**Verslag**

IOT LED / TEMPRATUUR – PHP REST API

Laravel – php

Jesse Helmes

2IAO3B

Inhoud

[Beschrijving opstelling 3](#_Toc56180201)

[Gebruikte hardware 4](#_Toc56180202)

[Hoe te gebruiken 5](#_Toc56180203)

[Uitleg code met snippets 6](#_Toc56180204)

[Api routings 6](#_Toc56180205)

[Omgeving tempratuur ophalen 7](#_Toc56180206)

[Cpu tempratuur ophalen 8](#_Toc56180207)

[Tempratuur model 8](#_Toc56180208)

[Leds ophalen 10](#_Toc56180209)

[Leds aansturen 11](#_Toc56180210)

[Led morse code. 12](#_Toc56180211)

[Front end 14](#_Toc56180212)

[Gegevens ophalen op load 14](#_Toc56180213)

[Json short get method 15](#_Toc56180214)

[Tempratuur geschiedenis lijst 15](#_Toc56180215)

[Setleds functie 15](#_Toc56180216)

[Led action 16](#_Toc56180217)

[Updaten van de led status kleur 17](#_Toc56180218)

[Cpu tempratuur 17](#_Toc56180219)

[Omgeving tempratuur 18](#_Toc56180220)

[Morse code 19](#_Toc56180221)

[Dark mode 19](#_Toc56180222)

[Interface 21](#_Toc56180223)

[21](#_Toc56180224)

[Cpu tempratuur geschiedenis 9](#_Toc56180225)

[Reflectie 22](#_Toc56180226)

[Links en commands die me hielpen tijdens dit project. 22](#_Toc56180227)

# Beschrijving opstelling

Mijn project is het lezen van tempratuur intern en extern. leds die je uit en aan kunt zetten en ook in morse code kunt laten knipperen.

De tempraturen worden elke 30 seconde refreshed en je kunt de geschiedenis van de cpu tempratuur ophalen.

Ik heb de api in de framework laravel gedaan en alleen daarvoor heb ik alleen php voor gebruikt.

de grafische kant heb ik met HTML, CSS en JQUERY gedaan. Dit draait op mijn eigen laptop hierin kun je de leds los van elkaar uit en aan zetten, morse code versturen, de tempraturen zien en de geschiedenis van de CPU temratuur ophalen.

# Gebruikte hardware

in dit project heb ik de volgende hardware onderdelen gebruit:

•Raspberry Pi 3 Model B+

•WiringPi prototype board

•breadboard

•DS18B20 temperatuur sensor

•1x rode LED

•1x groene LED

•Aantal weerstanden (2x 220, 4k7 Ohm)

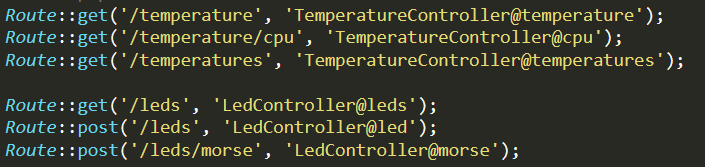
# Hoe te gebruiken

Je moet eerst met de raspberry pi verbinden en inloggen dan ga je naar de map api met de command 'cd /var/www/api' dan doe je sudo php artisan serve --host=<ip adres> voor mij was dit 192.168.178.86 vergeet niet om ‘sudo’ te doen want anders heb je geen permissie voor de leds.

Je kunt de api afsluiten door de ‘Ctrl + c’ te doen in de terminal.

# Uitleg code met snippets

## Api routings

Dit verbind de url met de juist class en functie 

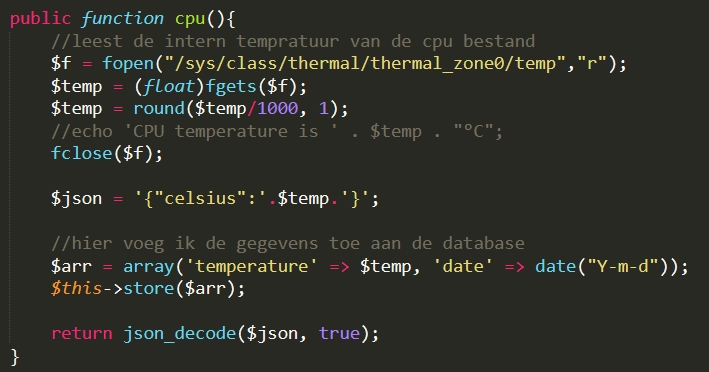
## Omgeving tempratuur ophalen

In deze functie werd de tempratuur van de omgeving op gevraagt (DS18B20 temperatuur sensor )



