|  |  |
| --- | --- |
|  | |
| 3-IT  IoT Advanced: Project parkingplace | |
| **Frederik Crauwels, Brent Druyts, Brandon Morren, Milan Sterkens** | IoT Advanced ITFactory |
|  |
| Academiejaar 2021-2022  Campus Geel, Kleinhoefstraat 4, BE-2440 Geel |



Inhoud

[1 IoT Advanced Challenge 4](#_Toc89186874)

[1.1 Introductie 4](#_Toc89186875)

[1.1.1 Groepsindeling 4](#_Toc89186876)

[2 Plan van aanpak 5](#_Toc89186877)

[2.1 Hoe wij dit project gaan aanpakken 5](#_Toc89186878)

[2.1.1 Introductie 5](#_Toc89186879)

[2.1.2 Verdeling van taken 5](#_Toc89186880)

[2.1.3 Projectplanning 6](#_Toc89186881)

[3 Documentatie 7](#_Toc89186882)

[3.1 Verkenningsfase 7](#_Toc89186883)

[3.1.1 Visualisatie in DB (“virtualisatie in website” en DB). 7](#_Toc89186884)

[3.1.2 Visualisatie op scherm aan de ingang. 10](#_Toc89186885)

[3.1.3 Ultrasoon sensor met parkeerbarrière. 10](#_Toc89186886)

[3.1.4 Nummerplaatherkenning. 12](#_Toc89186887)

[3.1.5 Remote webinterface. 15](#_Toc89186888)

[3.1.6 Push message uitsturen wanneer 4 parkeerplaatsen vol zijn (LDR + Push). 25](#_Toc89186889)

[4 Algemene uitwerking 26](#_Toc89186890)

[4.1 Groepsbespreking: minimum requirements 26](#_Toc89186891)

[4.1.1 Youtube link 26](#_Toc89186892)

[4.1.2 Github link 26](#_Toc89186893)

[5 Extra code 27](#_Toc89186894)

[5.1 Remote webinterface 27](#_Toc89186895)

[5.1.1 Index.php 27](#_Toc89186896)

[5.1.2 license.php 29](#_Toc89186897)

[5.1.3 Welcome.php 32](#_Toc89186898)

[5.1.4 main.py 35](#_Toc89186899)

[5.1.5 TestFile.py 36](#_Toc89186900)

[6 Bibliografie 37](#_Toc89186901)

# IoT Advanced Challenge

## Introductie

Voor het vak “IoT Advanced” krijgen we een interessante challenge voorgeschoteld. Het bedrijf 3-IT heeft een uitdaging voorzien om een oplossing te bekomen om een parking te monitoren, automatiseren en beheren.

Dit allemaal aan de hand van een totaaloplossing die voorziet in volgende aspecten:

* Parkeerbarrière met ultrasoon sensor.
* Analyze van de nummerplaat (nummerplaatherkenning).
* Visualisatie van de data in de database op een display.
* Remote webinterface voorzien.
* Push message sturen als er 4 plaatsen vol zijn.
* Professionele presentatie en demonstratie
  + Voorstellen groepsleden.
  + Plan van aanpak.
  + Aandeel per groepslid en verdelen van taken.
  + Demo en discussie.
* Project bundel (15 pagina’s) met youtube link (2 minuten).
* Evaluatie moment (peer evaluation).

Hier hebben we enkele weken de tijd voor. Het project loopt van:

Vrijdag 29 oktober tot vrijdag 26 november (4 weken).

Op vrijdag 3 december wordt iedereen on campus verwacht. Dit wordt het jury moment!

Uploaden van bundel, presentatie en youtube filmpje moet voor woensdag 1 december 23:59.

### Groepsindeling

Als groep hebben wij het nummer “8” gekregen. Wij zijn dus groep 8 met de volgende teamindeling:

|  |  |  |
| --- | --- | --- |
| **R-nummer** | **Voornaam + naam** | **Klas** |
| R0842921 | Brandon Morren | 2CCS01 |
| R0831536 | Brent Druyts | 2CCS02 |
| R0840939 | Milan Sterkens | 2CCS02 |
| R0802241 | Frederik Crauwels | 2CCS01 |

# Plan van aanpak

## Hoe wij dit project gaan aanpakken

### Introductie

Allereerst gaan wij voorzien dat iedereen de groepsindeling weet en een persoonlijke teams meeting aanmaken voor verdere uitwerking van het project. Dit bevorderd de communicatie en zorgt voor een betere samenwerking.

Graphical user interface, application, Teams

Description automatically generated

Tijdens de “business hours”, de normale uren van dit vak, voorzien wij de samenwerking. Vervolgens gaan de leerkrachten ook telkens er voor zorgen dat er een call komt op deze uren om na te kijken / vragen of alles in orde is, het project lukt,...

Verdere communicatie en uitwerking wordt dan ook in dit kanaal en de bijgevoegde meeting met de leerkracht uitgevoerd.

[Meeting met leerkracht](https://teams.microsoft.com/l/meetup-join/19%3ameeting_OTg4MWY3ZGEtY2VlZS00Y2FmLTkwZmYtMTQyYWEwZDVjODc0%40thread.v2/0?context=%7b%22Tid%22%3a%2277d33cc5-c9b4-4766-95c7-ed5b515e1cce%22%2c%22Oid%22%3a%226d115d7d-dd3f-4b24-8fcb-ca0f3eff2f1d%22%7d)

### Verdeling van taken

Voor de verdere uitwerking voorzien wij een soortgelijke aanpak zoals gezien bij het vak Systems Engineering. Eerst gaan we alles onderzoeken waarbij elk lid van het team een bepaald topic onderzoekt en documenteert.

De indeling is als volgt:

|  |  |
| --- | --- |
| **Voornaam + naam** | **Verantwoordelijk voor** |
| Brandon Morren | Visualisatie in DB (“virtualisatie in website” en DB). |
| Brent Druyts | Ultrasoon sensor met parkeerbarrière. |
| Milan Sterkens | Nummerplaatherkenning. |
| Frederik Crauwels | Remote webinterface. |
| Milan Valkenborgh | Push message uitsturen wanneer 4 parkeerplaatsen vol zijn (LDR + Push). |

Deze indeling is slechts beperkt tot de eerste bespreking. Naargelang de vorderingen in het project zullen deze overlappen waarbij het project als team aangepakt zal worden.

### Projectplanning

In eerste instantie worden de taken verdeeld per groepslid. In dit hoofdstuk gaan we specifiek kijken voor de indeling per datum en de algemene planning opstellen voor het gehele project. Dit houdt in: deadlines, werkuren, algemene planning per dag,...

|  |  |
| --- | --- |
| **Datum** | **To-Do** |
| Vrijdag 29 oktober | Eerste “werkdag”. Opstellen van projectbundel, taakverdeling en verduidelijking individuele opdracht voor vakantie start. Planningen opmaken, deadlines bepalen en groepswerking uitzoeken.  **Eerste taak:** algemene documentatie van elk individu zijn deelopdracht. Verkenningsfase waarin elk lid van het team een onderdeel gaat onderzoeken en documenteren.  **Tweede taak:** vakantie nemen... Tussentijdse vakantie moet ook dienen om rust te nemen. Voor de taken uit, maar neem ook tijd voor jezelf. |
| Vrijdag 12 november | Tussentijdse evaluatie hoe voor iedereen het project verloopt.  Twee weken verder na vakantieperiode. Documentatie en verkenningsfase worden afgesloten.  **ToDo:** samen projectbundel opstellen en verder uitwerken. Verkenningsfase documenteren en starten met overlappende groepswerking.  **Afspreken on-campus / op locatie P305-(4): 12:30 mag je langskomen of materiaal achterlaten (afgesloten kast).** |
| Vrijdag 19 november | Tussentijdse evaluatie hoe voor iedereen het project verloopt.  Finaliseren van werking systemen. Opstellen van basisopstelling en basiswerking van het systeem. De algemene werking van het gehele systeem moet in orde zijn: dit zijn de “basics”.  **ToDo:** kijk na of basisopstellig in orde is.  Werk een constructie uit (papier, karton,...) om de basisopstelling te faciliteren / voor te stellen.  **Extra content aanmaken** voor bonuspunten.  **Afspreken on-campus / op locatie?** |
| Vrijdag 26 november | Tussentijdse evaluatie hoe voor iedereen het project verloopt.  Extra content toevoegen die zijn aangemaakt en afronden het project.  **Ten laatste zondag 28 november: finaliseren projectbundel en afwerken documentatie / powerpoint om woensdag af te geven.**  **Afspreken on-campus / op locatie?** |
| Woensdag 1 december | **Afgeven van projectbundel en powerpoint presentatie**. |
| Vrijdag 3 december | **Jury waarbij iedereen aanwezig moet zijn. Voorstellen van project en demo voorzien.** |

# Documentatie

## Verkenningsfase

Ieder lid van de groep gaat zijn deelstuk verkennen, documenteren en verder uitwerking in dit hoofdstuk. Dit noemen wij de verkenningsfase. De bedoeling is vooral om nog niet in oplossingen te denken maar een brede visie te hebben en verkrijgen van mogelijke uitwerkingen. Dit kunnen wij vervolgens bekijken en bespreken wat voor onze de best mogelijke uitwerking lijkt. Kijk bij “verdeling van taken” om te weten wie waar aan heeft gewerkt.

### Visualisatie in DB (“virtualisatie in website” en DB).

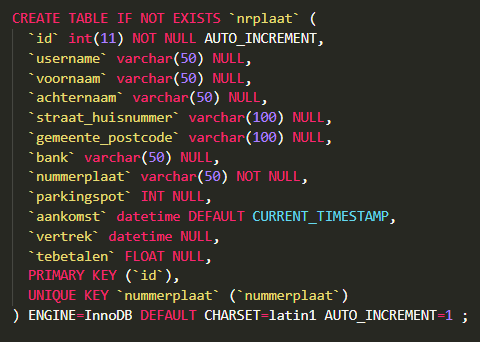
Als database hebben we gekozen voor MariaDB. Een makkelijke, eenvoudige database server die goed overweg kan met PHPMyAdmin. Onze database noemt “parkeergarage”.

Voor de opstelling hebben we drie tabellen:

* Nrplaat: geeft een overzicht van onze gebruikers. Persoonlijke gegevens, betaalgegevens, aankomst- en vertrektijden,...
* Users: algemene login gegevens. Het wachtwoord is geëncrypteerd waardoor zelfs de admin hier géén toegang tot heeft.
* Parkingspot: het aantal parkeerplaatsen en hun beschikbaarheid.

#### Nrplaat

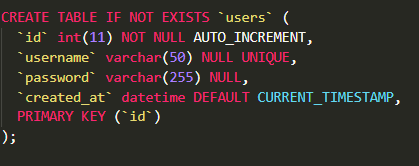
In deze tabel zien we een hoop persoonsgegevens verschijnen. Deze worden ook pas achteraf gelinkt met het account van de gebruiker. De extra gegevens worden ook pas achteraf voorzien door de gebruiker. In eerste instantie komt hier automatisch: een nummerplaat en aankomsttijd. Elke nummerplaat is een uniek record – indien de nummerplaat opnieuw voorkomt zal enkel de “aankomsttijd” veranderen.

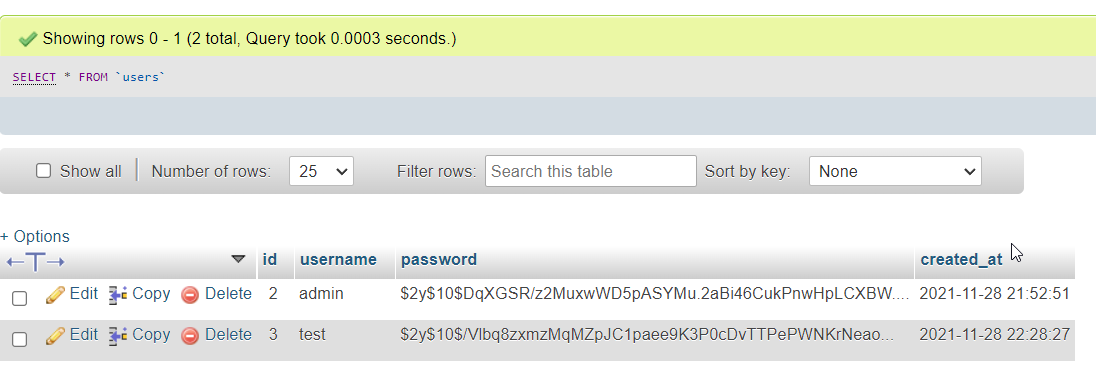




#### Users

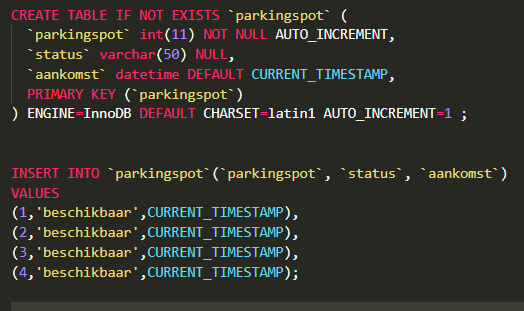
Een tabel voorzien voor de webpagina. Hierin staan login gegevens – deze zullen eerst voorzien worden door de remote webinterface. Indien een gebruiker een account aanmaakt zal deze eerst hier verschijnen. Pas achteraf zal deze gelinkt worden aan een nummerplaat en met tabel “nrplaat” via de webinterface voor gebruikers / admin.

