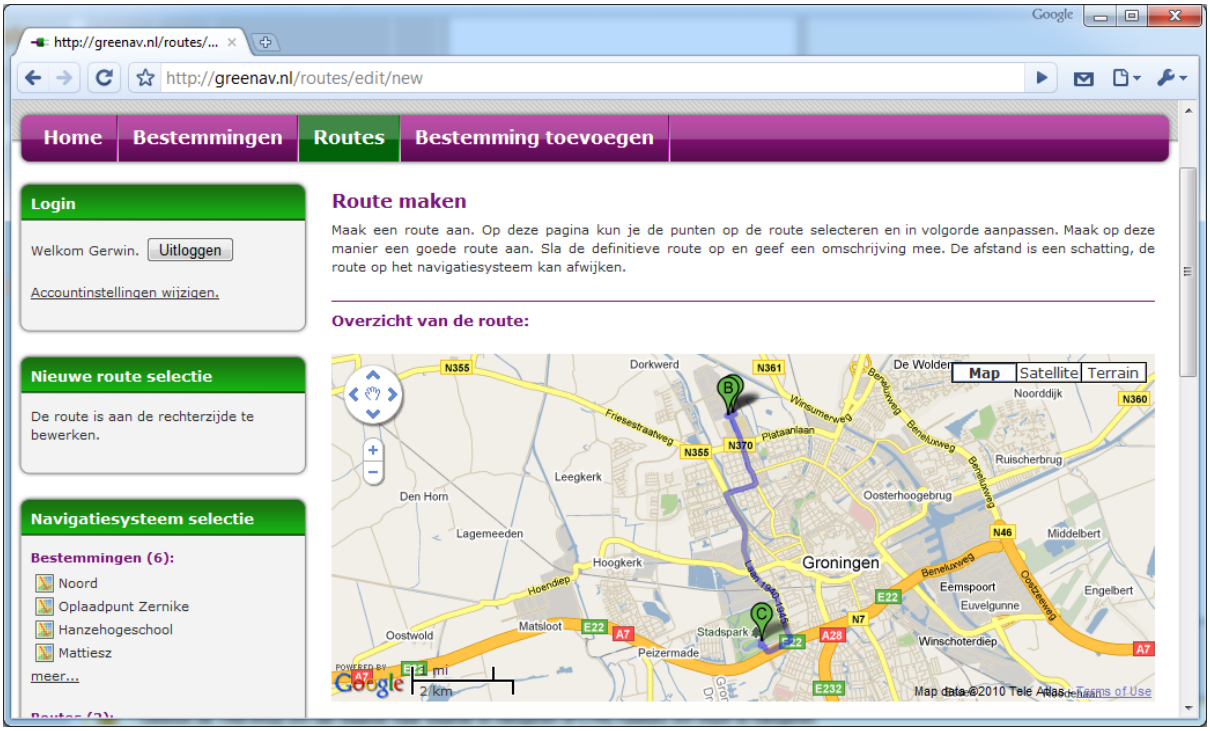
Voor we daadwerkelijk over gingen tot het bouwen van een website wilden we de werking van Google Maps uitzoeken. Hiervoor hebben we gebruik gemaakt van de diverse voorbeelden die aangeboden worden. Deze worden aangeboden op http://code.google.com/apis/maps/. Met de voorbeelden hebben we gekeken of we de gestelde eisen konden implementeren. Deze eisen waren als volgt:

- Locatie aanwijzen
- Coördinaten ophalen
- Adres zoeken
- Route weergeven

De documentatie bleek zeer uitgebreid te zijn. De meeste onderdelen konden we zo overnemen uit de voorbeelden. Andere eisen, zoals het wegschrijven van de coördinaten, konden we ook eenvoudig afleiden uit de code. Deze afzonderlijke delen waren daarna redelijk eenvoudig samen te voegen. De snelle ontwikkeling hiervan gaf ons genoeg vertrouwen om Google Maps te gebruiken binnen de website. Voor het weergeven van een route is echter wel een limiet van 25 punten ingesteld door Google.

8.5.2 Google Maps implementeren

Voor de website is de testcode opnieuw ingedeeld. Daarbij is deze opgedeeld naar de functies die op de website worden gebruikt. Dit zijn het aangeven van een statische locatie, het aanwijzen van een locatie, het zoeken op adres en het weergeven van een route. Figuur 17 laat de integratie met Google Maps zien.



Figuur 16 Het maken van een route op www.greenav.nl

Daarbij is ook integratie met de pagina gemaakt. Zo wordt ook de lengte van een route weergegeven en worden de coördinaten opgevangen in het informatieformulier. Tevens worden ook de adresgegevens vanuit Google Maps weggeschreven naar de juiste velden binnen het invulformulier.

Om de Javascript-code gemakkelijk te laden in de pagina zijn er een aantal functies in PHP geschreven. Zo kan de complete functionaliteit met twee of drie functieaanroepen worden geladen op de juiste plaats op de pagina.

8.6 Mogelijkheden

De uiteindelijke website bevat alle mogelijkheden die in eerste instantie nodig zijn voor het beheer. Zo kunnen bestemmingen worden aangemaakt, routes worden ingesteld en kan de inhoud van het navigatiesysteem worden beheerd.

8.6.1 Registratie

Om de selectie van de gebruiker te kunnen bewaren, moet deze een account hebben op de website. Op die manier kunnen we de data aan een bepaalde gebruiker koppelen. Hiervoor hebben we dan ook een loginsysteem ontwikkeld. Ook kan de gebruiker zijn gegevens wijzigen en het wachtwoord opvragen.

Het opslaan van de selectie die een gebruiker maakt is niet de enige reden om een loginsysteem te maken. Het zorgt er ook voor dat niet iedereen zomaar punten kan toevoegen aan het systeem. Hiermee tackelen we het probleem dat bots zouden kunnen gaan spammen door allemaal punten aan te maken. Ook geeft het de optie om eigen locaties privé te houden.

8.6.2 Overzichten

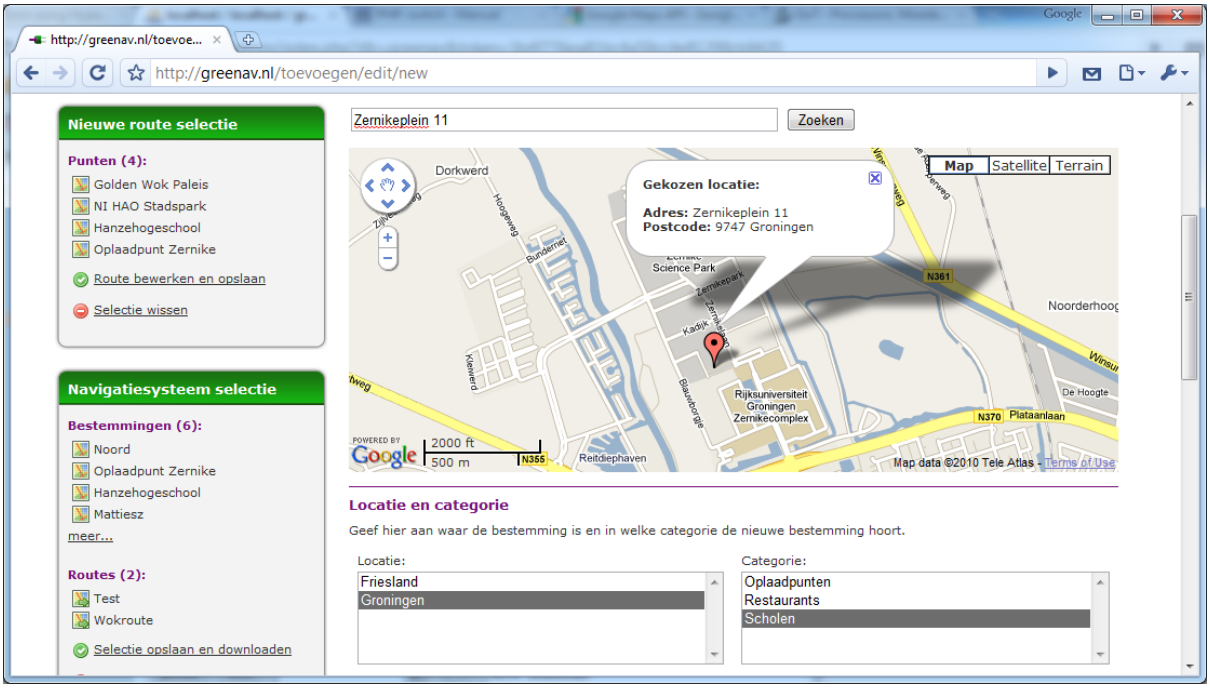
Het overzicht van alle bestemmingen en routes komt overeen met die van het navigatiesysteem. Zo bevatten de bestemmingen weer de indeling van locatie, categorie en dan bestemming. Hierbij worden dezelfde logo's gebruikt als in het navigatiesysteem. Ook zijn de routes allemaal met logo terug te vinden. In figuur 18 is te zien hoe de categorie overzichtspagina eruit ziet.

Figuur 17 Overzicht van categorieën op www.greenav.nl

Verder is er voor zowel een bestemming als route een overzichtspagina. Bij de bestemmingen is een afbeelding terug te vinden en wordt de locatie op de kaart weergegeven. Tevens wordt ook alle informatie weergegeven over de bestemming. Bij de route is de pagina vrijwel gelijk, alleen staan nu alle punten op de kaart aangegeven.

8.6.3 Bestemming aanmaken

De eindgebruiker kan middels de website nieuwe bestemmingen toevoegen. Hiervoor moet wel ingelogd worden op de website. Het aanmaken van een bestemming is opgedeeld in een paar onderdelen. Zo kan met behulp van Google Maps de locatie worden aangeklikt op de kaart, zoals in figuur 19 is afgebeeld. Verder moeten een locatie en categorie worden geselecteerd. Daarna kan informatie over de bestemming, zoals de naam en een omschrijving, worden ingevuld. Ook kunnen een logo en afbeelding worden toegevoegd.