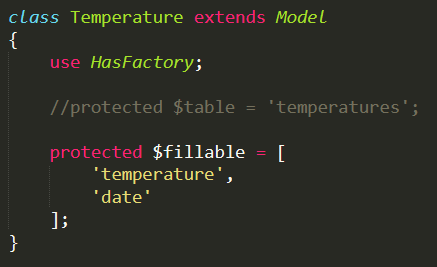
## Cpu tempratuur ophalen

Hier word te tempratuur van de cpu opgehaald en in de database gestopt



## Tempratuur model

Dit is de tempratuur model die gebruikt werd voor het opslaan en ophalen van de geschiedenis.



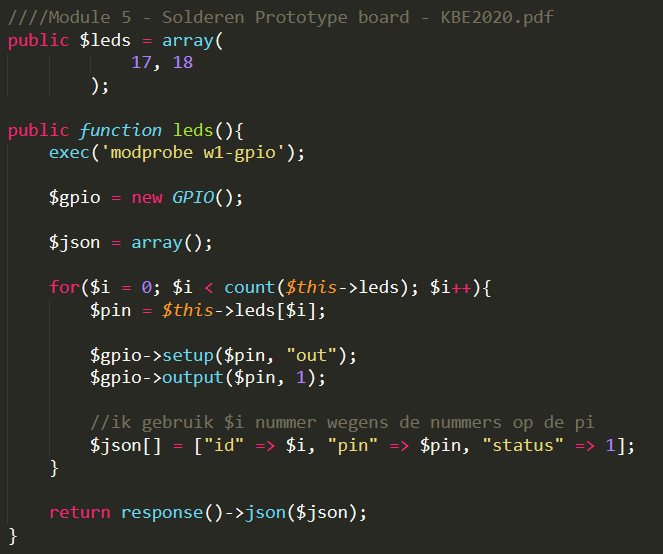
## Cpu tempratuur geschiedenis

Deze functie haalt de tempratuur geschiedenis op van de cpu



## Leds ophalen

Deze functie haalde alle pins op die je kon besturen wat bij deze project leds waren en boven in is de array met pins van de raspberry pi zelf die je nodig hebt om de goede pin aan te sturen. In deze geval nummer 0 en 1 op de WiringPi prototype board.



## Leds aansturen

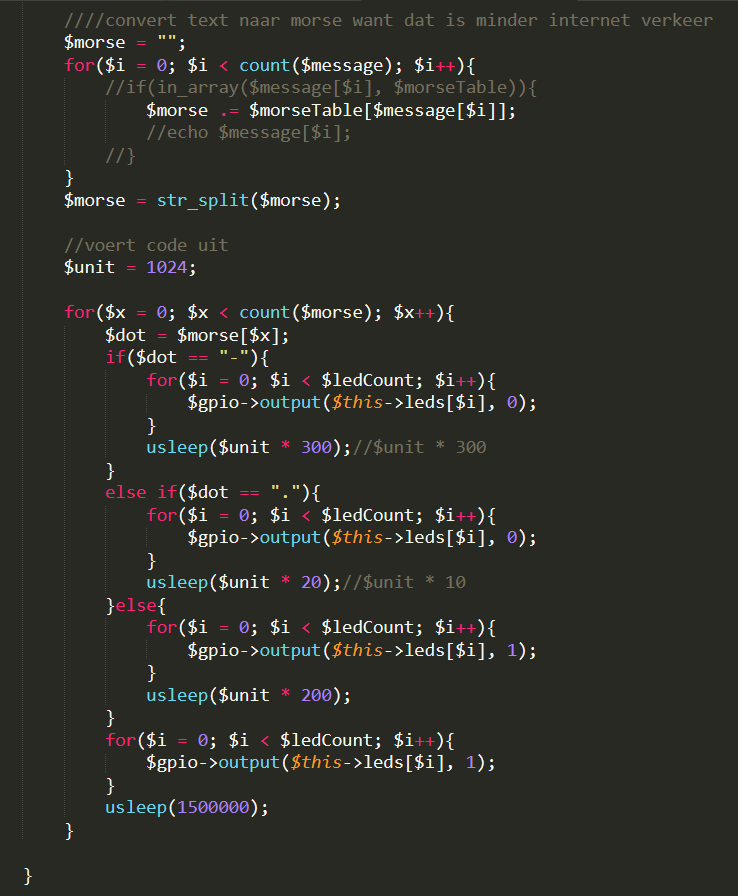
In deze functie kun je met de in komende json post de leds aansturen let wel op 1 is uit en 0 is aan. Het pakt namelijk de id van de json en de status en dan gevolgens kijkt die bij welke pin de id hoort en gebruikt dat om de pin aan te sturen.