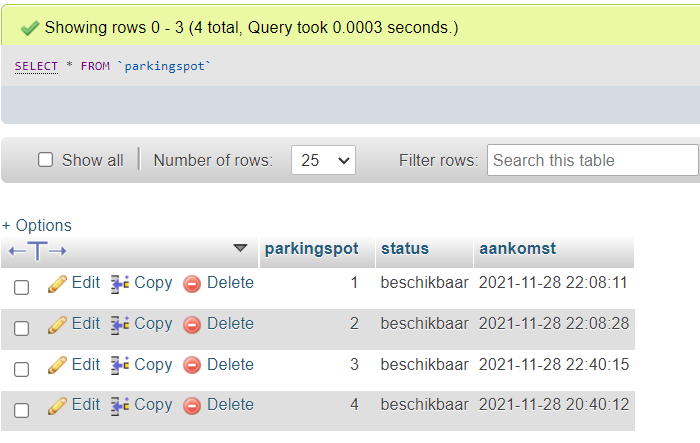




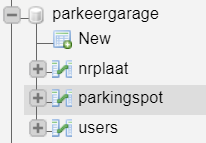
#### Parkingspot

Deze tabel zal gebruikt worden om onze parkeerplaatsen te monitoren. Per parkeerplaats heb je één rij waarin er staat of deze beschikbaar is of niet, en sinds wanneer deze niet meer beschikbaar is, of terug wel beschikbaar is. Deze parkeerplaatsen kunnen we al automatisch voorzien in ons script omdat deze vast staan – en dit gaat voor toekomstige uitbreidingen ook steeds op deze manier kunnen zonder manuele input.





Zo voorzien wij onze gegevens in één database: parkeergarage, met drie tabellen die hierboven vermeld werden. Dankzij PhPMyAdmin hebben we hier steeds een handig overzicht – waardoor ook deze webinterface een onmisbaar deel wordt in dit project.



#### TestFile.py

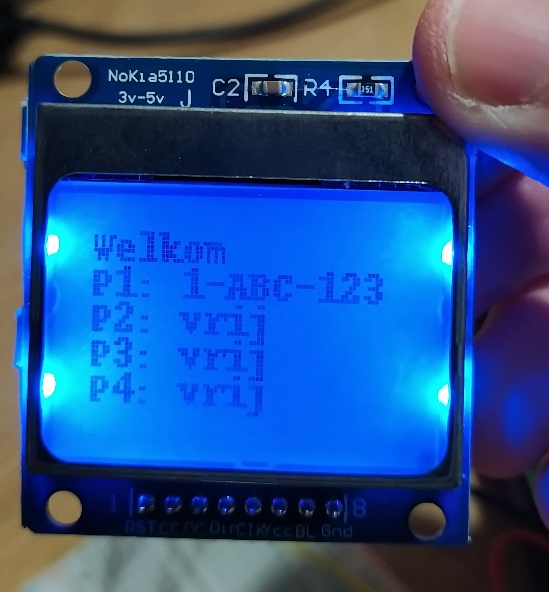
Deze file werd gebruikt in ons project om de data van- en naar de database uit te voeren in een geïsoleerde opstelling. Hierbij werden gegevens van tevoren “voorzien” om te testen of de formules daadwerkelijk toegepast werden. Indien dit werkte – zal dit in onze main.py (basis van on project) ook werken naargelang de gegevens parameters. Zie het als een soort “unit test” vooraleer deze werd toegepast in de praktijk. Deze code kan je helemaal achteraan bekijken bij de “extra code”.

Hier staan vooral de berekeningen voor de prijzen en het versturen / updaten van allerlei gegevens in verschillende tabellen. Denk maar aan nummerplaat, aankomst en vertrek en parkeerplaats beschikbaarheid.

### Visualisatie op scherm aan de ingang.

Aan de ingang van de parking hebben we een scherm (lcd) voorzien waarop je kan zien welke plaatsen bezet zijn en welke vrij zijn. Als er een auto binnenrijdt dan wordt er een foto genomen van de nummerplaat en dan wordt de nummerplaat opgeslagen in de database. Wanneer de auto dan op een parkeerplaats gaat staan dan wordt de nummerplaat van de auto gekoppeld aan de parkeerplaats. Vervolgens wordt het scherm en de web interface aangepast.

Afbeelding met tekst, monitor, binnen

Automatisch gegenereerde beschrijving

### Ultrasoon sensor met parkeerbarrière.

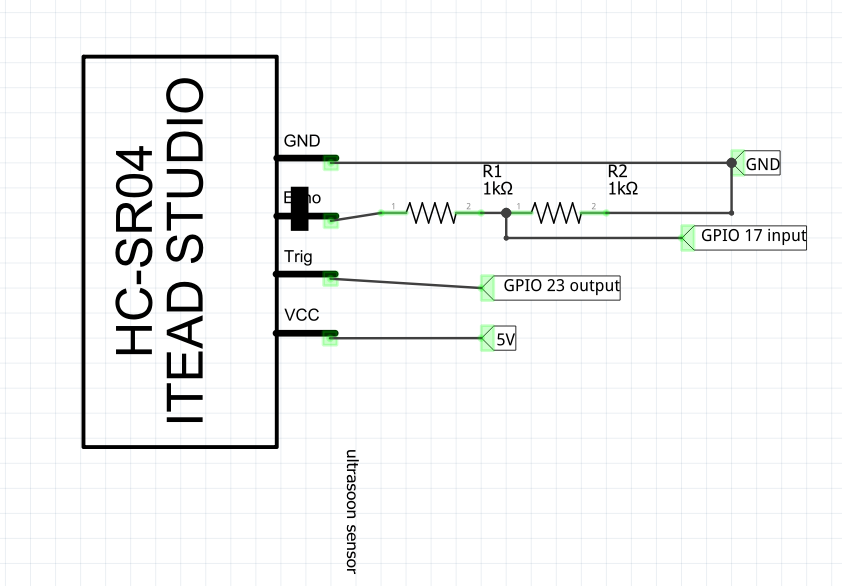
Ik heb als deelopdracht gekozen om te werken met de ultrasoon sensor en de slagboom. Ik heb voor de ultrasoon sensor gekozen omdat we hier vorig schooljaar al mee gewerkt hebben en ik dit een zeer interessant onderwerp vind. Ik heb ook voor de slagboom gekozen omdat ik in het middelbaar al een beetje ervaring heb opgedaan met het programmeren van stappen, DC en servomotoren.

Ultrasoon sensor:

Om te weten of er een auto bij de ingang of uitgang staat heb ik de ultrasoon sensor van vorig jaar gebruikt.

Afbeelding met binnen

Automatisch gegenereerde beschrijving

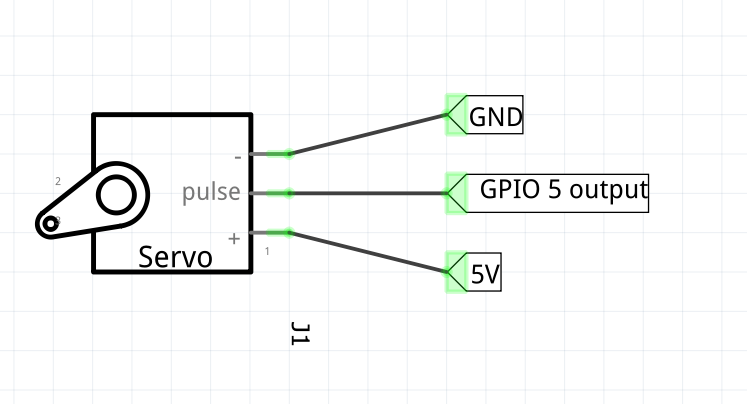


Voor het programma voor de ultrasoon sensor heb ik het basisprogramma van IoT essentials genomen en dit aangepast.

Motor voor slagboom:

Voor de motor voor de slagboom had ik eerst een stappenmotor genomen. Maar deze motor kan alleen maar naar links of naar rechts draaien, dit is dus niet handig voor een slagboom. Deze motor kan gebruikt worden voor een slagboom in combinatie met een tandwielenkast, deze is er niet standaard bij. Bij een servomotor kan je de precieze stand van de motor achterhalen omdat hier een tandwielkast bij zit. Daarom heb ik voor dit project gekozen om te werken met een servomotor.

Afbeelding met adapter

Automatisch gegenereerde beschrijving

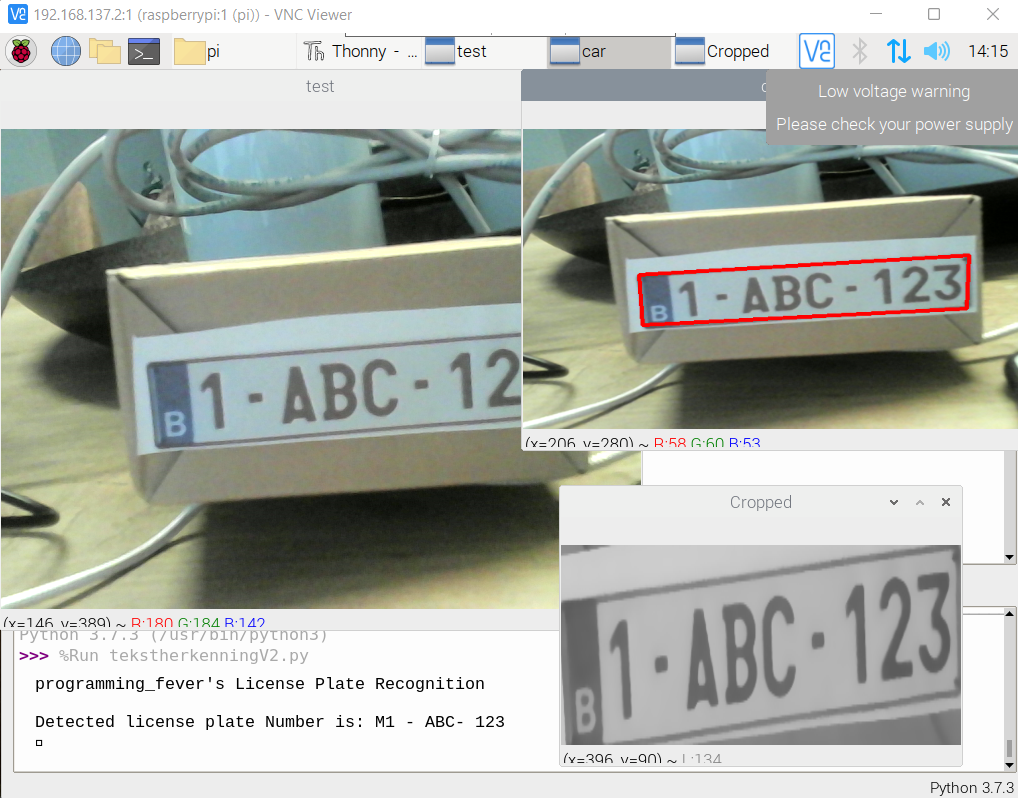
### Nummerplaatherkenning.

De nummerplaatherkenning zal dienen voor het optisch encoderen van de nummerplaatgegevens naar karakters die in de database gezet zullen worden. Zo kunnen we later met die gegevens de wagen factureren voor de tijdspanne dat hij geparkeerd stond.