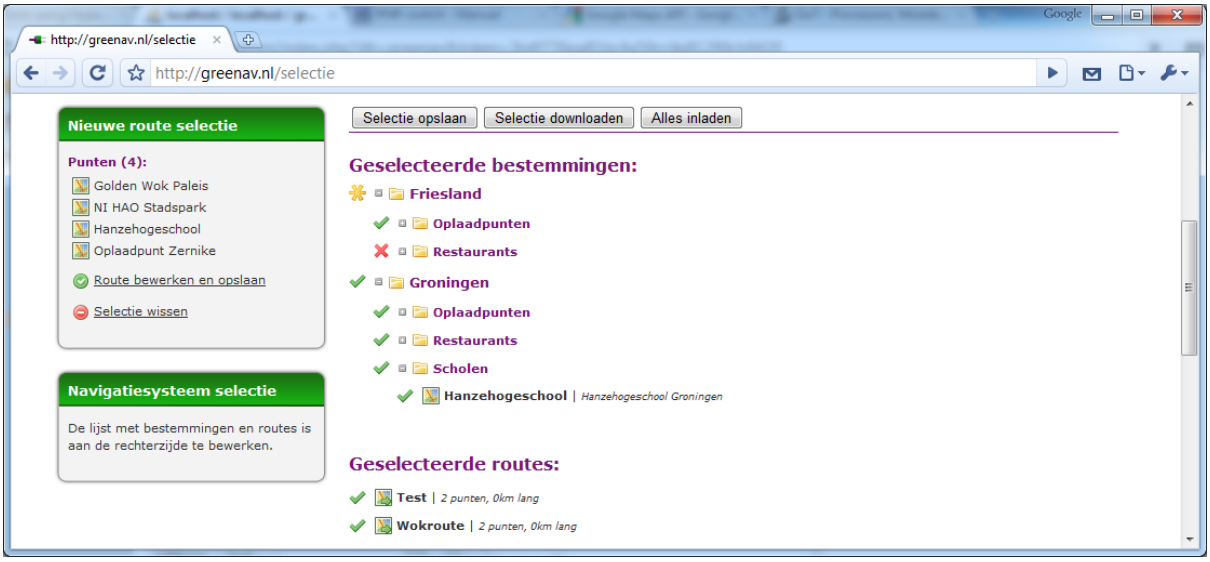
Figuur 18 Het aanmaken van een bestemming op www.greenav.nl

8.6.4 Route aanmaken

Het aanmaken van een route gaat doormiddel van het toevoegen van bestemmingen aan de selectie van de nieuwe route. Nadat de gewenste bestemmingen zijn toegevoegd, kan de gebruiker de nieuwe route maken. Met behulp van Google Maps wordt de route weergegeven, terwijl deze kan worden aangepast door de volgorde van de punten te veranderen. Daardoor kan de route naar eigen wens worden aangemaakt. Net als bij bestemmingen is het ook hier mogelijk om een logo en afbeelding aan te geven. Tevens zijn de velden voor een naam en omschrijving aanwezig.

8.6.5 Selectie voor navigatiesysteem maken

Om een eigen selectie van punten voor het navigatiesysteem op te stellen kan de gebruiker bestemmingen en routes selecteren. Dit is vergelijkbaar met een webshop: Je selecteert de bestemmingen en routes die je wil downloaden. Vervolgens kun je deze selectie downloaden. Ook is het mogelijk om alle data te importeren en hierin een selectie te maken. Hiervoor wordt gebruik gemaakt van een mappenstructuur. Figuur 20 toont deze mappenstructuur met bestemmingen. De gedownloade configuratie kan worden ingelezen door de downloadtool, waarop deze alle informatie zal downloaden en op de SD-kaart zal wegschrijven.



Figuur 19 Het maken van een selectie voor het navigatiesysteem op www.greenav.nl

9. Downloadtool

Om de data uiteindelijk op het navigatiesysteem te krijgen is een downloadtool geschreven. Deze maakt het mogelijk om eenvoudig de gemaakte selectie van het internet te downloaden en op de SD-kaart te schrijven. Ook kan gekozen worden om de kaart van een update te voorzien. De downloadtool is daarbij zo eenvoudig mogelijk gemaakt. Een student van Human Technology zou hier nog eens naar kunnen kijken.

9.1 Configuratie

Bij installatie zal het programma de eerste keer een initialisatie uitvoeren. Hierbij wordt alles geconfigureerd. Ook zal de tool de eerste keer een nieuwe kaart downloaden van de Benelux. Daarna is het programma klaar om gebruikt te worden.

Het downloaden van de configuratie begint op de website van GreeNav. Na het maken van de selectie wordt een GCF-bestand (GreeNav Config File) gedownload. Bij het openen van dit bestand wordt het programma automatisch gestart. Het GCF-bestand wordt op de website gegenereerd en bevat de XML-code voor alle punten en routes. De te downloaden bestanden worden toegevoegd als commentaar.

In dit commentaar bevindt zich het nummer van de afbeelding, welke terug te vinden is met behulp van de database op de website. Ook staat daarin aangegeven wat de locatie moet worden op de SD-kaart. De code ziet er als volgt uit:

```
<!-- IMG_DOWNLOAD=22# IMG_FILE=img/loc_0/cat_0/thumb_0.jpg# -->
```

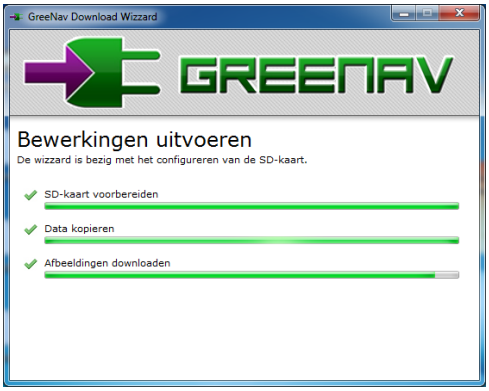
In dit voorbeeld staat het getal 22 voor de afbeelding. Deze afbeelding is terug te vinden als www.greenav.nl/img/22/22.jpeg op de server. Dit bestand wordt vervolgens gedownload naar x:\Greenav\img\loc_0\cat_0\thumb_0.jpg, waarin x: staat voor de locatie van de SD-kaart.

9.2 Werking van de Downloadtool

Wanneer het GCF-bestand wordt geopend, zal de downloadtool automatisch worden gestart. Daarna zullen 3 stappen moeten worden doorlopen. In de eerste stap, afgebeeld in figuur 21, zal de locatie van de SD-kaart moeten worden aangegeven. Deze wordt automatisch gedetecteerd als de SD-kaart aanwezig is, maar de optie om een andere locatie aan te wijzen blijft aanwezig. Hetzelfde geldt voor het configuratie. Automatisch wordt het bestand waarmee het programma wordt gestart gekozen, maar een ander bestand kiezen is ook mogelijk. In de laatste stap kunnen nog configuratieopties worden aangepast. In de huidige software kan daar enkel worden aangegeven of een nieuwe kaart moet worden gedownload.

Figuur 20 Het eerste scherm van de downloadtool

Daarna zal de downloadtool de taken uitvoeren. In eerste instantie zal de tool de SD-kaart opruimen. Daarna zullen de configuratie en de kaart naar de SD-kaart worden gekopieerd. Eventueel wordt eerst nog een nieuwe kaart van het internet gedownload. Als laatste zullen alle afbeeldingen van www.greenav.nl worden gedownload. Bij dit hele proces is duidelijk te zien met welke taak de software bezig is. In figuur 22 wordt dit proces weergegeven.



Figuur 22 Het beschrijven van de SD-kaart

9.3 Uitbreidbaarheid

De software zelf is tevens gemakkelijk te gebruiken voor andere methoden om data over te sturen. Zo kan ook een USB-stick worden gebruikt. Ook zou het mogelijk zijn om met een ander apparaat direct via een USB-kabel te communiceren wanneer deze als schijf wordt gezien.

10. Conclusie

Tijdens het afstuderen zijn alle deelonderdelen van het navigatiesysteem in stappen ontwikkeld. Deze manier van werken blijkt zeer effectief te zijn. Door de focus op kleine deelstappen te leggen is het overzichtelijker om tot een goede oplossing te komen. Daarmee wordt de implementatie van de functionaliteit een stuk eenvoudiger.

Met het GreeNav navigatiesysteem zijn we er in geslaagd om een integratie te maken met de gegevens uit de accu en een centrale database met oplaadpunten. Het navigeren aan de hand van afbeeldingen is daarbij een mooie methode om onbekenden op een visuele manier naar de eindbestemming te navigeren. Daarmee zou het systeem kunnen worden ingezet om toeristen een rondleiding te geven en tegelijkertijd het duurzame vervoer te promoten.

Daarmee is gevraagde functionaliteit voor de navigatiesoftware gerealiseerd. Helaas geldt dat niet voor het navigatiesysteem op de gebruikte hardware. Door de opzet van de navigatiesoftware is deze echter wel te gebruiken op andere hardware. Verder zien we nog genoeg ruimte voor verbetering en uitbreiding van de functionaliteit. Deze verbeterpunten geven we in de aanbevingen aan.

Het complete navigatiesysteem biedt daarmee potentie om het gebruik van duurzame vervoersmiddelen te vergroten. Doordat de gebruiker tijdig wordt gewaarschuwd als de accu's leeg dreigen te raken, wordt het gebruikersgemak verhoogd. Er is veel ruimte voor verdere uitbreidingen in de toekomst op de basis die nu is gelegd. De toekomst is duurzaam en met dit systeem kunnen we een steentje bijdragen in die verduurzaming door mensen de voordelen te laten ervaren en de twijfel weg te nemen.

11. Aanbevelingen

Tijdens het werken aan een groot project komen altijd ideeën naar boven om het eindresultaat te verbeteren. Helaas is er een limiet aan de tijd die in een afstudeertraject kan worden gestoken. Niet al deze ideeën kunnen daardoor worden gerealiseerd. In dit hoofdstuk staan dan ook enkele ideeën en verbeteringen die kunnen worden doorgevoerd door een volgende lichting studenten.

Allereerst blijkt dat er betere hardware nodig is om de navigatiesoftware soepel te kunnen draaien. Voor de daadwerkelijke integratie in de scooter zal dan ook goed moeten worden gekeken naar alternatieve hardware met genoeg rekenkracht. Een punt van aandacht daarbij is om te kiezen voor een goed scherm dat ook geschikt is om in de zon af te lezen. Uiteindelijk zou die hardware in de scooter kunnen worden weggewerkt.

Het tweede stuk hardware, de reken-unit, kan ook worden verbeterd. Met een beter inzicht in de accu's, door middel van metingen, kan een betere formule worden opgesteld. Ook kunnen factoren als slijtage worden meegenomen. Daarvoor zou onderzoek kunnen worden gedaan naar een soort kalibratiemethode om het daadwerkelijke vermogen van de accu te berekenen. Daarbij zouden de cijfers van de verbruikte energie kunnen worden vergeleken met de energie die tijdens het laden door de accu wordt opgenomen.

Om meer mensen gebruik te laten maken van het systeem zou de software naar smartphones kunnen worden overgezet. Navit heeft al ondersteuning voor Windows Mobile, Android en de iPhone. Met kleine aanpassingen kan de software op deze telefoons draaien. Dat zou de drempel om het systeem te gebruiken verlagen doordat de aanschaf van apparatuur niet meer nodig is. In dat geval is het ook mogelijk om toepassingen te bedenken met het mobiele internet. Zo zou eventueel informatie via Wikipedia kunnen worden geladen bij de bestemmingen. Ook zou de dichtstbijzijnde, vrije, paal via het internet kunnen worden gezocht.

Wat betreft de gebruiksvriendelijkheid is het verstandig dat studenten van Human Technology hier nog eens kritisch naar kijken. Op een aantal punten zal het één en ander te verbeteren en te verduidelijken zijn. Door de nodige veranderingen door te voeren op onduidelijke punten, wordt het systeem een stuk beter inzetbaar bij een groot publiek.