## Led morse code.

Dit was een extra idee waar bij de tekst die je mee geeft word omgezet naar morse code en dan daar heen werd geloopt om alle leds aan en uit te zetten, de tijd dat ze aan bleven hing af van de lengte van de streepje.





# Front end

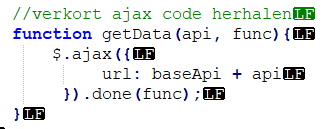
## Gegevens ophalen op load

Hier werden alle gegevens opgehaald op load en werden de hoofd html items die aangepast werden aangemaakt voor een const, om de 30 seconde werd de tempraturen vernieuwd. Ook heb ik hier een dark mode aan toegevoegd met een tijd en mijn naam omdat ik van dark mode hou.



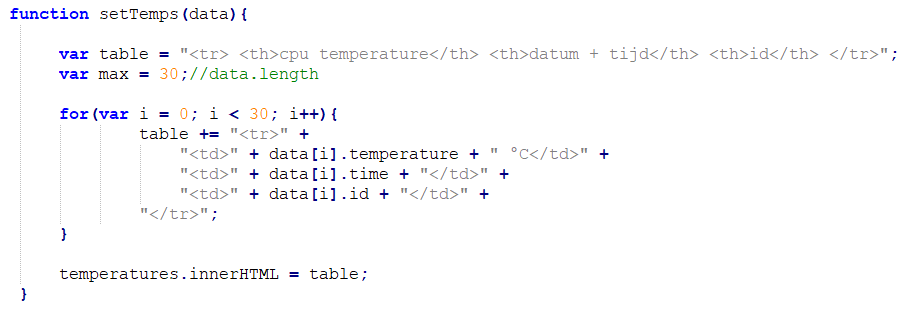
## Json short get method

Zoals de comments al zegt het verkort de code gebruik in de project



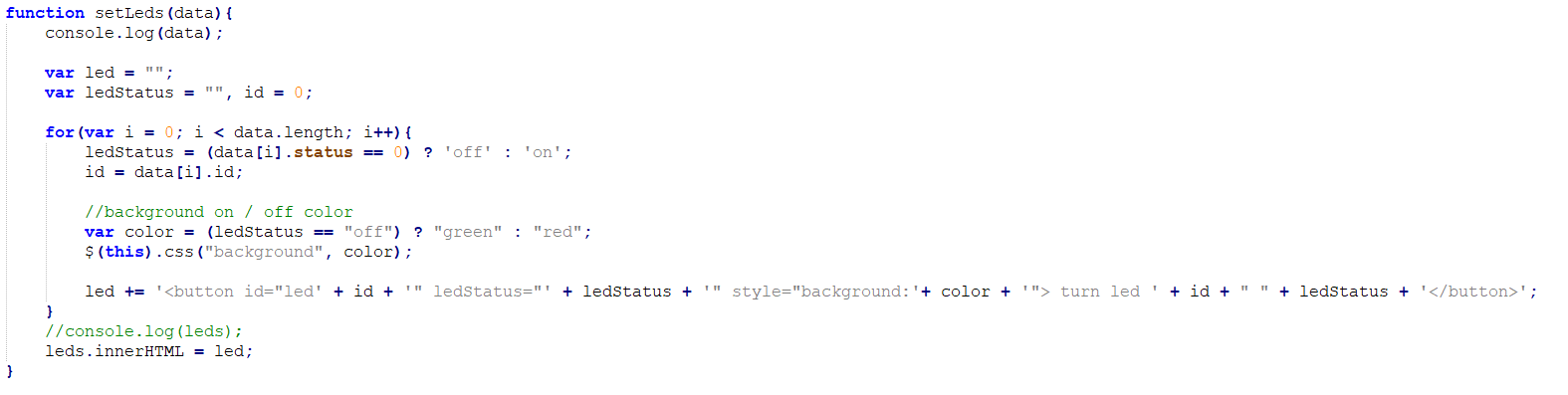
## Tempratuur geschiedenis lijst

Deze functie maakt een tabel aan op basis van de inkomende gegevens van de history van de cpu en weergeeft het op de scherm.