#### Werking

We gebruiken hiervoor natuurlijk als hoofdbestandsdeel de raspberry pi. De beelden worden gemaakt met een HD-usb camera. Met de software OpenCV en Tesseract sturen we een algoritme op het beeldmateriaal af die teksten kan aflezen en deze terugstuurt als een string. Deze string gaan we achteraf in onze database zetten.

Op de volgende foto van onze testopstelling is het resultaat duidelijk te zien:



#### De code

De herkenningssoftware met bijhorende code was op het internet slechts te vinden voor de dedicated raspberry pi camera. Daarom moest ik een groot deel van de code aanpassen zodat deze zou werken bij een usb-camera.

def scanNummerplaat():

    #while loop om scannen evt te herhalen

    i = 0

    while i < 1:

        #camera inlezen

        ret,frame=cam.read()

        time.sleep(0.5)

        img=frame

        #foto nummerplaat vorm herkennnen

        img = cv2.resize(img, (600,400) )

        gray = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY)

        gray = cv2.bilateralFilter(gray, 13, 15, 15)

        edged = cv2.Canny(gray, 30, 200)

        contours = cv2.findContours(edged.copy(), cv2.RETR\_TREE, cv2.CHAIN\_APPROX\_SIMPLE)

        contours = imutils.grab\_contours(contours)

        contours = sorted(contours, key = cv2.contourArea, reverse = True)[:10]

        screenCnt = None

        for c in contours:

            peri = cv2.arcLength(c, True)

            approx = cv2.approxPolyDP(c, 0.018 \* peri, True)

            if len(approx) == 4:

                screenCnt = approx

                break

        #als er geen nummerplaat is herkent

        if screenCnt is None:

            detected = 0

            print ('No contour detected')

        else:

            detected = 1

        #nummerplaatcontouren tekeken

        if detected == 1:

            cv2.drawContours(img, [screenCnt], -1, (0, 0, 255), 3)

        mask = np.zeros(gray.shape,np.uint8)

        new\_image = cv2.drawContours(mask,[screenCnt],0,255,-1,)

        new\_image = cv2.bitwise\_and(img,img,mask=mask)

        (x, y) = np.where(mask == 255)

        (topx, topy) = (np.min(x), np.min(y))

        (bottomx, bottomy) = (np.max(x), np.max(y))

        Cropped = gray[topx:bottomx+1, topy:bottomy+1]

        #tekst van nummerplaat herkennen

        text = pytesseract.image\_to\_string(Cropped, config='--psm 11')

        print ('License Plate Recognition\n')

        print('Detected license plate Number is:',text,'\n')

        img = cv2.resize(img,(500,300))

        Cropped = cv2.resize(Cropped,(400,200))

        #foto's tonen om te debuggen:

        #cv2.imshow('car',img)

        #cv2.imshow('Cropped',Cropped)

        #foto schrijven naar opslag met tijdstip in de naam

        img\_name = 'numberplate\_image\_{}.png'.format(tijdstip)

        cv2.imwrite(img\_name, frame)

        time.sleep(2)

        i=i+1

    return text

### Remote webinterface.

Voor deze opdracht moeten we een “remote webinterface” voorzien voor de admin (owner) van de parkeergarage. De admin kan hierop de status bekijken van bezoekers van de parkeergarage, als ook extra informatie.

Wij hebben gekozen voor een algemene **apache2** webinterface, met een **php** uitvoering om actief data op te halen uit een **mariaDB database**.

Buiten het feit dat we de website lokaal hosten op de raspberry pi, kan deze in allerlei vormen online worden voorzien via webhosting. Voor deze opdracht laten we de website lokaal, toegankelijk via apache2. Zo kan je steeds via het **ip adres van de raspberry pi** terecht komen op onze aanmeld pagina.

Een **extra toepassing of toegankelijkheid** kan gebeuren op verschillende manieren. **Toegang tot Raspberry Pi:**

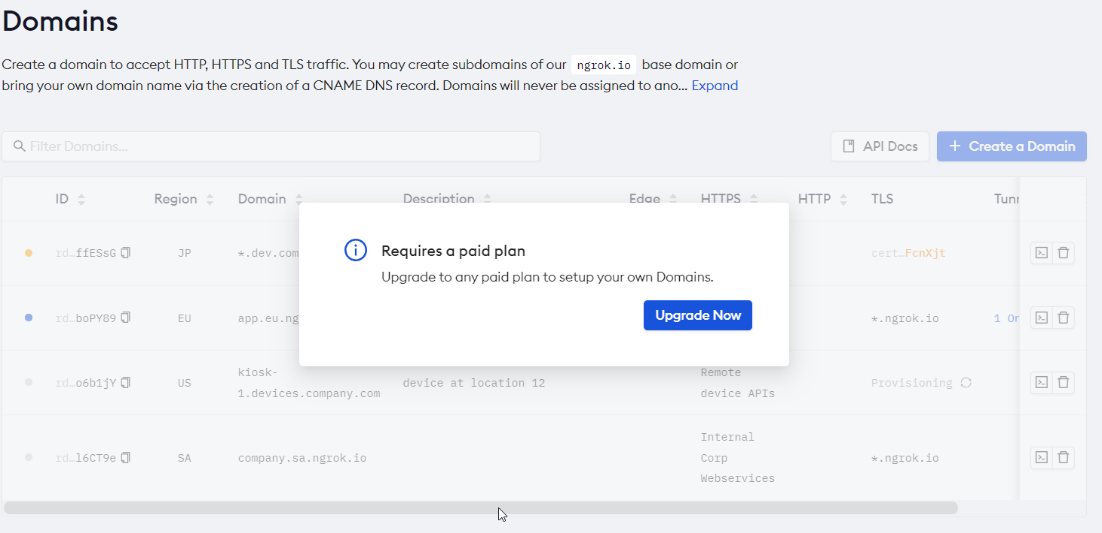
* Via **VNC Viewer:** met deze tool kan je steeds connectie maken naar je raspberry pi om het een en ander na te kijken.
* Via **NGROK:** deze tool gaan we verder gebruiken in dit hoofdstuk – al is het in dit project eerder primitief – het zal de webpagina “remote” toegankelijk maken (apache2), als ook de raspberry pi. Win=win, je hebt toegang tot een webinterface als ook rechtstreeks tot de Pi.

#### NGROK

NGROK is een gratis tool (of betalend) om toegang te krijgen tot jouw toestellen – zij het via port 80 (HTTP), 443 (HTTPS) (webinterface) of ook port 22 (SSH). Zo kan je zowel rechtstreeks als onrechtstreeks aan de informatie voor onze parkeergarage, als ook aan de raspberry pi in noodgevallen.

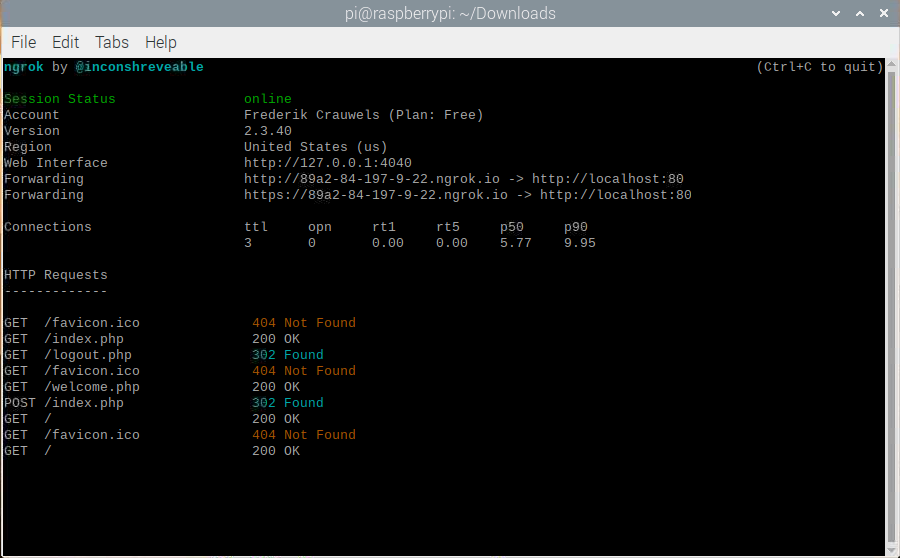
Alhoewel de apache2 client voldoende is, krijg je hier nog een extra stap om een beveiligde website te testen (HTTPS).

Het **nadeel van NGROK:** de gratis versie geeft elke keer opnieuw een random URL. Deze is ook heel “vaag” waardoor deze er niet netjes uit ziet. Alsnog bied NGROK deze mogelijkheid aan om een vaste, statische URL aan te bieden mits betaling:

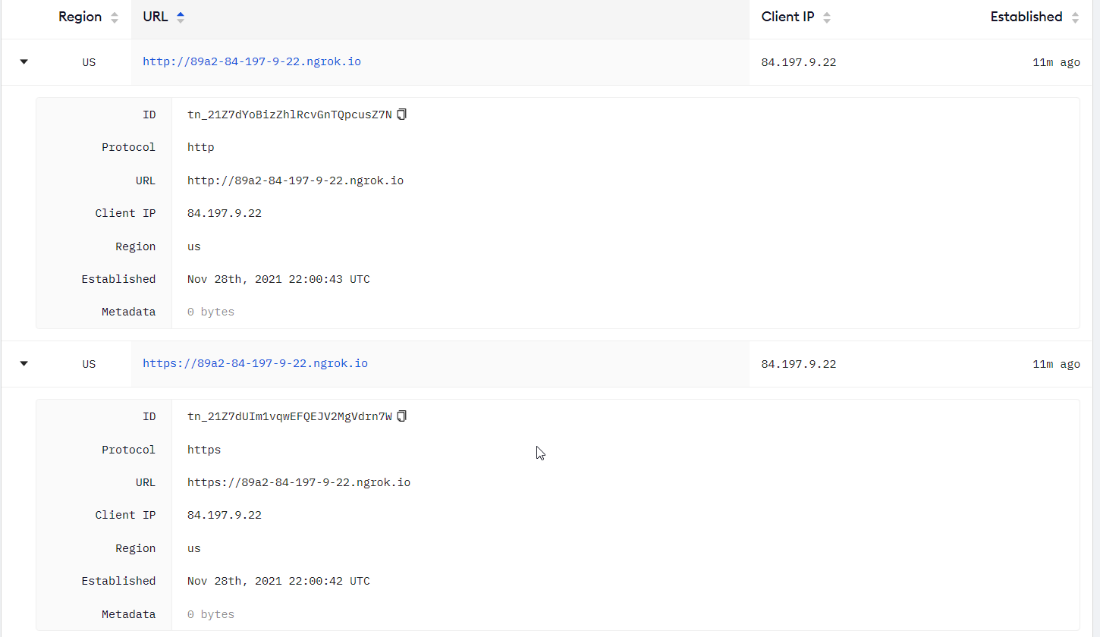


Voor deze opdracht is dit eerder irrelevant omdat het niets veranderd aan onze uitwerking, buiten een “propere” link.