Ook zien we ruimte voor een heel businessplan om het duurzame mobiliteit te promoten met het systeem. Steeds meer oplaadpalen worden gesponsord door bedrijven die een groen imago willen verkrijgen. Datzelfde geldt voor bijvoorbeeld restaurants die een oplaadpunt voor klanten openen. Door deze, tegen vergoeding, op te nemen in de afbeeldingen van oplaadpalen en in de routes ontstaat verdere promotie van deze bedrijven. Een student van Technische Bedrijfskunde zou kunnen kijken naar de haalbaarheid van zo'n constructie.

Er valt op tal van punten nog genoeg te verbeteren en uit te breiden. Studenten uit diverse opleidingen kunnen samen aan het project verder werken. De opleidingen Elektrotechniek, Human Technology, Werktuigbouwkunde, Informatica en Technische Bedrijfskunde kunnen nog genoeg toevoegen aan het project.

Literatuurlijst

Literatuur
Stefan Huisman en Patrick de Vries, Elektrische scooter onderzoek, Groningen, 2009

Internet	
duurzaamstestad.groningen.nl	Website over de energie doelen van de stad Groningen
http://www.hanze.nl/home/Onderzoek/Kennisportal/Kenniscentra/Energie+Kennis+Centrum/	Website van het Energiekenniscentrum
www.navit-project.org	Website van het Navit project
wiki.navit-project.org	Wiki van het Navit project vol met informatie
www.openstreetmap.org	Opensource kaarten die gebruikt worde in GreeNav
code.google.com/apis/maps	Informatie over de Google Maps API
www.w3schools.com	Tutorials en informatie over het bouwen van een website
www.php.net	Uitleg van functies in PHP
delphi.about.com	Website met voorbeelden van Delphi code
www.zipcodeworld.com	Voorbeeldcode in C
www.jiongtang.com	Website met drivers voor seriële communicatie
en.smartdevices.com.cn	Fabrikant van de SmartQ5
www.debian.org	Website met informatie over de Linux variant die gebruikt is
www.ubuntu.org	Linux distributie die op de laptops is gebruikt

Bijlagen

Bijlage 1 Hardwareschema

Bijlage 2 Installatiehandleiding

Bijlage 3 GreeNav XML-configuratie

Bijlage 4 SD-kaart indeling

Bijlage 5 Documentatie aangepaste Navit-code

Bijlage 6 Menustructuur

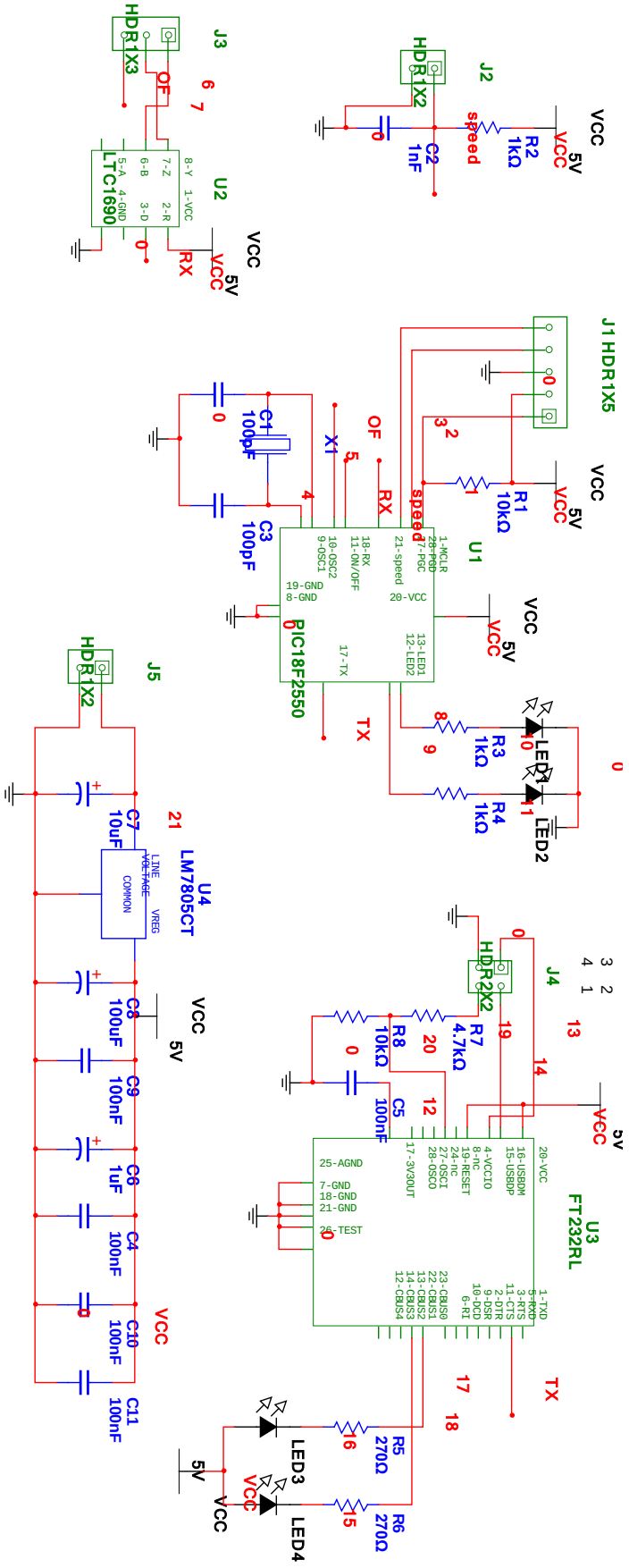
Bijlage 7 GreeNav compileren

Bijlage 8 Framework website

Bijlage 9 Databaseontwerp

Bijlage 10 DVD

Bijlage 1 Hardwareschema



Bijlage 2 Installatiehandleiding

Installatie ontwikkeltools

De dingen die we gebruikt hebben:

- Ubuntu x86 9.04
- Ubuntu updates installeren
- Anjuta (via Installeren/Verwijderen in Ubuntu)
- Build-essential

Als OS is gekozen voor Ubuntu, dat lijkt op het OS van de SmartQ5 (Debian based). Na installatie via de update manager gewoon alle updates binnenhalen.

Installatie Navit en GreeNav

Open een commandoprompt en voer de volgende regel uit:

---

sudo apt-get install build-essential pkg-config automake libglib2.0-dev libtool libxmu-dev libfribidi-dev gettext zlib1g-dev cvs gpsd gpsd-clients libgps-dev libdbus-glib-1-dev libgtk2.0-dev freeglut3-dev glutg3-dev libcegui-mk2-dev libdevil-dev libglc-dev libpcre3-dev libmng-dev libfreeimage-dev

---

Wiki met informatie over de installatie: http://wiki.navit-project.org/index.php/Interactive_help.

Na installatie van de benodigde pakketten in de commandoprompt kunnen we Navit installeren. Gewoon de *.deb installer openen en installeren.

Installatie van de kaarten

Download een Openstreetmap van http://wiki.navit-project.org/index.php/OpenStreetMaps.

Zet de map vervolgens in een map op je harde schijf. Bijvoorbeeld in:
“/home/gebruiker/Documenten/Scooter/Map/”

Daarna zal het configuratiebestand moeten worden ingesteld zoals op de wiki staat aangegeven. Het configuratiebestand is, na installatie van Navit, te vinden in de map van Navit.

Kopieer navit.xml naar “~/.navit”, bijvoorbeeld “/home/gebruiker/.navit”

Open deze navit.xml met een tekst verwerker en pas deze aan om de kaarten toe te voegen in de configuratie.

Zoek nu het volgende deel en verander de maplocatie:

```
</mapset>
<!-- Mapset template for openstreetmaps -->
<mapset enabled="no">
<map type="binfile" enabled="yes" data="/media/mmc2/MapsNavit/osm_europe.bin"/>
</mapset>
```

Hierin moet de de locatie worden aangeven waar de map te vinden is. In ons voorbeeld is dat dan dus:

```
<mapset enabled="yes">
<map type="binfile" enabled="yes" data="/home/gebruiker/Documenten/Scooter/Map/osm_bbox_2.1,48.9,7.8,54.5.bin" />
```

Let ook op de regel die er boven staat:

```
<!-- If you dont want to use the sample map, either set enabled="no" in the next line or remove the xml file from the maps directory -->
<mapset enabled="yes">
<xi:include href="$NAVIT_SHAREDIR/maps/*.xml"/>
</mapset>
```

Verander hierbij de regel <mapset enabled=”yes”> in:

```
<mapset enabled="no">
```

Navit en GreeNav aanpassen en compileren

Navit kan met behulp van de Anjuta IDE worden aangepast. Het is echter ook mogelijk om dit met behulp van een normale tekstverwerker te doen. Nadat de code is gewijzigd kun je Navit in Anjuta compileren. Je kan hem ook bouwen en uitvoeren via de commandline. Start een commandoprompt op en voer volgende code uit (let op de voorbeeldmap):

```
cd /home/gebruiker/Documenten/Navit-0.1.1
sudo ./configure
sudo make
cd navit
./navit
```

Bijlage 3 GreeNav XML-configuratie

Om GreeNav te configureren wordt het navit.xml configuratiebestand gebruikt. Hierin is een eigen type aangemaakt met de naam "gsn_xml". In het label daarvan kan de data worden ingevoerd. Bij het starten van Navit zal de XML worden ingelezen en zal de GreeNav API de invoer verwerken.

De invoer wordt met behulp van strepen ("|") gescheiden. Op deze manier kunnen een aantal parameters worden doorgegeven. De eerste parameter geeft aan waartoe de invoer behoort. De rest van de parameters bevatten informatie en waarden.