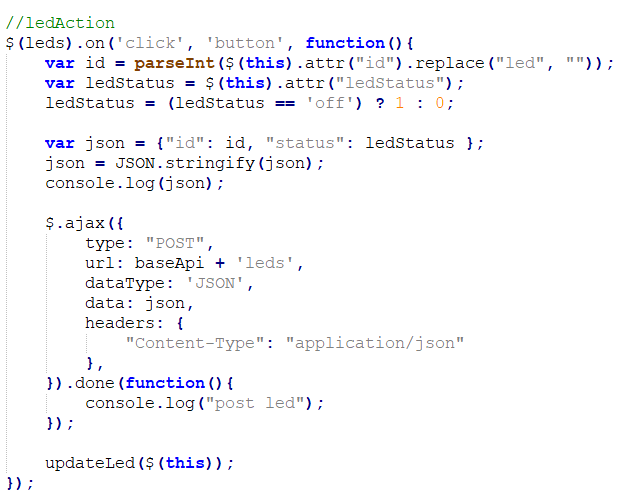
## Setleds functie

Deze functie maakt de knoppen aan voor de leds op basis van de leds de het binnen krijgt, het gebruikt de status om te bepalen of het een uit of aan knop is en voegt de id aan de knop toe zodat het weet over welke knop het ging die je later gebruik voor de post functie. De knop veranderd zelfs van kleur als die aan of uit staat.



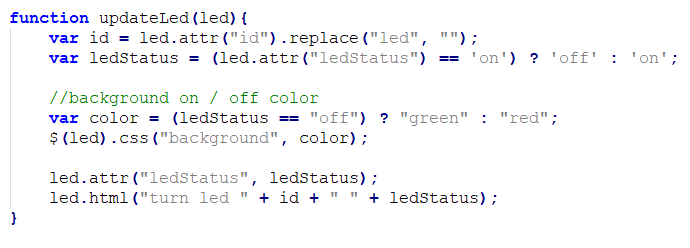
## Led action

In deze functie word de led aan en uit gezet door de status te versturen en de id.



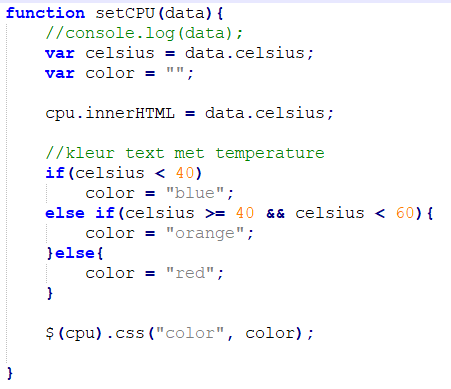
## Updaten van de led status kleur

Hier word de led in de pagina die je aan of uit gezet hebt up-ge-date met de kleur groen of rood



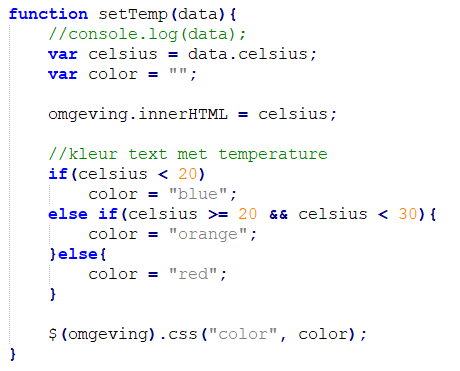
## Cpu tempratuur

Hier word de tempratuur op de pagina gezet en krijgt het een kleur op basis van de warmte blue is koud, orange is normaal en red is heet.