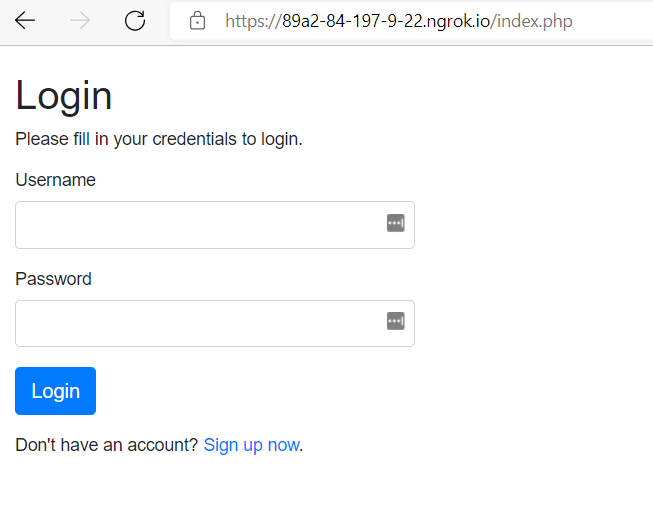
Als we NGROK dan opstarten met de nodige instellingen krijgen we dit te zien:



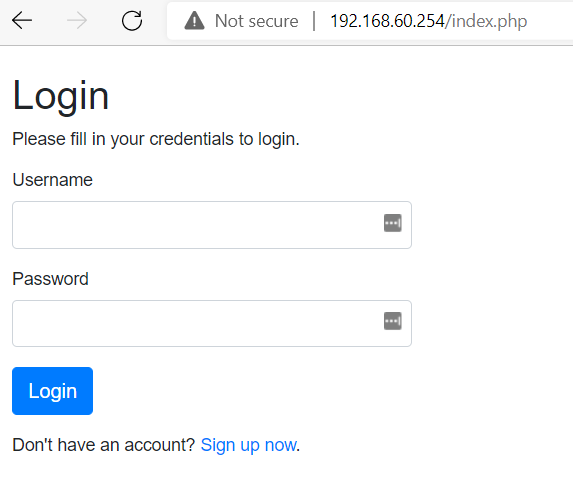
De locatie kan je ook kiezen – in mijn geval koos NGROK steeds voor een “US” link. Als we dan verder kijken op de website van NGROK krijg je dezelfde details:



Zo creëren we een directe tunnel naar onze raspberry pi om allerlei toepassingen te testen en uit te voeren. In ons geval wordt dit een remote webinterface die je nu overal kan bereiken:



Uiteraard om het hoofdstuk “NGROK” verder af te sluiten, krijg je exact dezelfde pagina zoals die via localhost. Hier hebben we wel géén keuze uit HTTPS als extra beveiliging / test:

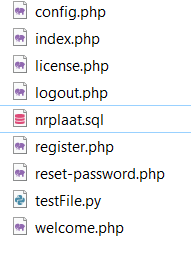


Hiermee willen we graag aantonen dat we ook verder hebben gekeken dan enkel apache2. De optie via NGROK geeft als extra dat je deze ook al kan hosten online en toegankelijk maken voor iedereen. Voor onze opdracht blijft het relatief hetzelfde; via NGROK of via apache2 lokaal. Het extra voordeel blijft vooral dat je als owner ook extra toegang hebt met SSH naar je raspberry pi; voor testing en toegang “indien er iets fout loopt”. NGROK bied ook nog een tal van extra voordelen aan die in de toekomst veel opties bieden:

* IP restrictions & password protection: je kan de webpagina extra beveiligen met een extra aanmeld pagina – specifiek om aan de webpagina te geraken (dus niet in-browser). Daarnaast kan je ook IP-restricties toepassen – bv om je eigen parkeergarage WiFi enkel te laten toestaan om de webpagina te betreden.
* SSH, TLS certificaten, APIs, “Events”,…: extra beveiliging en zelfs API toegang voor bepaalde doeleinden zijn mogelijk in de toekomst. Events kunnen ook “in de cloud” aangestuurd worden om zo bv. Login log informatie bij te houden (wanneer meld iemand zich aan, met welk IP,…). Veel opties dus, waar wij nu jammer genoeg geen toegang tot hebben (beta of betalend).

#### Remote webinterface

Voor onze remote webinterface hebben we gekozen voor een **PHP website / uitvoering**. PHP biedt de mogelijkheid om consistent nieuwe data aan te leveren terwijl die wordt verstuurd via een python programma. Vandaar ziet onze webinterface er zo uit:

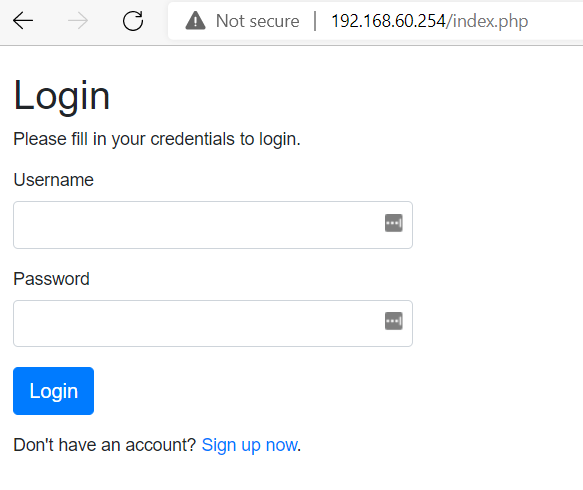


* **Index.php:** dit is de “startpagina” zoals iedereen deze wel kent. De “home” pagina waarop je als eerst komt. Hier zie je een “login” pagina staan, en indien je nog geen login hebt kan je deze aanmaken met de link onderaan.
* **Config.php:** deze file dient voor de basisgegevens aan te leveren. Dit gaat voornamelijk over de instellingen voor de database, bijvoorbeeld.
* **Nrplaat.sql:** deze SQL file staat ten dienste van onze database (zie vorige hoofdstuk). Met deze file kunnen we automatisch onze gehele setup opzetten in PHPmyadmin.
* **Welcome.php:** Je zal na het aanmelden op deze pagina verschijnen. Hier staan een tal van informatie voor de gebruiker. Afhankelijk van welke login (admin of “user”) zal je een ander layout en gegevens te zien krijgen.
* **Register.php:** deze pagina dient om je te registeren indien je nog geen account hebt. Username + password is voldoende! Er zijn validatieregels voorzien die de lengte van een wachtwoord bepalen.
* **License.php:** op deze pagina ga je jouw persoonlijke account linken met een nummerplaat die geregistreerd is via onze nummerplaatherkenning. Als jouw nummerplaat in onze DB voorkomt zullen de gegevens ingevuld worden en krijg je deze ook te zien op welcome.php.
* **Reset-password.php:** wachtwoord vergeten? Geen probleem! Je kan deze makkelijk en snel zelf wijzigen – zonder tussenkomst van admin.
* **Logout.php:** een standaard doorverwijspagina die, indien je op welcome.php (of andere pagina) wilt afmelden – je automatisch doorstuurt naar index.php. Ook indien een gebruiker een andere pagina wil betreden zal dit script in werking treden om de gebruikers terug naar index.php te verwijzen.
* **PHPmyadmin** (niet zichtbaar): een algemene admin pagina voor onze database. Je kan de database op twee manieren bekijken: de gegevens staan (voor admin) op de welcome.php als ook in PHPmyadmin. Voor specifieke uitwerkingen (resetten van ww, veranderen van records) zal je steeds hier terecht moeten. Om alle gegevens te bekijken van gebruikers kan je de welcome.php pagina gebruiken.

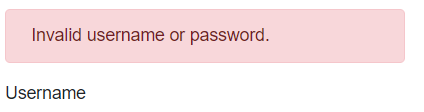
De code voor elke pagina apart zal te zien zijn helemaal achteraan. De volgende hoofdstukken focussen op wat je te zien zal krijgen op elke pagina.

#### Index.php

De standaard login pagina, zoals je deze al gezien hebt voordien.

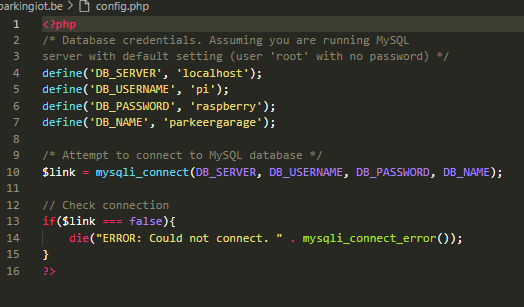


Aanmelden met username en password. Heb je deze niet? Klik op “Sign up now” om te registeren – foute gegevens? Dan laat de browser je niet door!



#### Config.php

Basis configuratie voor onze database.



#### Nrplaat.sql

Basis configuratie voor onze database (MariaDB op raspberry pi lokaal).

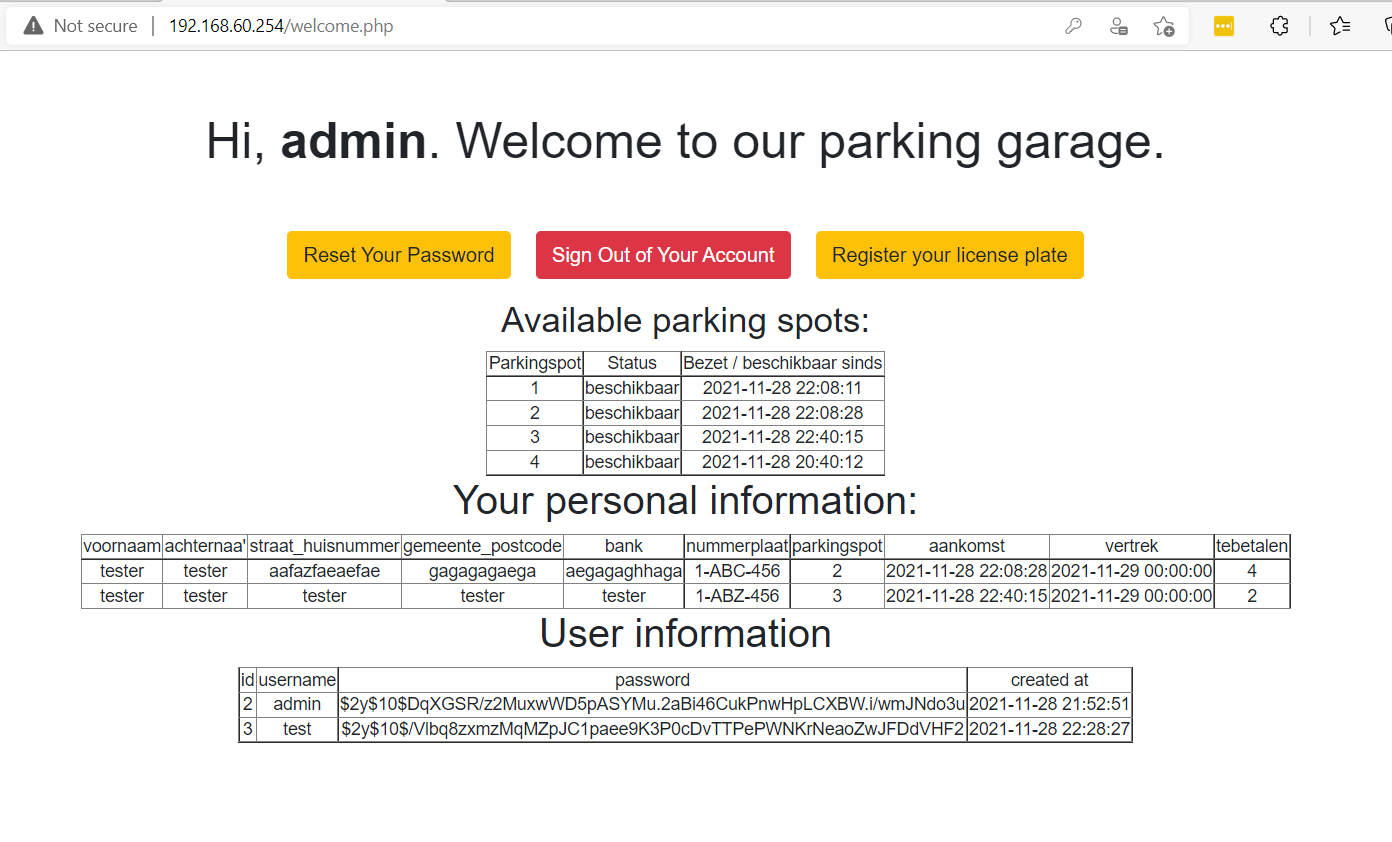


In totaal tels onze database “parkeergarage” 3 tabellen: users om geregistreerde gebruikers te bekijken, nrplaat voor algemene gegevens per gebruiker en parkingspot om de beschikbaarheid van parkeerplaatsen weer te geven.