Init
Gegevens die beginnen met "init" moeten als eerste worden geplaatst. Deze zorgen ervoor dat GreeNav bij het starten de juiste waarden krijgt, maar ook de juiste grootte van de array. Zonder een juiste configuratie hiervan kan de software crashen.

init|start
Markeert de start van de initialisatie voor GreeNav

init|end
Het einde van de initialisatie

init|poi|array_loc|$array_grootte
gsn_poi_loc[] krijgt $array_grootte elementen

init|poi|array_cat|$locatie|cat|$array_grootte
gsn_poi_loc[$locatie].cat[] krijgt $array_grootte elementen

init|poi|array_poi|$locatie|cat|$categorie|poi|$array_grootte
gsn_poi_loc[$locatie].cat[$categorie].poi[] krijgt $ array_grootte elementen

init|route|array_route|$array_grootte
gsn_route[] krijgt $array_grootte elementen

init|route|array_poi|$route|poi|$array_grootte
gsn_route[$route].poi[] krijgt $array_grootte elementen

init|paal|array_paal|$array_grootte
gsn_palen[] krijgt $array_grootte elementen

poi

Poi (Point of Interest) geeft een bestemming aan. Hiermee kunnen locaties, categorieën en bestemmingen worden ingeladen. Deze bevatten onder andere informatie over de bestemming.

poi|loc|$locatie|$veld|$waarde
gsn_poi_loc[$locatie].$veld krijgt de waarde $waarde

poi|cat|$locatie|cat|$categorie|$veld|$waarde
gsn_poi_loc[$locatie].cat[$categorie].$veld krijgt de waarde $waarde

poi|poi|$locatie|cat|$categorie|poi|$bestemming|$veld|$waarde
gsn_poi_loc[$locatie].cat[$categorie].poi[$bestemming].$veld krijgt de waarde $waarde

route

Route geeft de routes aan. Hiermee kunnen routes worden aangemaakt in het navigatiesysteem.

route|route|$route|$veld|$waarde
gsn_route[$route].$veld krijgt de waarde $waarde

route|dest|$route|point|$point|$veld|$waarde
gsn_route[$route].point[$point].$veld krijgt de waarde $waarde

palen

De oplaadpalen kunnen hiermee worden aangemaakt in het systeem.

paal|paal|$paal|$veld|$waarde
gsn_palen[$paal].$veld krijgt de waarde $waarde

Aanduidingen met een $ zijn variabel. Afhankelijk van het veld of de index moet dit een string of integer zijn.

Bijlage 4 SD-Kaart indeling

De SD-kaart bevat alle data voor het navigatiesysteem. Hier zijn de configuratie, de foto's en de kaart op te vinden. Door de beperkingen in de Navit software moet de SD-kaart "SDDATA" worden genoemd. In dat geval wordt deze als "/media/SDDATA/" in Linux gemount.

Indeling
Om de inhoud overzichtelijk te houden is er een indeling vastgesteld. De hoofdmap met alle data is "Greenav" genaamd. Daarbinnen staat het XML-configuratiebestand. In de map "map" is de kaart te vinden. De map "img" bevat alle afbeeldingen. De map "sys" bevat alle benodigde systeemgegevens zoals afbeeldingen voor het OSD.

Greenav
In de Greenav map komt de configuratie te staan. Binnen de map bevinden zich ook nog de mappen "map", "img" en "sys".

Mogelijke indeling:

```
/media/SDDATA/Greenav/navit.xml
/media/SDDATA/Greenav/map
/media/SDDATA/Greenav/img
/media/SDDATA/Greenav/sys
```

Greenav/map
De inhoud van de map "map" bestaat enkel uit de kaart. De te gebruiken kaart is te specificeren in het configuratiebestand navit.xml.

Mogelijke indeling:

```
/media/SDDATA/Greenav/map/benelux.bin
/media/SDDATA/Greenav/map/europa.bin
```

Greenav/img
De map "img" bevat alle afbeeldingen die gebruiker er op heeft gezet. De structuur hiervan komt overeen met de nummering in de struct. De afbeeldingen en mappen worden zo genummerd dat deze één op één overeenkomen met de waarden in de XML.

Locatieafbeeldingen zullen in een map komen met de naam "loc_x", waarbij x een nummer is dat overeenkomt met de index in de data-struct en XML. De categorieën en bestemmingen zullen eveneens op deze wijze worden ingedeeld in de mappen "cat_y" en "poi_z", waarbij y en z eveneens waarden zijn die overeenkomen met de index. Afbeeldingen voor routes komen in de map "route_x".

De afbeeldingen zelf zullen de naam "img_x" en "thumb_x" meekrijgen. Ook hier is x weer gelijk aan de index. Verder zorgt deze naamgeving er voor dat de kleine thumbnails ook overzichtelijk terug te vinden zijn.

Mogelijke indeling:

```
/media/SDDATA/img/loc_0/thumb_0.jpg
/media/SDDATA/img/loc_0/cat_0/thumb_0.jpg
/media/SDDATA/img/loc_0/cat_0/poi_0/img_0.jpg
/media/SDDATA/img/loc_0/cat_0/poi_0/thumb_0.jpg
/media/SDDATA/img/loc_0/cat_0/poi_1/img_1.jpg
/media/SDDATA/img/loc_0/cat_0/poi_1/thumb_1.jpg
/media/SDDATA/img/loc_0/cat_1/poi_0/img_0.jpg
/media/SDDATA/img/loc_0/cat_1/poi_0/thumb_0.jpg
/media/SDDATA/img/loc_0/cat_1/poi_1/img_1.jpg
/media/SDDATA/img/loc_0/cat_1/poi_1/thumb_1.jpg
/media/SDDATA/img/route_0/img_0.jpg
/media/SDDATA/img/route_0/thumb_0.jpg
```

Greenav/sys

In de "sys" map komt alle data die nodig is voor het systeem. Dit zijn onder andere afbeeldingen voor het OSD, maar mogelijk ook opgeslagen log-bestanden.

Mogelijke indeling:

```
/media/SDDATA/sys/img/arrow_next.jpg
/media/SDDATA/sys/img/arrow_prev.jpg
```

Overzicht

Een overzicht van de totale structuur op de SD-kaart:

/media/SDDATA/											
Greenav	/	navit.xml									
	/	sys	/								
			/	img							
	/	map	/	map.bin							
	/	img	/	loc_x	/	thumb_x.jpg					
					/	cat_y	/	thumb_y.jpg			
							/	poi_z	/	img_z.jpg	
									/	thumb_z.jpg	
			/	route_x	/	thumb_x.jpg					
					/	img_x.jpg					

Bijlage 5 Documentatie aangepaste Navit-code

Om de API op te stellen zijn een aantal functies in de bestaande Navit-code aangepast. De locaties en code zijn in deze bijlage gedocumenteerd.

osd_core.c

In osd_core.c is het één en ander aangepast om de xml goed in te laden, en om de diverse knoppen en labels aan te kunnen passen aan de juiste waarde.

1. **static void** osd_button_draw(**struct** osd_button *this, **struct** navit *nav)

Deze functie is aangepast om de juiste afbeelding te laden. Als het een Greenav afbeelding betreft zal vanuit een API functie een string met de afbeeldingslocatie worden doorgeven.

```
249    if(strstr(this->src, "/media/SDDATA/Greenav/sys/img"))
       {
...    ...
278    }
```

2. **static void** osd_text_draw(**struct** osd_text *this, **struct** navit *navit, **struct** vehicle *v)

De code in deze functie zorgt voor het weergeven van de juiste tekst op de labels, aan de hand van de waarden in Greenav. Ook worden hier de coördinaten doorgegeven

```
808    char *gsn_geo;

822    if(strstr(str, "${gsn."))
       {
...    ...
854    }

946    if(!strcmp(subkey, "position_coord_geo"))
       {
....   ...
951    }
```

3. **static void** osd_text_init(**struct** osd_text *this, **struct** navit *nav)

Deze functie is aangepast om een fout te verhelpen waarbij alle labels opnieuw aangemaakt worden. Door de code wordt dit enkel gedaan bij de initialisatie van Greenav.

```
1075   if(gsn_init_return_init_state() == 1)
       {
...    ...
1080   }
```

4. **void** plugin_init(**void**)

In deze functie worden xml typen aangeduid. Daarop zal de juiste functie worden aangeroepen binnen osd_core.c. De items "gsn_img" en "gsn_xml" zijn hier aan toegevoegd.

```
1482    plugin_register_osd_type("gsn_img", osd_gsn_img_poi_new);
        plugin_register_osd_type("gsn_xml", gsn_xml_cmd);
```

5. Verder zijn er vanaf regel 1345 nog enkele functies toe gevoegd. Deze zijn te herkennen aan de "gsn_"-prefix.

```
1345    struct osd_gsn_img_poi {
...     ...
1469    }
```

navit.c

In navit.c zijn de meeste functies aan te treffen die voor de aansturing van het geheel zorgen. Zo worden onder andere commando's gegeven om het scherm opnieuw op te bouwen.

6. Alle variabelen en structs zijn terug te vinden aan het begin van dit bestand.

```
75      #include <fcntl.h>   /* File control definitions */
...     ...
250     struct gsn_struct_palen *gsn_palen;
```

7. **static struct** command_table commands[] = {}

Hierin worden de commando's aangegeven. Deze zijn gekoppeld aan de commando's die door het drukken op de knop worden gegeven. Daarop wordt de aangegeven functie uitgevoerd.

```
810     {"gsn_set_destination_new",command_cast(gsn_api_set_destination_new)},
...     ...
830     {"gsn_warning_nee", command_cast(gsn_btn_warning_nee)}
```

8. **void** navit_init(**struct** navit *this_)

In deze functie wordt het commando gegeven om GreeNav te initialiseren.

```
1480    gsn_init_start(this_);
1582    gsn_init_end();
```

9. **static void** navit_vehicle_update(struct navit *this_, **struct** navit_vehicle *nv)

In deze functie wordt de melding gegeven dat de bestemming is bereikt

```
2290    gsn_route_destination_reached(this_);
```

10. #define GSN_CODE

Onderaan navit.c is onze eigen code te vinden. Hierin staan alle onze functies.

```
2473    #define GSN_CODE
...     ...
7115    #endif
```

start.c

In start.c worden de nodige initialisatie onderdelen van Navit uitgevoerd. Ook wordt hier de xml ingeladen.

11. **int** main(**int** argc, **char** **argv)

Hierin is locatieaanduiding van het configuratiebestand navit.xml aangepast naar de locatie op de sd-kaart.

```
147     config_file = "/media/SDDATA/Greenav/navit.xml";
        if (file_exists(config_file))
        {
...     ...
159     }
```

cursor.c

In cursor.c wordt de huidige locatie aangegeven op de kaart. Deze willen we buiten beeld hebben binnen onze menu's en daarom is deze aangepast.

12. **void** cursor_draw(**struct** cursor *this_, **struct** graphics *gra, **struct** point *pnt, **int** lazy, **int** angle, **int** speed)

Hierin is de positie van de cursor aangepast.

```
116     if(!gsn_api_draw_navigation())
        {
...     ...
120     }
```

Bijlage 6 Menustructuur

Statemachine menu
De menustructuur in GreeNav is heeft wat weg van een statemachine. Dit is dan ook de wijze waarop wordt bepaald welk scherm moet worden weergegeven. Er zijn meerdere statemachines in de software aanwezig.

Menu
De statemachine voor het menu geeft aan in welke pagina men in het menu is. Dit menu, waarin de bestemming kan worden gekozen, werkt als een soort wizard. De totale statemachine is in drie delen opgesplitst. Het eerste deel houdt bij in welke modus de software zich bevindt. Binnen deze modus is een statemachine aanwezig om aan te geven in welke stap men zich bevindt. De pagina geeft tenslotte aan welke pagina binnen deze state is gekozen.

Middels de "Home"-knop zal de statemachine weer terugkeren naar het homescherm. Daarbij worden alle waarden weer gereset. Door op de knop "Terug" te drukken zal de gebruiker een stap terug gaan in de statemachine. De volgende stap zal bepaald worden aan de hand van een keuze op het scherm. De pagina kan worden gewijzigd aan de hand van de pagina navigatieknoppen aan de zijkanten.

De statemachine kan worden gezien als een tabel waarin alle schermen zijn opgenomen:

	State 0	State 1	State 2	State 3	State 4	State 5
Modus 0	**Home-scherm**	**Dashboard**				
Modus 1		**Navigatie**				
Modus 2		**Locaties** + pagina	**Categorieën** + pagina	**Bestemmingen** + pagina	**Informatie**	**Navigatie**
Modus 3		**Routes** + pagina	**Informatie**	**Informatie**	**Navigatie**	

De verschillende modi:

Modus 0: "Rijden zonder navigatie"
Modus 1: "Rijden met navigatie"
Modus 2: "Navigeer naar locaties"
Modus 3: "Maak een rondrit"

Toestandswisseling

De toestand van de statemachine wordt gewijzigd door een actie van de gebruiker. In onderstaande drie tabellen wordt weergegeven wat de volgende toestand is binnen de statemachine na het drukken op een knop.

Bij een keuze

De uitgevoerde actie bij het kiezen van een locatie, categorie, bestemming of dergelijke.

	State 0	State 1	State 2	State 3	State 4	State 5
Modus 0		→				
Modus 1		→				
Modus 2		→	→	→	→	→
Modus 3		→	→	→	→	

Bij het drukken op de terugknop

Wanneer op de terugknop wordt gedrukt, zal er een stap terug in de statemachine worden gezet.

	State 0	State 1	State 2	State 3	State 4	State 5
Modus 0	←					
Modus 1	←					
Modus 2	←	←	←	←	←	
Modus 3	←	←	←	←		

Bij het terugkeren naar home

Wanneer er op de home-knop wordt gedrukt zal het navigatiesysteem terugkeren naar het beginscherm.

	State 0	State 1	State 2	State 3	State 4	State 5
Modus 0						
Modus 1						
Modus 2						
Modus 3						

Bijlage 7 GreeNav compileren

In deze bijlage wordt beschreven hoe het compileren van GreeNav voor ARM11 in zijn werk gaat middels de Qemu-emulator.

Installeren van Qemu

Middels de Qemu-emulator kunnen we een ARM11 processor emuleren op de computer. Alhoewel er een versie voor Windows is, kiezen we voor de Linux versie van Qemu. Deze is beter doorontwikkeld en getest. We hebben Qemu gebruikt onder Ubuntu (versie 9.04 en 9.10, i386 architectuur (x86)). Stappen met asterisk (*) hoeven in principe maar eenmalig worden doorlopen.

1. **Installeer Qemu***
   Open een terminalvenster en type het volgende commando in:

   ```
   sudo aptitude install qemu
   ```

2. **Download de images***
   Download de images voor Qemu en zet deze in een map. Wij hebben gekozen voor "~/qemu". De images zijn te vinden op http://people.debian.org/~aurel32/qemu/armel/. Download de volgende bestanden:

   debian_lenny_armel.qcow2
   initrd.img-2.6.26-1-versatile
   vmlinuz-2.6.26-1-versatile

3. **Maak een map aan voor gedeelde documenten***
   Maak een map "share" aan in de map waar ook de zojuist gedownloade bestanden staan. Zet in deze map ook de sourcecode van GreeNav.

4. **Start Qemu vanuit de terminal**
   Start de emulator met al zijn instellingen vanuit een terminal. Navigeer eerst naar de map met bestanden. Voer het volgende uit:

   ```
   cd ~/
   cd qemu
   qemu-system-arm -M versatilepb -kernel vmlinuz-2.6.26-1-versatile -initrd initrd.img-2.6.26-1-versatile -hda
           debian_lenny_armel.qcow2 -append "root=/dev/sda1" -hdb fat:rw:~/qemu/share/
   ```

5. **Heb geduld**
   Geduld is een schone zaak! De emulator is zeer traag, dus neem de tijd en doe ondertussen wat anders. Het besturingsysteem start ondertussen wel.

6. **Inloggen**
   Log in met de volgende gegeven: username: user, password: user.

7. **Start een root terminal**
   Omdat de grafische userinterface heel traag reageert is het het beste om een terminal te starten. Onder Application » Accessories » Root terminal. Voer daarna het root wachtwoord in. Deze is "root".

8. **Mount de gedeelde map**
   Om de gedeelde map te kunnen benaderen moet deze eerst worden aangekoppeld in Linux. Voer achtereenvolgens de volgende commando's uit om deze te mounten aan /media/share:

```
cd /media
mkdir share
cd /dev
mount sdb1 /media/share
```

Compileren van GreeNav

Nadat Qemu is geïnstalleerd en draait kunnen we gaan compileren in de emulator. Daarvoor zijn ook enkele handelingen nodig. Stappen met asterisk (*) hoeven in principe maar eenmalig worden doorlopen.

1. **Dependencies installeren***
   Om een goede build te krijgen moeten eerst enkele pakketten, waar Navit afhankelijk van is, worden geïnstalleerd. Deze zijn ook terug te vinden op http://wiki.navit-project.org/index.php/Debian_dependencies

```
sudo aptitude update
sudo aptitude install libtool automake autoconf libglib2.0-dev zlib1g-dev gettext cvs libgtk2.0-dev libtiff4-dev libgps-dev
```

2. **Code kopiëren**
   Om de broncode van GreeNav te kunnen compileren moeten we deze eerst in kopiëren. In dit voorbeeld staat de broncode in de map "greenav":

```
cd /media/share
cp -r greenav ~/
```

3. **Compileren**
   Het compileren op zich gaat zoals we gewend zijn met het ./configure en make commando. Echter duurt het compileren een zeer lange tijd. Laat de computer dus rustig zijn werk doen. Ook in dit voorbeeld is de map met broncode greenav genaamd:

```
cd  ~/greenav
./configure
make
```

4. **Inpakken en naar de gedeelde map kopiëren**
   Nu we een gecompileerde versie hebben, kunnen we deze inpakken en naar de gedeelde map sturen. Door alles in te pakken zorgen we er voor dat het proces van kopiëren sneller en foutloos gaat. Deze handeling neemt wat tijd in beslag.

```
cd ~/
tar zcvf Greenav.tar.gz greenav/
cp -r Greenav.tar.gz /media/share
```

Overzetten naar de SmartQ5

We hebben nu een uitvoerbaar bestand van GreeNav. Deze kunnen we vervolgens op een SD-kaart of USB-stick zetten en in de SmartQ5 plaatsen. Daarna kunnen we de bestanden uitpakken en GreeNav starten. Stappen met asterisk (*) hoeven in principe maar eenmalig worden doorlopen. Dit gaat als volgt:

1. **Installeren gdb***
   Om debug meldingen te weergeven hebben we gdb nodig. Zorg eerst voor een internet-verbinding via één van de mobile-netwerken. Probeer ze allemaal. Vergeet daarna niet aan te melden via de browser. Start vervolgens een terminal en voer het volgende uit:

```
sudo aptitude update
sudo aptitude install gdb
```

2. **Bestanden kopiëren en uitpakken**
   We kunnen nu simpelweg de bestanden kopiëren en uitpakken op de volgende manier.

```
cd /media/SDDATA
sudo cp -r Greenav.tar.gz ~/
cd ~/
tar xvf Greenav.tar.gz
```

USB Host aanzetten bij opstarten*
Om standaard bij het opstarten de USB-host functie aan te zetten, moet een systeem-bestand worden verander. Open dit bestand met onderstaand commando:

```
gksu gedit /etc/rc.local
```

Plak hierin de volgende code voor het "exit 0"-statement:

```
echo 1 > /sys/devices/platform/hhtech_gpio/usbpwr_en;
echo 1 > /sys/devices/platform/hhtech_gpio/usbotgdrv_en;
modprobe ohci-hcd;
modprobe usb-storage;
```

Sla het bestand vervolgens op.

3. **Communicatie met USB mogelijk maken***
   Om met de microcontroller te kunnen communiceren zijn nog een aantal drivers nodig. Deze staan op de DVD in de map »Navigatie software\SmartQ5«. Kopieer de map "usb_module" naar de /home/user-map van de SmartQ5.

   Deze functie wordt door het opstartscript gestart. Voor testdoeleinden is deze ook te starten in de console middels onderstaande commando 's:

```
cd /home/user
cd usb_module
sudo insmod usbserial.ko
sudo insmod ftdi_sio.ko
```

4. **Pakketten voor GPSD installeren***
   Om met GPS-module te communiceren is het programma GPSD nodig. De versie die via apt-get te downloaden is werkt echter niet met GreeNav. Daarom staan er op de DVD 2 pakketten voor GPSD die wel werken. Deze zijn terug te vinden in » Navigatie software\SmartQ5\Packagess«. Installeer eerst "libgps17_2.37-7_armel.deb" en daarna " gpsd_2.37-7_armel.deb"

5. **Herstarten***
   Enkele pakketten vereisen na installatie een reboot. Start de SmartQ5 dus opnieuw op.