#### Welcome.php

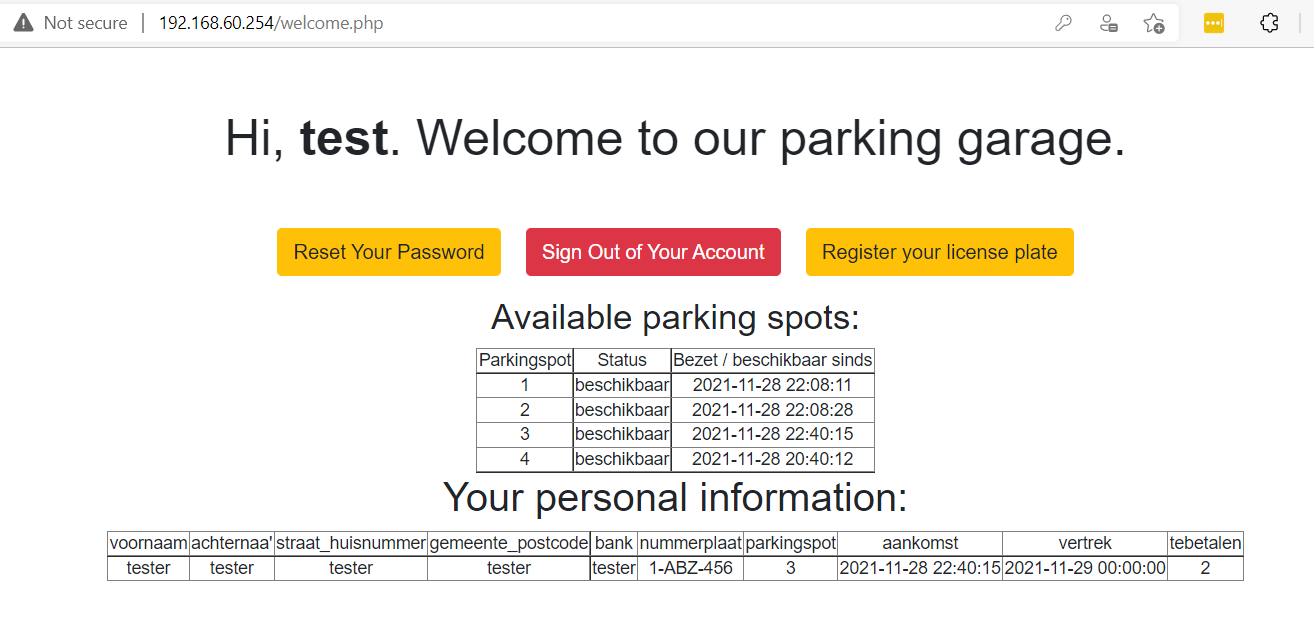
Dit is de pagina waar je op terecht komt indien je succesvol bent aangemeld. Je krijgt hier **twee soorten pagina’s te zien:** een ADMIN pagina, of een gebruiker pagina.

**ADMIN:** krijg je alleen te zien indien je als gebruikersnaam “admin” aanmeldt.



Alleen de admin kan alle gegevens zien van gebruikers. Zowel persoonlijke als aanmeldgegevens. Wachtwoorden zijn nooit zichtbaar – deze worden tijdens het aanmelden geëncrypteerd zodat ook de admin niet zomaar alles kan zien. Je ziet hier duidelijk alle gegevens die je moet weten van de parking, als ook van de gebruikers.

**USER:** users kunnen enkel hun persoonlijke gegevens bekijken. De gegevens voor andere gebruikers, als ook de “user information” tabel zijn niet zichtbaar.

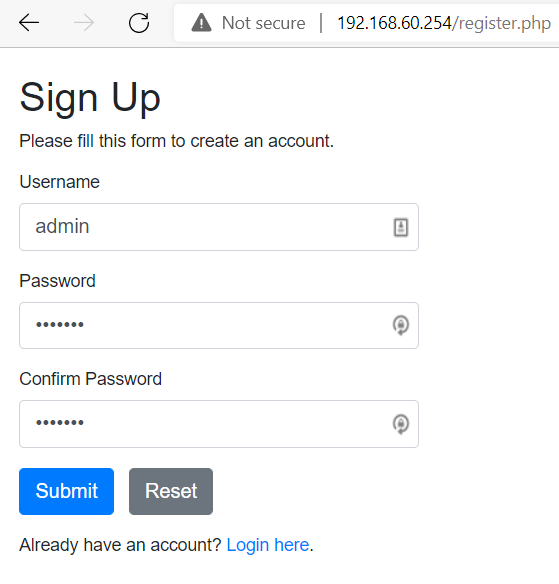


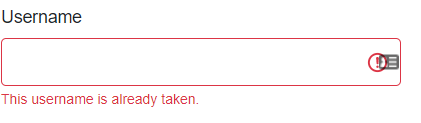
Enkel jouw eigen gegevens (indien gevalideerd via “register your license plate” kan de gebruiker zien.

Reset your password is om je wachtwoord te veranderen. Sign out om af te melden en register your license plate om jouw aanmeldgegevens (gebruikersnaam) te linken aan jouw nummerplaat gegevens en extra persoonlijke informatie.

#### Register.php

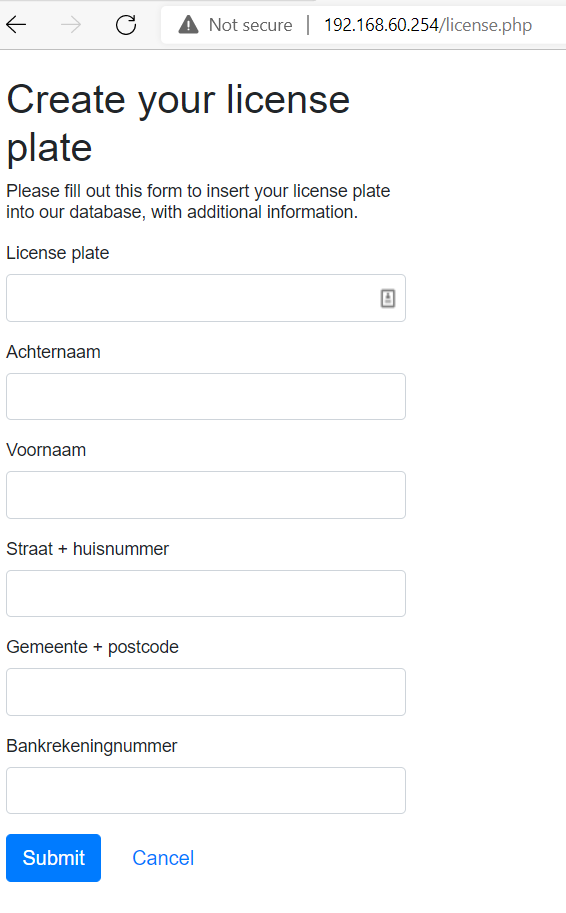
Op deze pagina kan een nieuwe gebruiker zich aanmelden. Er wordt telkens gevalideerd of er al een soortgelijke gebruikersnaam in onze database zit.





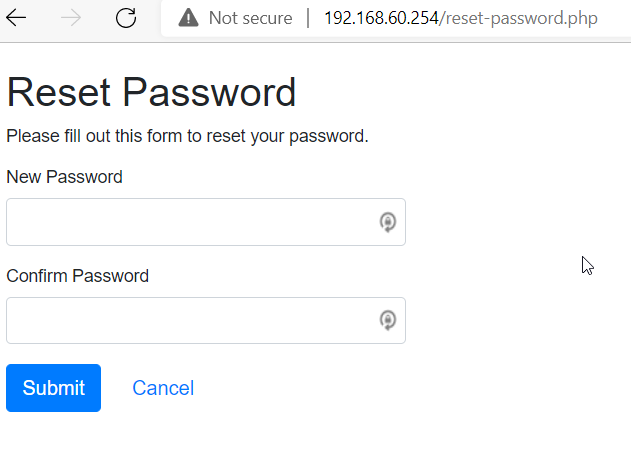
#### License.php

Op deze pagina moet de gebruiker extra informatie toevoegen. Allereerst om de nummerplaat te linken met dit account – als ook extra informatie voor de toekomst. Voor betalingen, persoonlijk adres… Zo kan je steeds een gebruiker in de “user” tabel linken met een persoon in de “nrplaat” tabel. Ook hier wordt aan gegevensvaldiatie gedaan zodat gegevens zeker lang genoeg zijn om in de tabel geplaatst te worden. De nummerplaat wordt niet nagekeken – dit zou een extra stap kunnen zijn om te zien of deze herkend wordt (zoals hierboven met username). Indien foutief, krijg je gewoon géén gegevens te zien bij welcome.php.



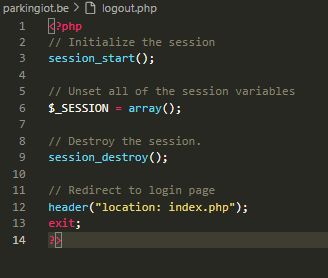
#### Reset-password.php

Deze pagina staat er om je wachtwoord te veranderen indien je het bent vergeten. Zoals eerder aangehaald zijn wachtwoorden geëncrypteerd. Een admin kan dus niet zomaar jouw wachtwoord vertellen. Alleen jij weet het – en anders pas je het hier makkelijk zelf aan.



#### Logout.php

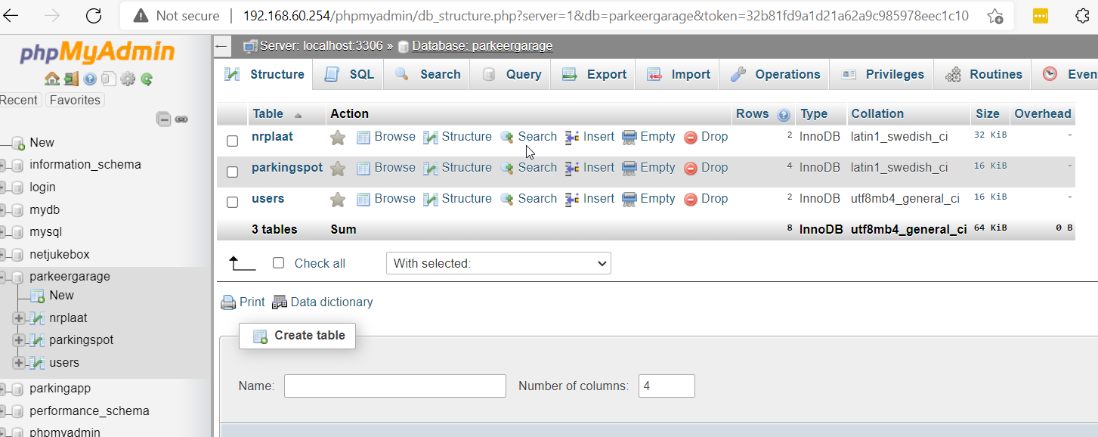
Logout.php dient enkel om door te linken naar index.php (login pagina). Telkens je “sign out” of niet juist bent aangemeld zal je door dit PHP script worden teruggestuurd naar index.php.



#### PHPmyadmin (DB)

De standaard phpmyadmin is ook beschikbaar en helemaal onmisbaar voor dit project. Wil je makkelijk de gegevens van de database nakijken? Dan gaat dit eenvoudig via deze pagina. Hier voeren we ook de SQL query door die voordien ter sprake kwam om alle gegevens automatisch te voorzien zonder al te veel manuele input. Dit is ook handig voor testing purposes of als admin om iets aan te passen – een foutieve nummerplaat, verkeerde gegevens ingevoerd / doorgestuurd / …

Zelfs met een remote webinterface kom je eigenlijk niet toe als admin. De PHPMyAdmin is hier onmisbaar en moet als tweede remote webinterface behandeld worden zodat de owner op alle vlakken in controle is.