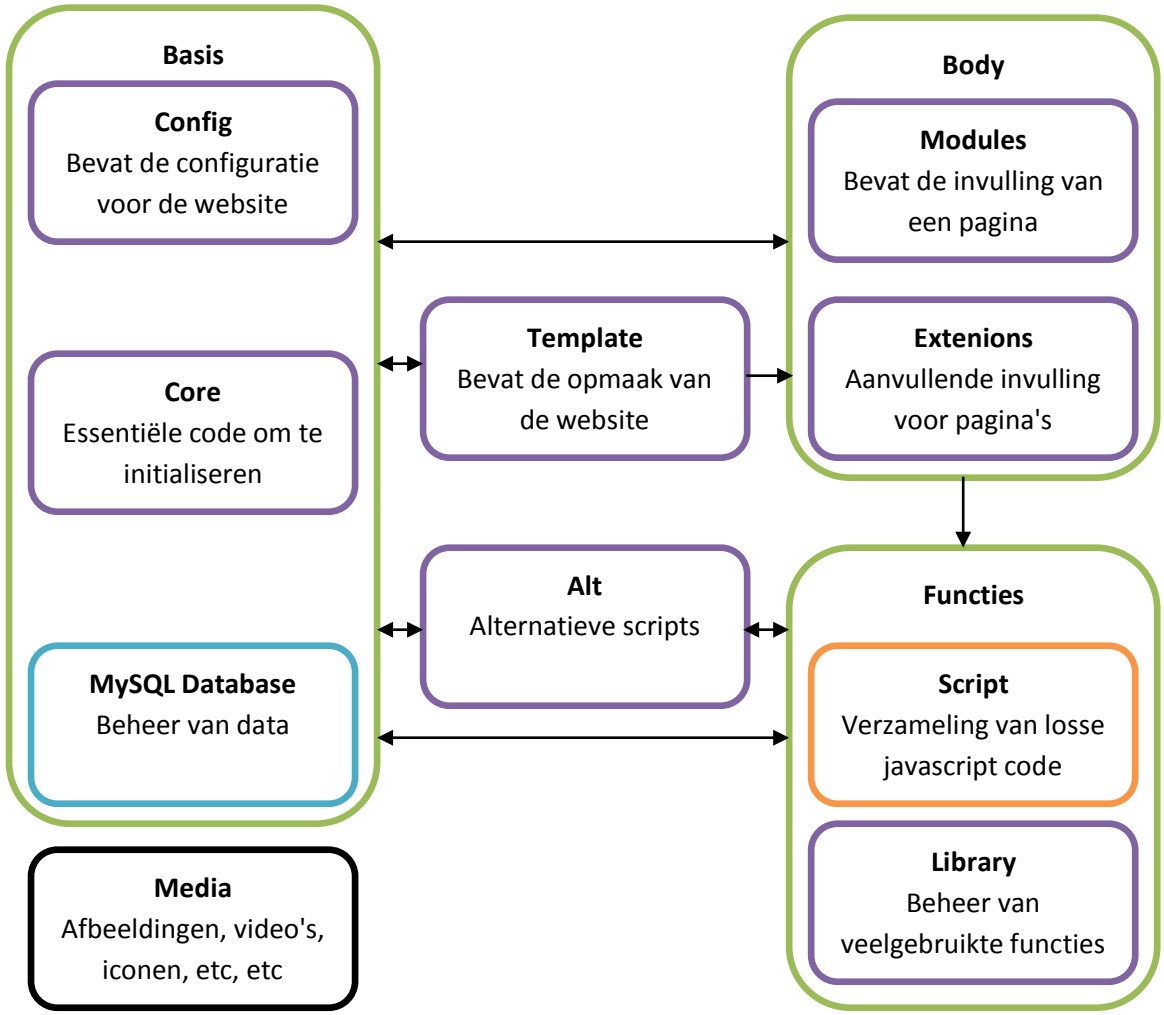
6. **GreeNav starten**
   Vervolgens is het slechts een kwestie van GreeNav starten met het uitvoerbaar bestand dat op de DVD staat. Door op greenav-start.sh te dubbelklikken zal het navigatieprogramma starten. Dit bestand is te vinden in »Navigatie software\SmartQ5\Bash-bestanden«

Bijlage 8 Framework website

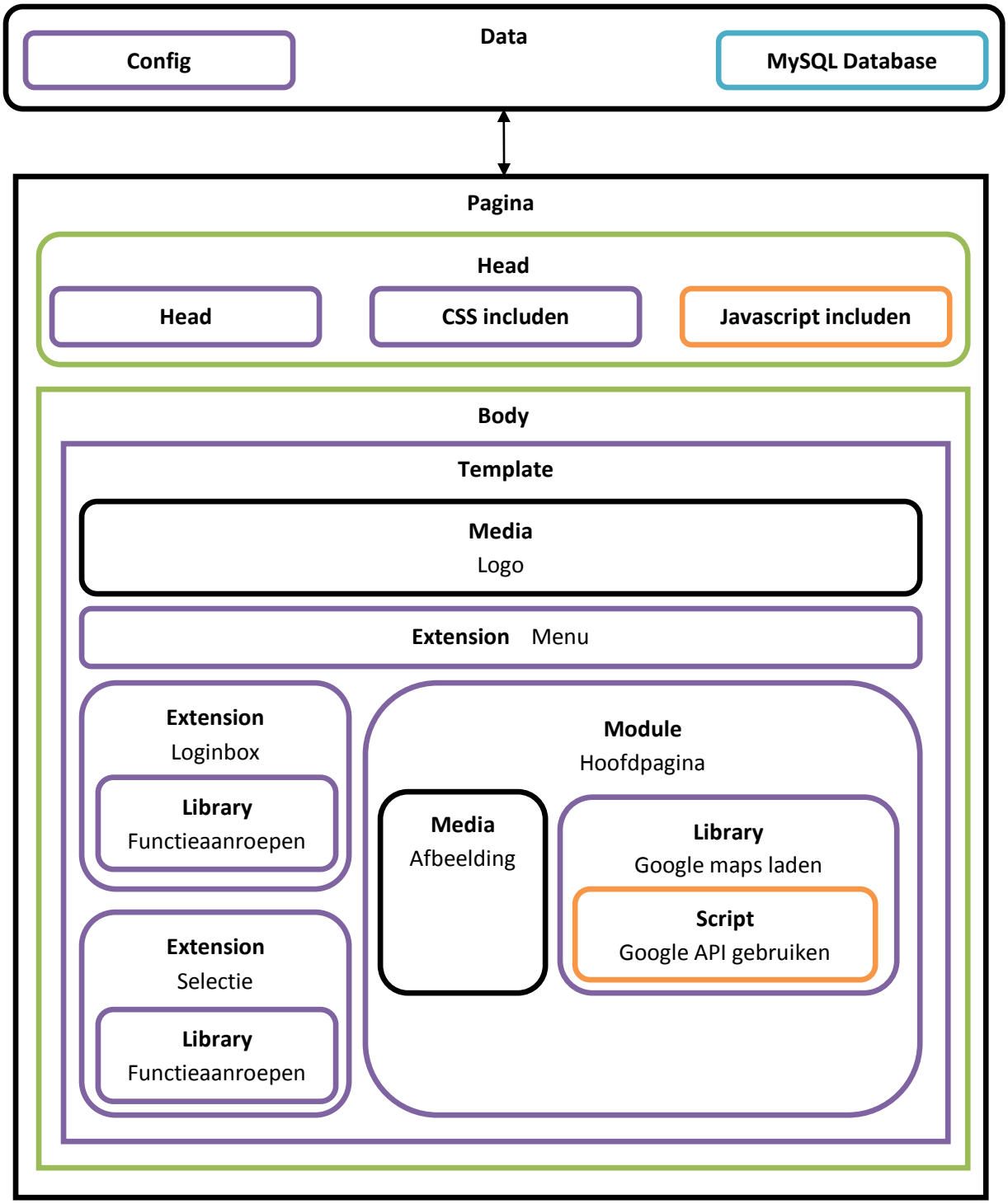
In deze bijlage beschrijven we de opzet van het framework voor de complete website. In deze bijlage wordt onder andere beschreven hoe alles wordt geladen, wat belangrijke bestanden zijn, hoe alles inelkaar steekt en wat de belangrijkste variabelen zijn.

8.1 Opzet van het framework

Hieronder een overzicht van de verhoudingen tussen de diverse onderdelen in het framework.



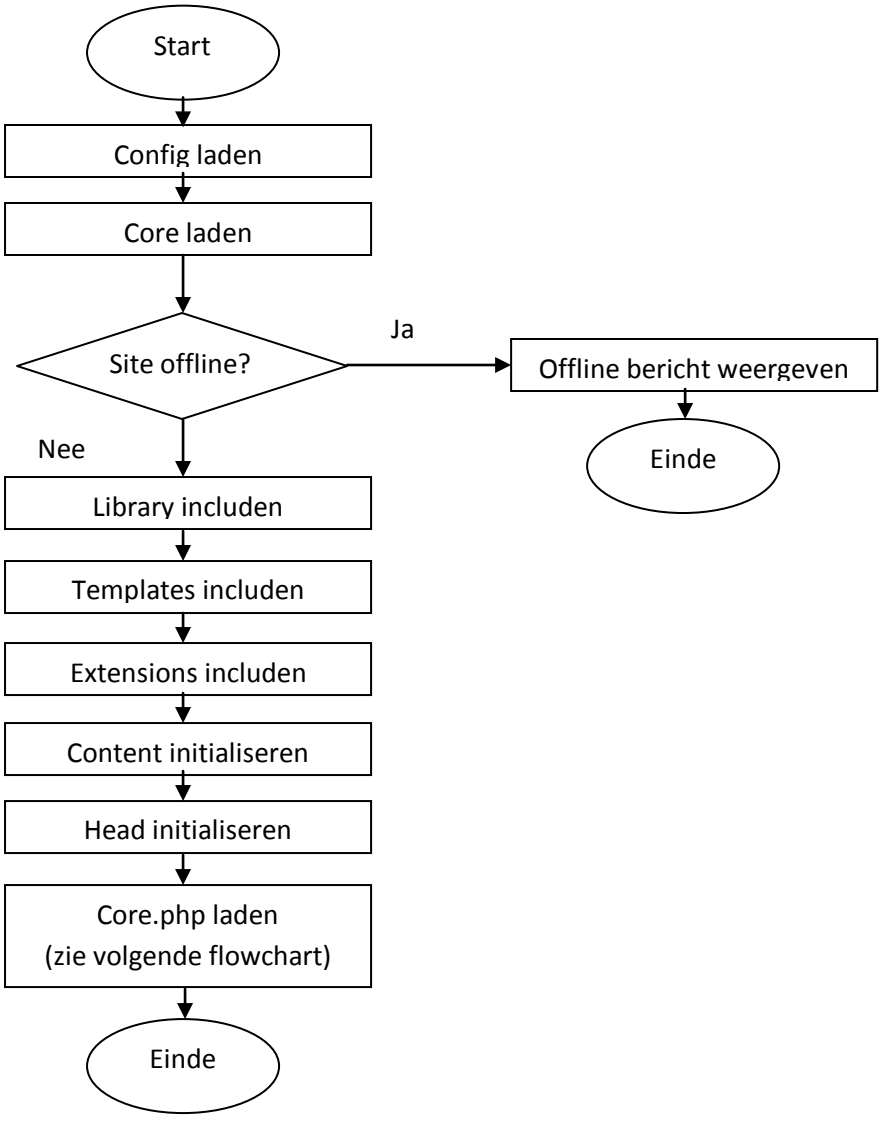
Dit schema is ook te vertalen naar een zichtbare website die door de browser zal worden weergegeven. In onderstaand afbeelding staat een schematische weergave van de opbouw van een pagina op de site.