**Extra code die nog niet te zien was in bovenstaande screenshots vind je achteraan terug voor de bronvermeldingen**.

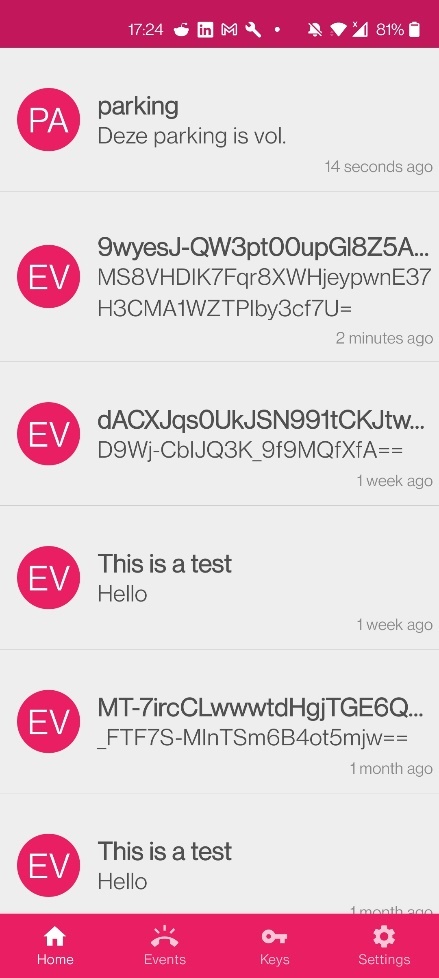
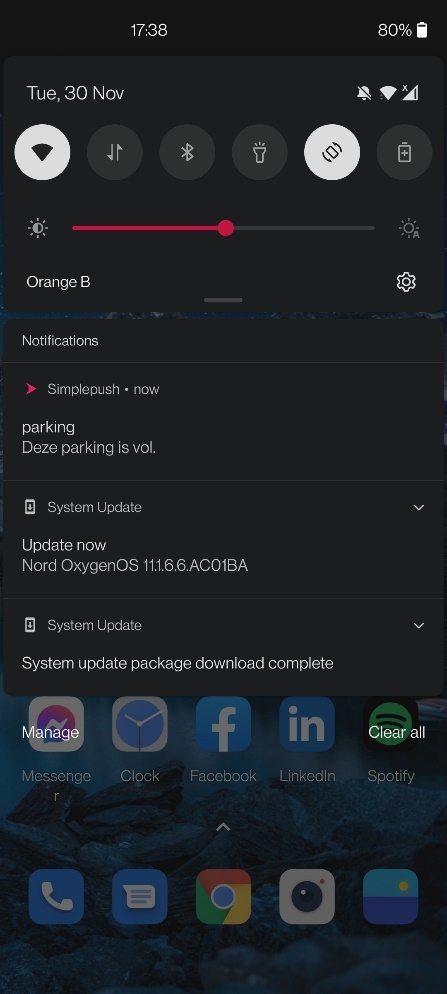
In de bronnen vind je ook het grootste deel van de basisopstelling van deze PHP pagina’s terug. De basis setup komt van een online bron – waarbij telkens op verschillende pagina’s extra’s zijn voorzien door ons. Deze extra’s gaan van: een compleet nieuw script (license.php) als ook het voorzien van data voor de gebruiker en admin (allebei aparte view). Veel gegevens worden ook doorgestuurd naar de database – die opnieuw door ons is voorzien (nrplaat en parkingspot) buiten de “users” tabel.

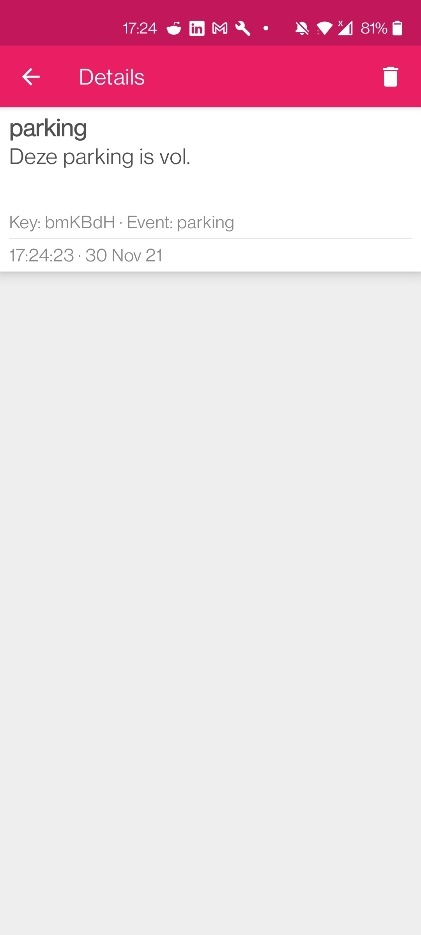
### Push message uitsturen wanneer 4 parkeerplaatsen vol zijn (LDR + Push).

LDR:

Op elke parkeerplaats in de ondergrond zit een LDR die de hoeveelheid licht meet. Wanneer er een auto op de parkeerplaats staat dat meet de LDR veel minder licht en dan weet het programma dat er een auto op die parkeerplaats staat.

Push:

De owner krijgt een push bericht indien de parking volledig vol is. Deze ziet er als volgt uit:



# Algemene uitwerking

## Groepsbespreking: minimum requirements

|  |  |
| --- | --- |
| **Parking barrier** | Ultrasoon sensor (servo, slagbboom).  Led: parkeergarage is vol. |
| **Display entrance parking spaces** | 4 led lichtjes die aangeven welke parkeerplaats vol is of niet (a.h.v. LDR). |
| **Photo USB cam en license plate analyze** | Full HD 1080p webcam  Nummerplaatherkenning. |
| **Display (webinterface) + remote access** | Lokaal (Ngrok) client kunnen bekijken.  Website / interface fixen met data uit DB.  Basis login pagina voorzien (Php MyLogin). |
| **Push message + LDR** | Licht voorzien voor parkeergarage (indien we in donkere hoek zitten).  LDR op breadboard (1x) -> in echte setup 4x (groep voorziet andere 3).  Push message kunnen aanleveren. |
| **Database** | Haal gegevens uit alle programma’s (zie GitHub). -> aanmaken github en centrale plaats voor de “programma’s”.  Test bestanden voor aanleveren en ontvangen gegevens.  Van python bestanden naar MariaDB naar PHP webclient. |

**Tussendeadline:** woensdag november 17 om 20:00. Tussentijdse update.

### Youtube link

<https://youtu.be/CzePaAIzDDs>

### Github link

[brandonmorren/parkeergarage (github.com)](https://github.com/brandonmorren/parkeergarage)

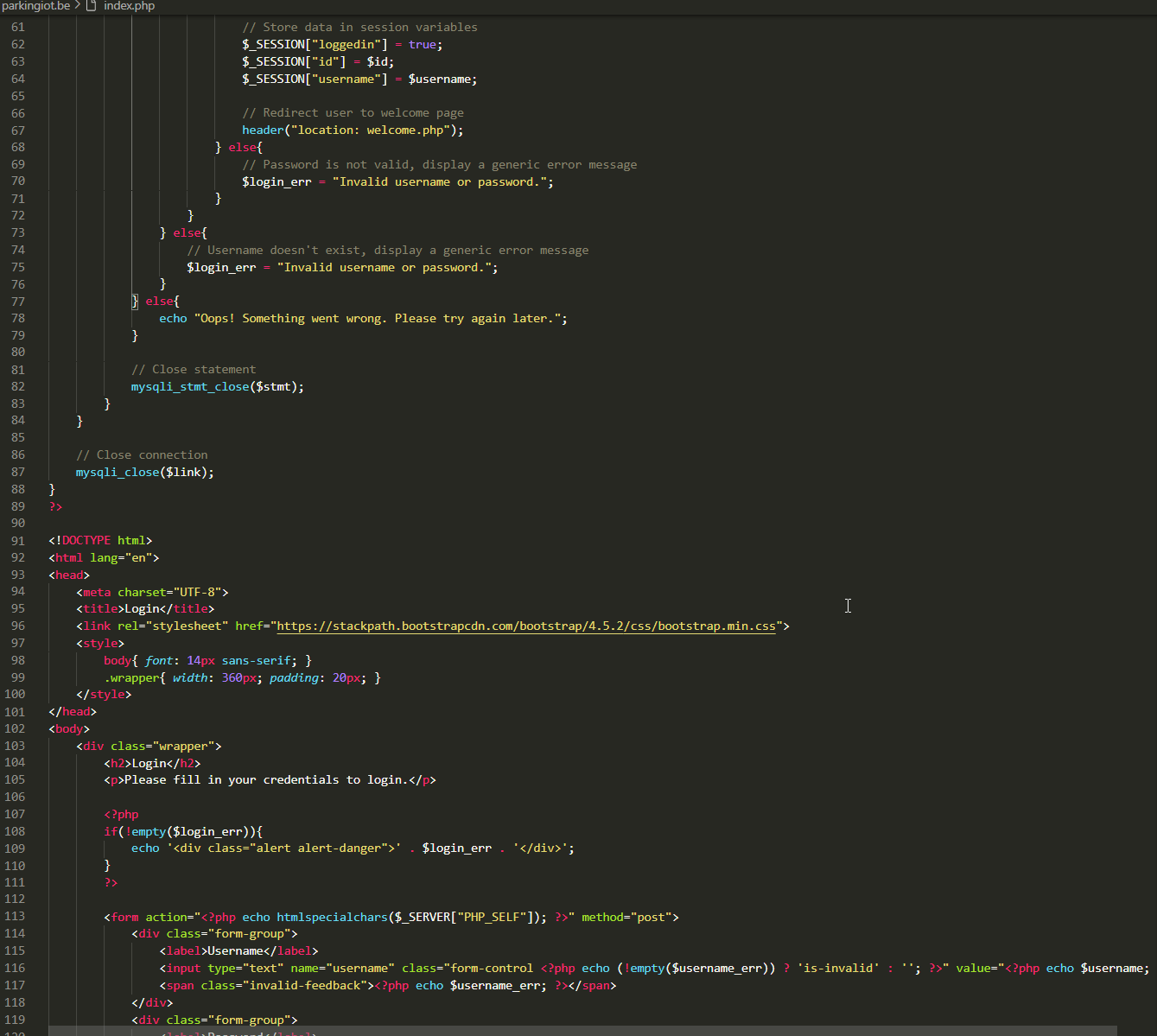
# Extra code

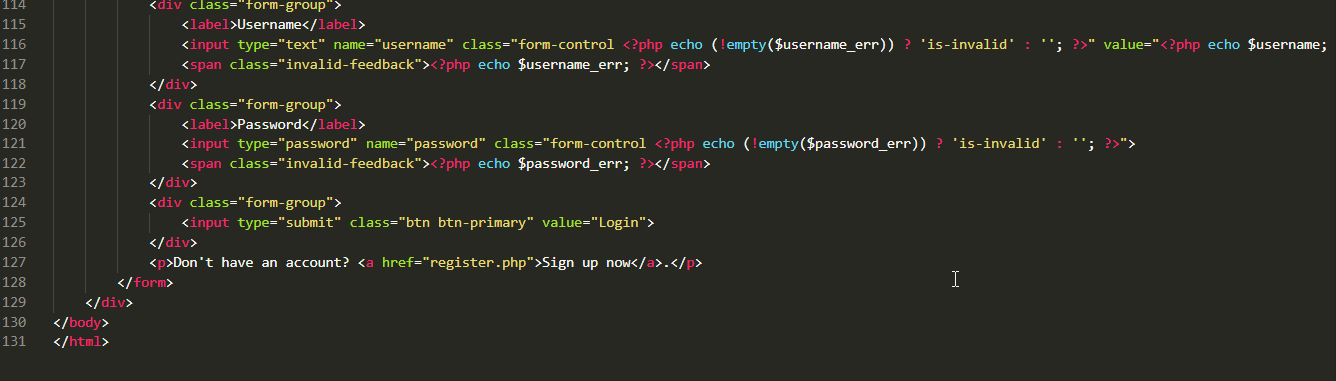
## Remote webinterface

Meer informatie per hoofdstuk kan je terugvinden in het bovenstaande hoofdstuk “remote webinterface”.

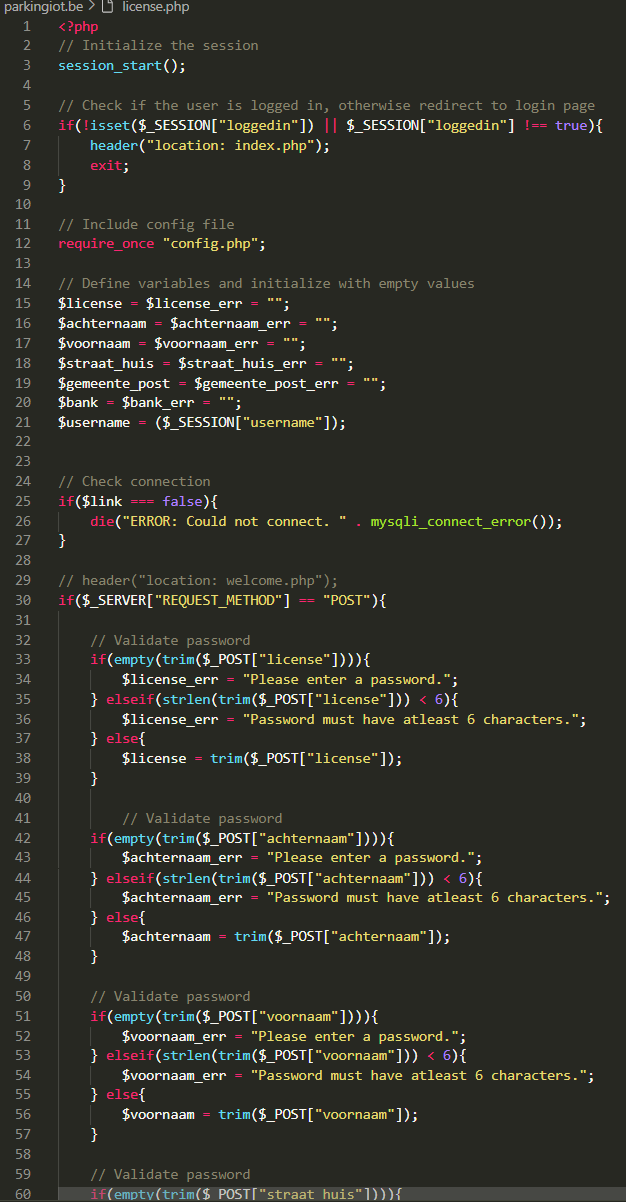
### Index.php



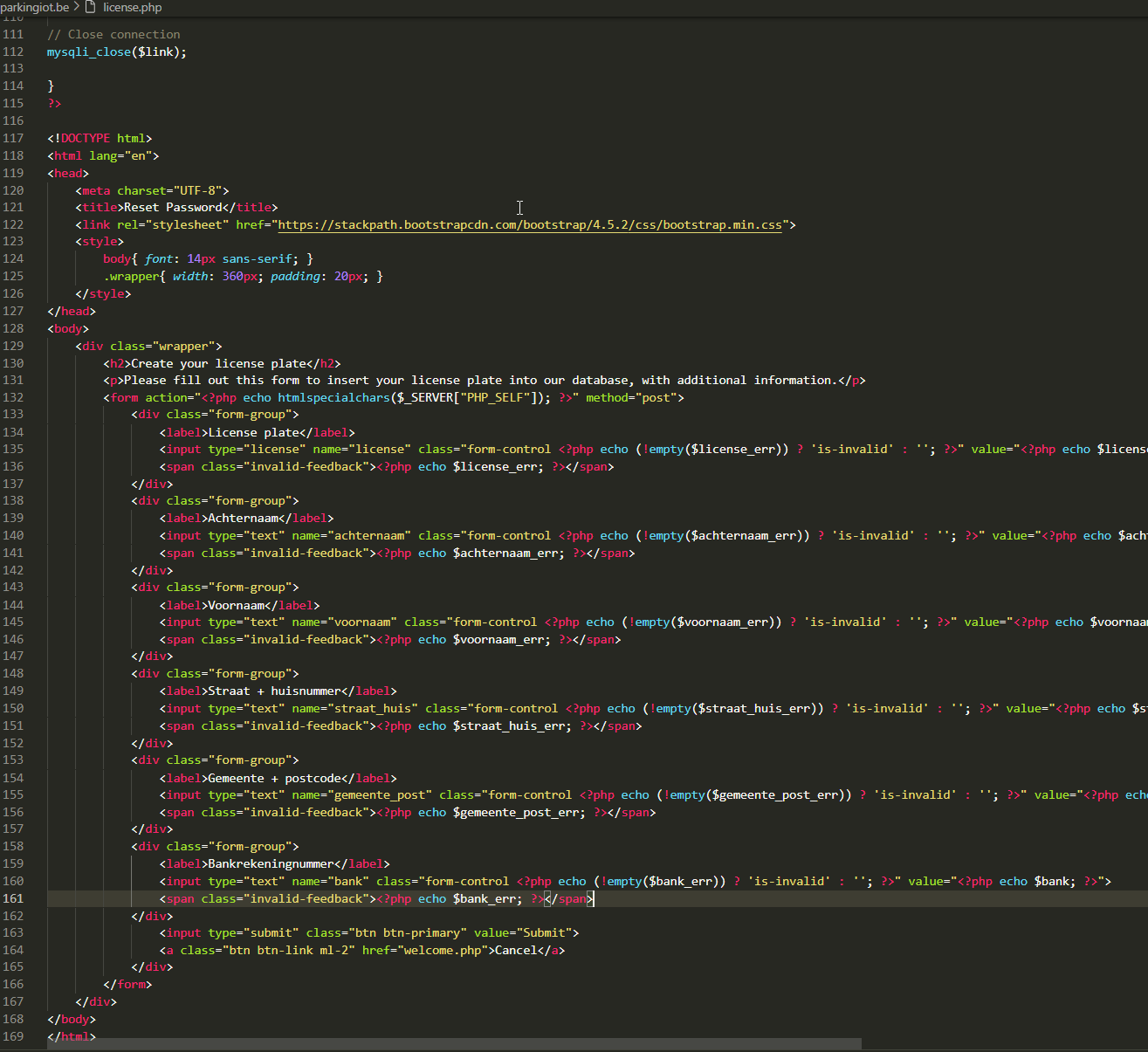




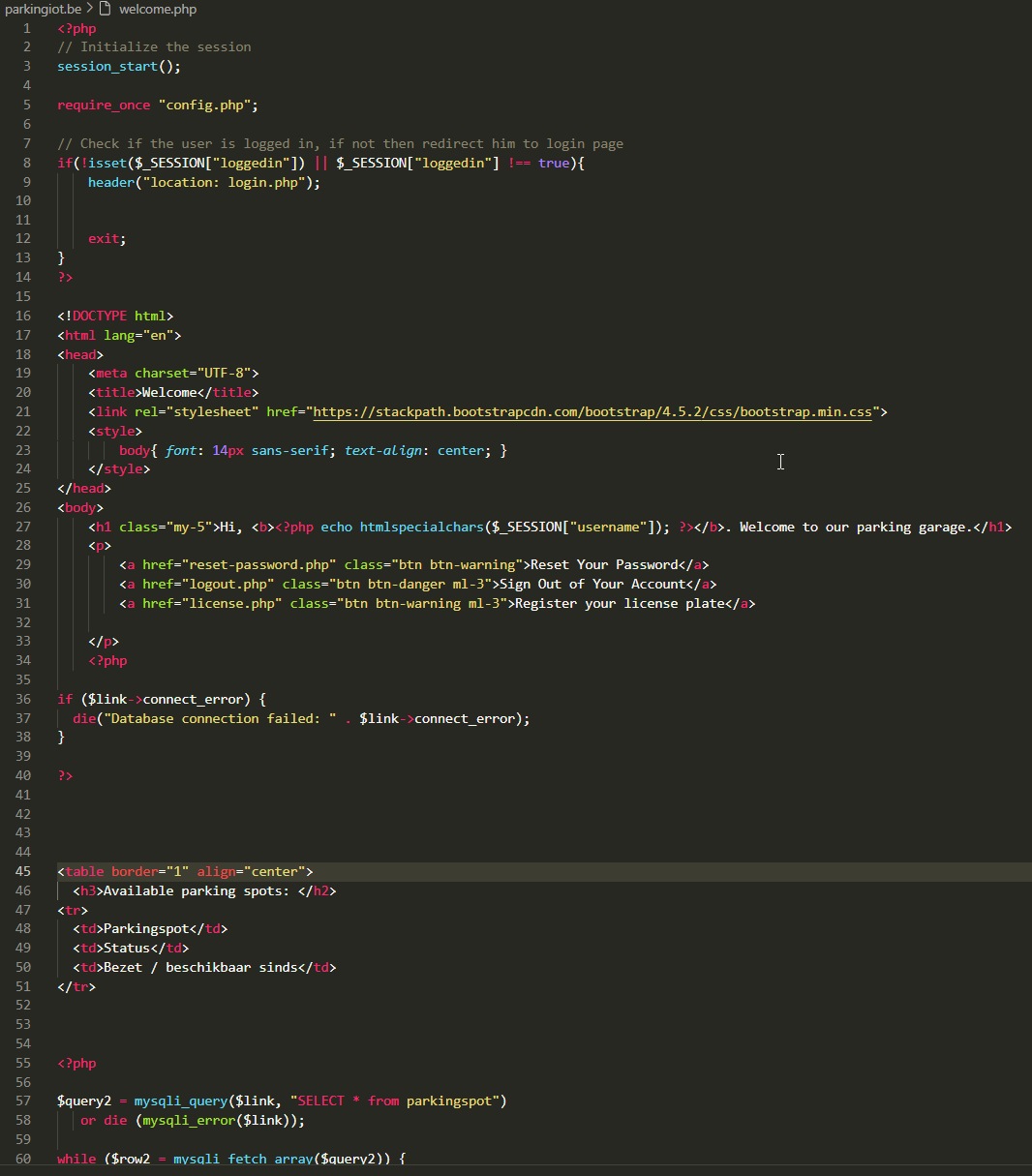
### license.php

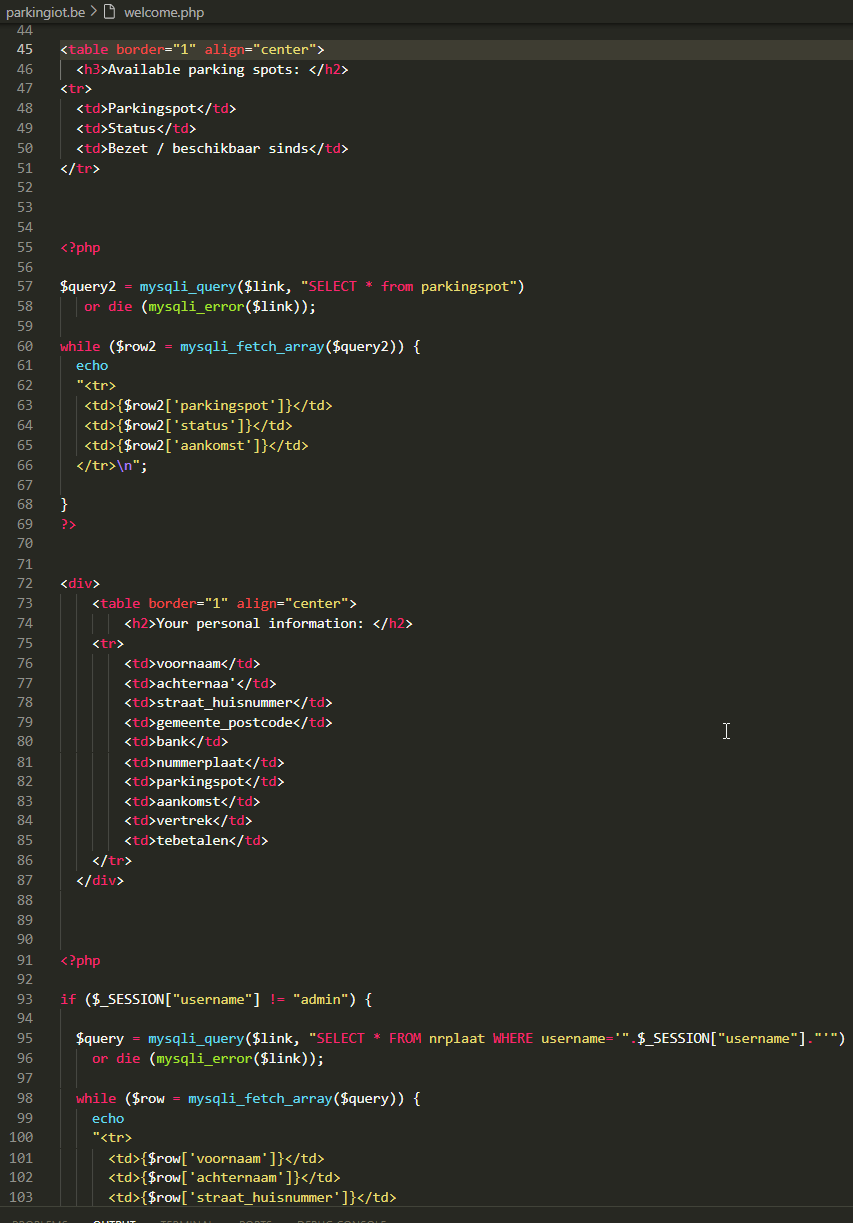


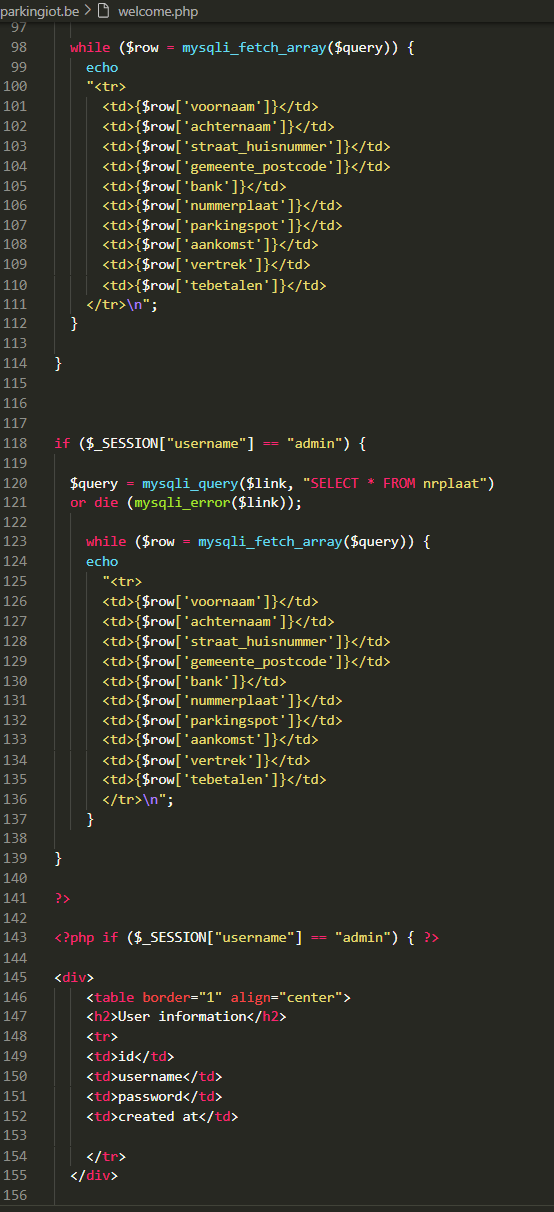


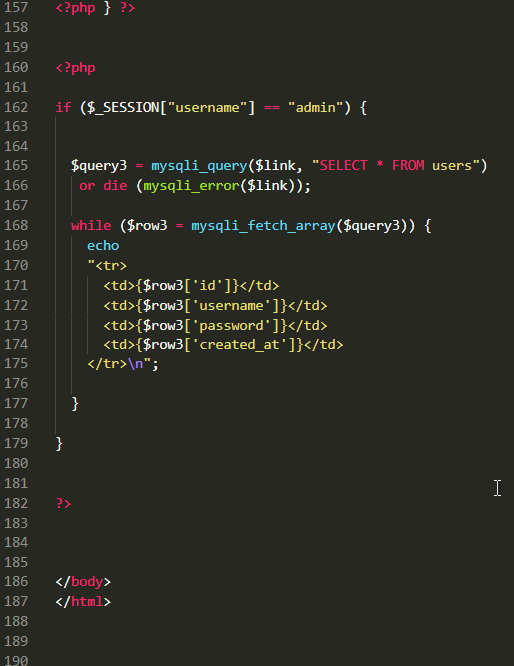


### Welcome.php









***De overige pagina’s (register.php, reset-password) blijven zo goed als onveranderd van de basis setup. Deze kan je steeds bekijken door in bibliografie te gaan kijken naar de link in verwijzing.***

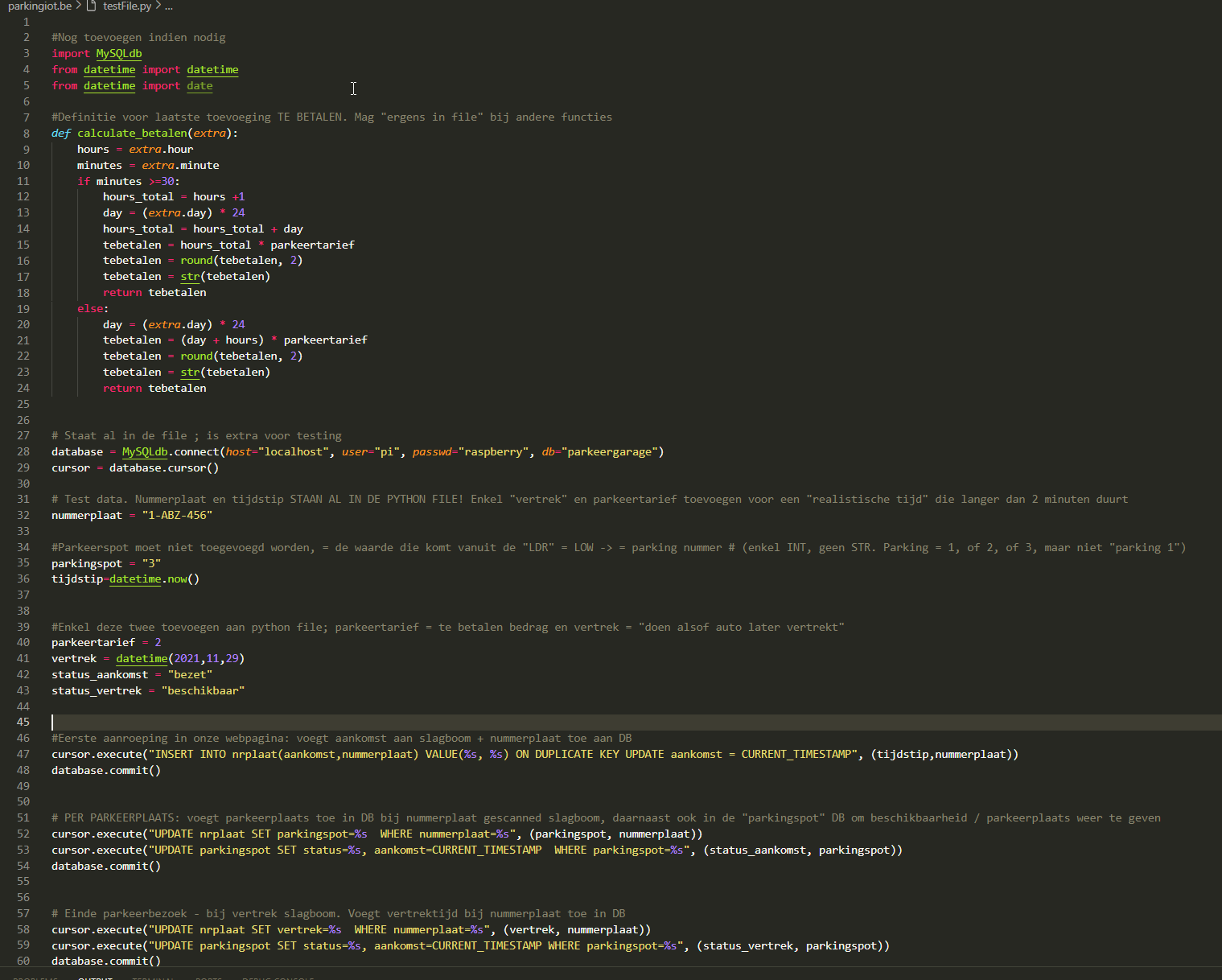
### main.py

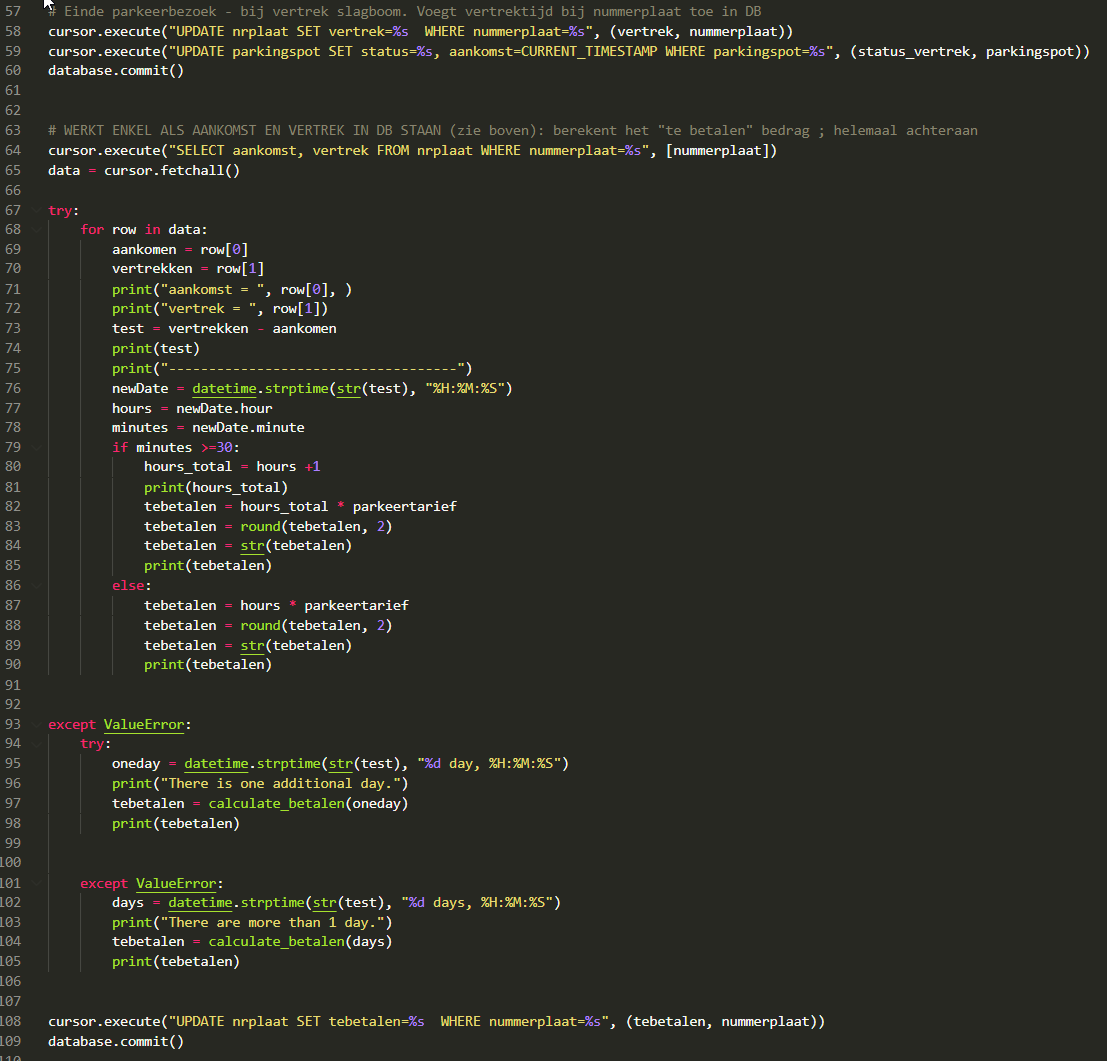
Onze algemene code van het programma met de slagboom, LDR,... kan je terugvinden via de Github link die eerder vermeld werd:

[brandonmorren/parkeergarage (github.com)](https://github.com/brandonmorren/parkeergarage)

### TestFile.py

Testfile die wordt gebruikt bij het voorzien van data van webinterface / python programma naar de database / webinterface visualisatie. Deze wordt geïntegreerd met onze main.py file (de basis voor ons project).