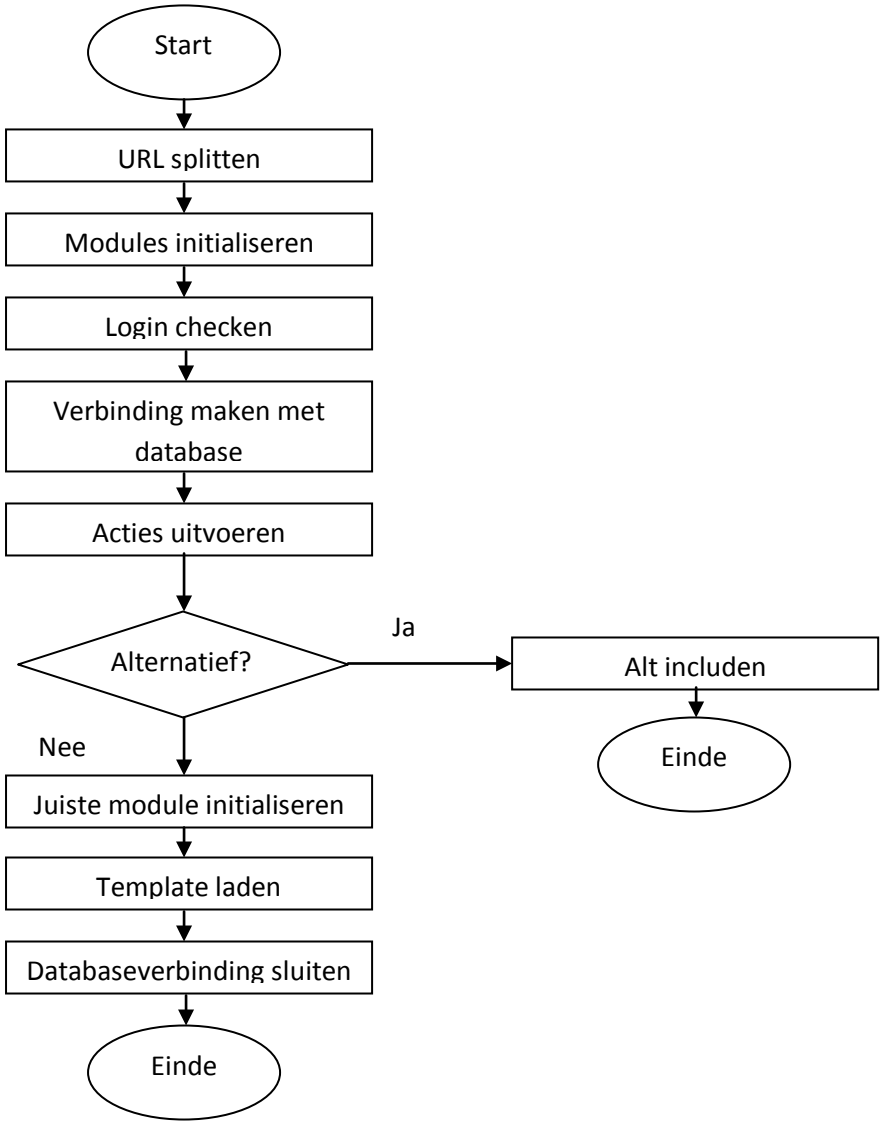
8.2 Flowcharts

Flowchart core/main.php

Hieronder een flowchart met de manier waarop alles wordt ingeladen. Deze cyclus wordt elke keer herhaald wanneer een pagina wordt geopend.



Flowchart core/core.php



8.3 Bouwstenen website

8.3.1 Alt

Binnen de Alt-map (alternatief) komen de scripts die niets te maken hebben met de website zelf. Dit zijn voornamelijk onderdelen die door andere applicaties kunnen worden gebruikt. Een bekend voorbeeld is een RSS-feed, waarmee je automatisch updates krijg van bijvoorbeeld nieuwsberichten. Andere opties zijn bijvoorbeeld een sitemap in XML-formaat, welke door onder andere Google kan worden gebruikt om de website te indexeren.

Nog geen code geschreven. Main.php wordt echter wel aangeroepen op het moment dat de url met "http://greenav.nl/alt/" begint.

8.3.2 Config

Deze map bevat de configuratie van de website. De centrale configuratie zorgt er voor dat instellingen gemakkelijk te wijzigen zijn. Hieronder vallen onder andere het menu en de naam van de website. Belangrijker is echter de serverconfiguratie. Hierdoor kunnen we de website lokaal testen met de instellingen die hiervoor gelden. Voor de webhosting hoeven een aantal instellen te worden gewijzigd, waarna de website ook daar loopt.

Bevat de hele configuratie. Elementen die in de toekomst mogelijk kunnen veranderen zouden een eigen variabele moeten krijgen binnen de configuratie. Dit is onder andere al gedaan voor het menu en de tabellen in de database.

Belangrijk is main.php. Dit bestand wordt door index.php geladen. Binnen main.php worden alle configuratiebestanden toegevoegd aan de code. Zorg ervoor dat nieuwe bestanden binnen config ook aan main.php worden toegevoegd zodat deze worden geladen.

8.3.3 Core

Core bevat de basis van de website. Dit essentiële deel wordt als eerste ingeladen nadat de configuratie is ingeladen. De core zorgt onder andere voor het maken van verbinding met de database. Verder zal ook de juiste template worden gekozen. Ook zullen de library, scripts en de juiste module worden ingeladen. De core kan het best worden vergeleken met een initialisatie en de main-loop in C.

Bij het laden van index.php zal ook de core worden toegevoegd. Dit gebeurt met behulp van het includen van main.php. Vanuit main.php worden enkele andere onderdelen toegevoegd. Daarna zal core.php worden aangeroepen die zorgt voor de initialisatie van de opgevraagde webpagina. De core bevat verder de functies voor het laden van het template, een module en de extensies. Deze code is terug te vinden in de mappen.

8.3.4 Extensions

Hierin zijn de extensies te vinden. Dit zijn componenten die los worden ingeladen en vanuit een module of template kunnen worden aangeroepen. Het zijn doorgaans kleine en simpele bouwstenen voor de website. Voorbeelden van extensies zijn een loginbox of hoofdmenu.

Binnen de map "extensions" vinden we wederom een main.php, welke ook bij initialisatie wordt ingeladen. De extensies zelf komen in mappen te staan. Hierin kunnen vrij sourcefiles worden aangemaakt voor de code. Let echter wel op dat "extension.php" verplicht is. Dit is het bestand dat wordt aangeroepen op de plaats waar de extensie moet komen. De mapnaam bepaald de naam van de extensie. Deze dient gebruikt te worden bij het aanroepen van een extensie. Dat laatste gaat met de functies die te vinden zijn in "/core/content/extensions.php".

8.3.5 Library

De library bevat allemaal functies. Deze bibliotheek kan vanuit de hele code worden gebruikt en maakt het gemakkelijk om functies te hergebruiken die vaker voor komen. De library bevat zo onder andere functies om informatie over de gebruiker op te vragen. Ook zijn er functies voor onder andere Google Maps, de MySQL database en afbeeldingen terug te vinden.

Alle functies zijn gebundeld per doel. Zo zijn er functies voor Google Maps, image resizing, e-mail, sorteren, etc, etc. Nieuwe functies voor een nieuw doel kunnen in nieuwe bestanden worden geplaatst. Ook kunnen bestaande onderdelen natuurlijk worden uitgebreid. Zorg bij het aanmaken van een nieuwe sourcefile wel dat deze ook in "main.php" wordt aangegeven.

8.3.6 Media

In de map media zijn onder ander afbeeldingen en video's terug te vinden. Ook is er een standaard pakket van icoontjes en smiley's terug te vinden. De map kent verder geen scripts en wordt verder niet geladen.

Hierin kunnen alle losse bestanden worden geplaatst. Deze worden met simpele HTML-code aangeroepen en behoeven geen verdere aanpassingen. Let wel op met de images-map. Deze moet schrijfbaar zijn (CHMOD 777) om vanaf de website afbeeldingen up te loaden.

8.3.7 Modules

In de map modules komen de echte pagina's te staan. Er kan slechts één module worden geladen per pagina. Deze bevat dan ook de zogenaamde 'content' van een pagina. Voorbeelden hiervan zijn onder andere het overzicht van locaties of het laten zien van een nieuwsbericht.

Modules staan, net als extensies, in hun eigen map. Binnen de map modules staat het bestand main.php, welke zorgt voor het initialiseren van onderdelen. Een module wordt ook wel gezien als de pagina. Op deze manier kun je voor iedere pagina bepaalde eigenschappen instellen zoals de te laden modules.

Binnen iedere module komt ook een main.php voor. Deze wordt echter enkel geladen op het moment dat die module ook echt geladen zal worden.

Bij het aanmaken van een nieuwe module zal deze main.php aanwezig moeten zijn. Hierin moet op zijn minst de $module variabele worden aangegeven. Deze dient dezelfde naam te krijgen als de mapnaam. Ook is een module.php verplicht. Dit is het bestand dat wordt aangeroepen op het moment dat de zojuist aangegeven module wordt geladen. In module.php kunnen vervolgens functies worden aangeroepen.

8.3.8 Scripts

Hierin is alle Javascript terug te vinden. Bij het laden van de pagina zal een stuk PHP-code zorgen dat de links naar de Javascript bestanden zal plaatsvinden binnen de head-tag van de HTML-code. Deze map bevat onder andere het script om de selectie te wijzigen. Ook standaard scripts van het internet, zoals Lytebox voor het weergeven van grote afbeeldingen zijn aanwezig.

In deze map is alle Javascript code terug te vinden. Deze is verdeeld over een aantal mappen, waaronder ook standaard scripts van het web. Ook hier moeten de te gebruiken scripts worden aangegeven in main.php. Dit gaat echter middels HTML-code, zoals ook te zien is in het aanwezige voorbeeld. Het script core.js wordt ook uitgevoerd bij het laden van de pagina. Hierin kan dus initialisatie van variabelen binnen Javascript plaatsvinden.

8.3.9 Template

Hierin komt het template van de website. Dit is het geraamte waarin de website zal worden weergegeven. Deze zorgt dan ook voor het uiterlijk van de website zelf. Binnen een template vinden we veel HTML-code voor de indeling. Daarbinnen worden op de juiste plaatsten de module en extensies aangeroepen. Ook vinden we hier de CSS-bestanden terug.

Templates worden ook aangeroepen op basis van de mapnaam. De te gebruiken template wordt aangegeven in de configuratie. Binnen de templates vinden we ook een main.php terug. Deze bevat ook weer initialisatiegegevens. Opgemerkt moet worden dat hier ook een functie in moet worden aangemaakt, namelijk "template_load()". Deze wordt aangeroepen op het moment dat het template moet worden weergegeven.

8.4 Variabelen

De meeste variabelen worden slechts lokaal gebruikt. Dit is in de lijn met de manier waarop PHP werkt. Hierbij zijn alle variabelen in principe enkel lokaal te benaderen. Uitzonderingen hierop zijn de zogenaamde "superglobals" binnen PHP. Dit zijn onder andere de variabelen $_GLOBAL, $_POST en $_GET.

Voor het framework zijn echter nog een aantal variabelen aangemaakt die veel gebruikt worden binnen de site. Deze zijn te gebruiken binnen een functie met behulp van de "global" functie Een vooerbieeld hiervan is de code "global $cfg;". Hieronder een overzicht van de belangrijkste variabelen die hiertoe behoren.

Variabele $cfg

Deze variabele bevat de complete configuratie. De configuratie-variabele komt veel voor wegens de belangrijke data die deze bevat. Het is een grote array met veel waarden. Alle variabelen worden ingesteld in de config-map van het framework.

Met behulp van deze variabele kan de website gemakkelijk worden overgezet naar een andere host. Ook kunnen databasegegevens gemakkelijk worden veranderd. Andere onderdelen zijn onder andere de gebruikte modules en menu-items. Het doel van de configuratie is dan ook om de website gemakkelijk aan te kunnen passen zonder alle bestanden langs te moeten.