# Bibliografie

Docenten IoT essentials. (2021, November 2). *Chapter5\_Python3\_GPIO*. Opgehaald van IoT essentials: https://thomasmore.instructure.com/courses/15211/files/2240798?module\_item\_id=680300

*Documentation*. (sd). Opgehaald van ngrok.com: ngrok.com/docs

*licence plate recognition using openCV & python*. (sd). Opgehaald van medium.com: https://medium.com/programming-fever/license-plate-recognition-using-opencv-python-7611f85cdd6c

*OCR with OpenCV & Tesseract*. (sd). Opgehaald van maker.pro: https://maker.pro/raspberry-pi/tutorial/optical-character-recognizer-using-raspberry-pi-with-opencv-and-tesseract

*OCR with tesseract*. (sd). Opgehaald van nanonets.com: https://nanonets.com/blog/ocr-with-tesseract/

*php mysql login system*. (sd). Opgehaald van tutorialrepublic.com: https://www.tutorialrepublic.com/php-tutorial/php-mysql-login-system.php

*Python how to capture image from wabcam*. (sd). Opgehaald van stackoverflow.com: https://stackoverflow.com/questions/34588464/python-how-to-capture-image-from-webcam-on-click-using-opencv/34588758#34588758

Raspberry Pi Tutorials. (2021, November 2). *Raspberry Pi Servo Motot control*. Opgehaald van Raspberry Pi Tutorials: https://tutorials-raspberrypi.com/raspberry-pi-servo-motor-control/