Belangrijke array's binnen $cfg:

$cfg['core']: Bevat informatie over de website zoals de naam en url.
$cfg['database']: Bevat de inloggegevens voor de database.
$cfg['dbtables']: Bevat de gebruikte tabellen.
$cfg['server']: Bevat informatie over de serverconfiguratie.

Variabele $url

De url-variabele bevat de opgesplitste url. Dit is een vervanging van de $_GET-methode die een onoverzichtelijke url geeft. Binnen de core wordt de opgevraagde url opgesplitst naar een array. Deze data kan worden gebruikt om de meegegeven argumenten te achterhalen. Een voorbeeld:

URL: http://greenav.nl/locaties/list/bestemmingen/1/4/1

Array:
$url[0] -> "locaties"
$url[1] -> "list"
$url[2] -> "bestemmingen"
$url[3] -> 1
$url[4] -> 4
$url[5] -> 1

In dit voorbeeld zal een pagina met bestemmingen worden weergegeven. Aan de hand van $url[0] ("locaties") wordt de module locaties geladen. Aan de hand van "list" en "bestemmingen" wordt het script om bestemmingen te laten zien geladen. $url[3 - 5] bevatten de argumenten voor dit script. Deze zijn respectievelijk de gekozen locatie, categorie en pagina.

Variabele $extension

Deze wordt gebruikt om extensies te koppelen. Op deze manier kunnen extensies dynamisch per pagina worden geladen op het punt waar de extensie met het meegegeven id wordt aangeroepen in de template. Dat laatste gebeurt met de functie "load_extension_by_id()". Zorg dat de te laden extensies bij het initialiseren worden aangegeven. Dit zijn de "main.php" bestanden binnen de modules.

Voorbeeld van een extensie:

```
$extension[0]['tite'] = "Login" //titel van de extensie
$extension[0]['name'] "loginbox" //naam van de extensie, MOET GELIJK ZIJN AAN DE MAPNAAM!
```

Deze code zal de loginbox-extensie laden op het punt waar "load_extension_by_id(0)" is geplaatst in de code. Daarvan zal de titel "Login" zijn. Let op dat de opgegeven naam gelijk is aan de mapnaam van de extensie!

Variabele $module

Deze variabele moet worden ingesteld om de juiste module te kunnen laden. Deze wordt ingesteld in de main.php van de betreffende module.

8.5 Configuratie

Mocht het voorkomen dat de website moet worden aangepast, of zelfs op een andere host wordt gezet, dan is dat mogelijk. Het framework is relatief eenvoudig aan te passen op de belangrijke punten. Hieronder een overzicht van wat zoal mogelijk is en waar op gelet moet worden:

8.5.1 Configureren van de hosting

Uiteraard zullen alle bestanden eerst via ftp naar de webhoster moeten worden geupload. Er zijn echter nog een aantal zaken die moeten worden gewijzigd.

1. De core-configuratie. Deze is te vinden in /config/core.php. Hiervan moet mogelijk de url worden aangepast
2. De server-configuratie. Niet iedere host kent een goede superglobal. Vandaar dat we $cfg['server']['root'] gebruiken in plaats van $_GLOBALS['DOCUMENT_ROOT']. Gebruik een andere root wanneer dit nodig is.
3. Pas het .htaccess bestand aan. Deze zorgt voor het herschrijven van de url (zo wordt onder andere "www." Weggelaten). Zorg dat de rewriteconditions aan de host worden aangepast.
4. Eventueel moeten ook de permissies van bepaalde mappen worden aangepast. De mappen "/media/images" en "/media/images/temp" dienen "777" als toegangsrecht te krijgen.

8.5.2 Configureren van de database

Om de data op te slaan zal de database moeten worden aangemaakt. Ook dient deze in de configuratie te worden aangegeven.

1. Maak een gebruiker en database aan binnen het controlepaneel van de hosting.
2. Maak de tabellen aan in PHPMyAdmin. Hiervoor kun je de importeer-functie gebruiken. De code om de tabellen aan te maken is terug te vinden op de cd.
3. Zorg dat de databaseconfiguratie goed staat in /config/database.php.

8.5.3 Nieuwe module aanmaken

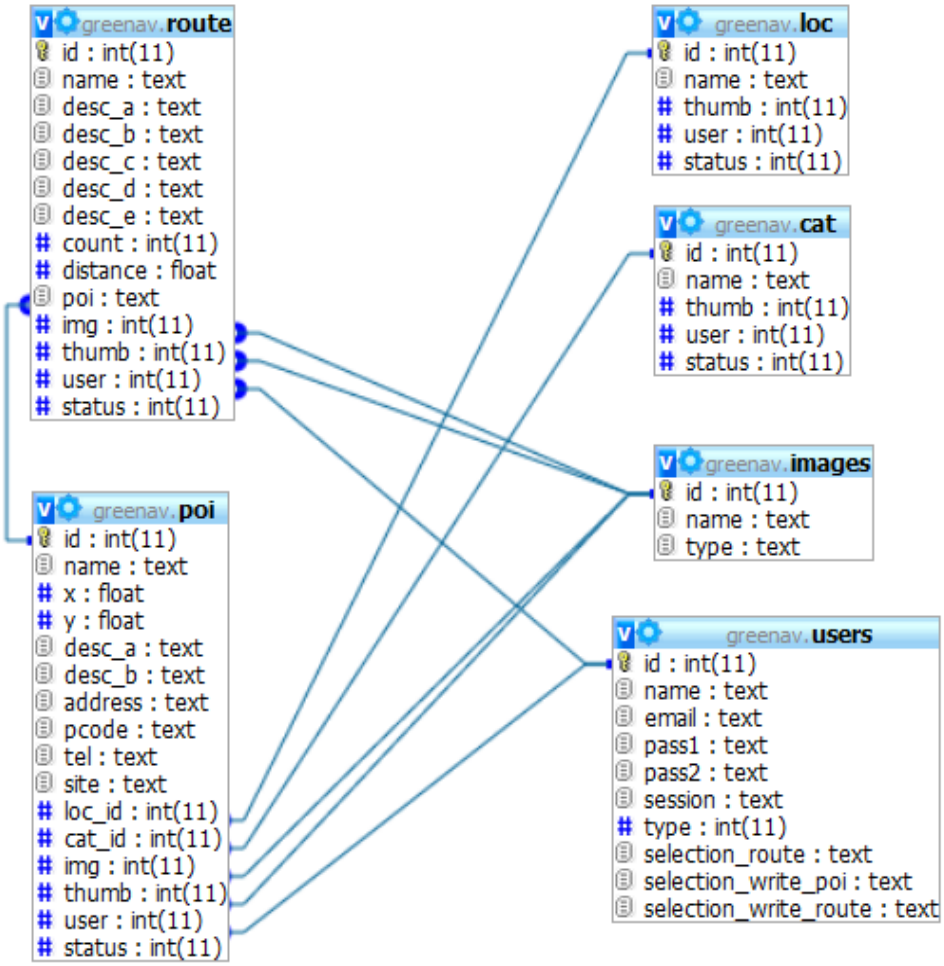
Eerst moet een nieuwe module worden aangemaakt. Nadat een nieuwe module is aangemaakt moet deze ook nog worden toegevoegd aan de configuratie.

1. Kopieer binnen /modules de map dummy en hernoem deze naar de nieuwe module.
2. Zorg dat de variabele $module dezelfde naam krijgt als de map.
3. Schrijf de module in module.php en eventuele losse bestanden (vergeet deze niet te includen!).
4. Voeg een "binding" toe naar de module in /config/binding.php (zie commentaar).
5. Voeg een menuknop toe in /config/menu.php.

Bijlage 9 Databaseontwerp

Overzicht

Hieronder een overzicht van de database en zijn koppelingen



Database greenav

Hieronder zijn alle tabellen van de database terug te vinden.

Tabelstructuur voor tabel cat

Veld	Type	Aanvulling
id	int(11)	
name	text	
thumb	int(11)	Verwijzing naar afbeelding
user	int(11)	Verwijzing naar eigenaar
status	int(11)	0 = openbaar, 1 = Inzichtelijk, 2 = privé

Tabelstructuur voor tabel loc

Veld	Type	Aanvulling
id	int(11)	
name	text	
thumb	int(11)	Verwijzing naar afbeelding
user	int(11)	Verwijzing naar eigenaar
status	int(11)	0 = openbaar, 1 = Inzichtelijk, 2 = privé

Tabelstructuur voor tabel poi

Veld	Type	Aanvulling
id	int(11)	
name	text	
x	float	x-coördinaat
y	float	y-coördinaat
desc_a	text	Omschrijving a
desc_b	text	Omschrijving b
address	text	
pcode	text	Postcode
tel	text	
site	text	
loc_id	int(11)	Verwijzing naar locatie
cat_id	int(11)	Verwijzing naar categorie
img	int(11)	Verwijzing naar afbeelding
thumb	int(11)	Verwijzing naar afbeelding
user	int(11)	Verwijzing naar eigenaar
status	int(11)	0 = openbaar, 1 = Inzichtelijk, 2 = privé

Tabelstructuur voor tabel route

Veld	Type	Aanvulling
id	int(11)	
name	text	
desc_a	text	Omschrijving a
desc_b	text	Omschrijving b
desc_c	text	Omschrijving c
desc_d	text	Omschrijving d
desc_e	text	Omschrijving e
count	int(11)	Aantal punten
distance	float	
poi	text	Opgenomen punten, komma gescheiden (vb: 1,4,7)
img	int(11)	Verwijzing naar afbeelding
thumb	int(11)	Verwijzing naar afbeelding
user	int(11)	Verwijzing naar eigenaar
status	int(11)	0 = openbaar, 1 = Inzichtelijk, 2 = privé

Tabelstructuur voor tabel images

Veld	Type	Aanvulling
id	int(11)	
name	text	
type	text	Vb: "thumb", "image"

Tabelstructuur voor tabel users

Veld	Type	Aanvulling
id	int(11)	
name	text	
email	text	
pass1	text	MD5 hash
pass2	text	MD5 hash
session	text	
type	int(11)	-1 = banned, 0 = normaal, 1 = geregistreerd, 2 = moderator, 3 = admin
selection_route	text	Opgenomen punten, komma gescheiden (vb: 1,4,7)
selection_write_poi	text	Opgenomen punten, komma gescheiden (vb: 1,4,7)
selection_write_route	text	Opgenomen routes, komma gescheiden (vb: 1,4,7)

Tabelstructuur voor tabel articles

Veld	Type	Aanvulling
id	int(11)	
published	int(11)	0 = offline, 1 = online
locked	int(11)	timestamp
section	text	
title	text	
author	text	
datetime	text	
lead	mediumtext	
views	int(11)	

Tabelstructuur voor tabel pages

Veld	Type	Aanvulling
id	int(11)	
articleid	int(11)	Verwijzing naar articles
published	int(11)	0 = offline, 1 = online
page	int(11)	paginanummer
title	text	
content	mediumtext	

Bijlage 10 DVD