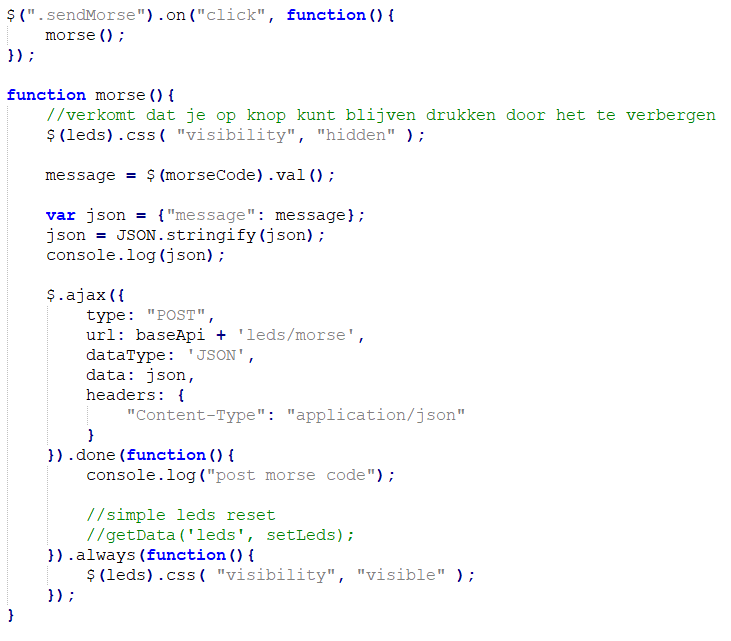
## Omgeving tempratuur

Zelfde als de cpu tempratuur



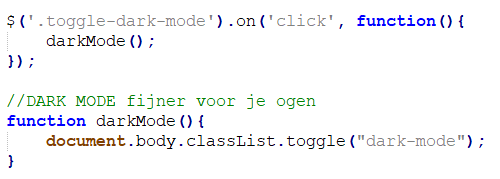
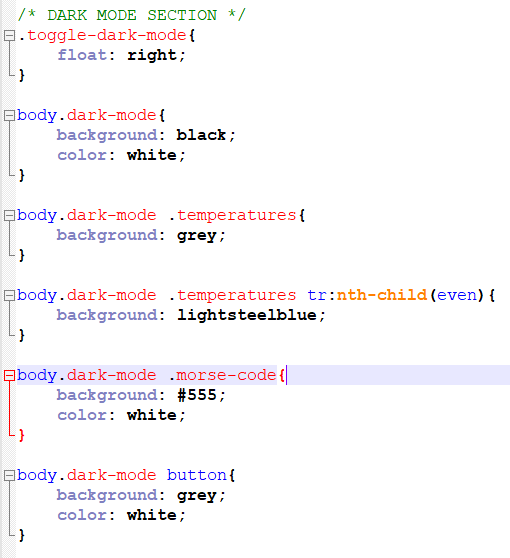
## Morse code

In deze functie word de bericht die je in de textarea voor de morse code getypt hebt gebruikt en verstuurt naar de leds. De leds worden ook verborgen om aan te geven dat ze bezig zijn.



## Dark mode

Dit heb ik voor mezelf ingevoerd omdat ik dark mode fijner vond. Ook kwam ik achter dat je niet elke class hoeft te veranderen door simple ‘body.dark-mode’ en dan class of id kun je alles veranderen zonder extra code in de javascript te schrijven.

# Interface

Dit is de interface die ik gemaakt had om de leds en tempratuur te beheren.

# 

# Reflectie

Ik had best veel problemen met het maken van deze project.

in de begin wal ik het in java met maven doen maar dat niet wal werken dus ging ik over na Laravel wat makkelijker ging om voor mij een api te maken.

het lezen van de temperaturen was best te doen zodra ik gevonden had hoe het moest ik liep wel tegen dat ik het eest moest casten naar float.

de leds daar ging het meeste tijd aan uit waardoor ik niet wist of ik het wel af zou. ik liep ook tegen de database waar ik niet mee kon verbinden voor deze dingen zou ik eerst hulp krijgen maar kwam er later toch uit.

voor de leds voerde ik steeds de opstart commando uit zonder sudo waardoor het geen permissies had om de gpio pins aan te passen, daarna liep ik tegen aan dat ik naar de verkeerde nummers keek ik keek namelijk naar de WiringPi prototype board dus 0, 1, 2 in plaats van de nummer waarmee die verbonden waren op de respberry pi zelf. ik liep ook nog eens tegen ajax post api maar dankzij Postman -> code kon ik zien wat ik nodig had.

er waren veel dingen waar ik tegen aan liep niet alleen in dit project maar ook daar buiten, maar toen het werkte was ik best opgelucht, ik vond het project in de begin niet echt leuk maar dat kwam door de dingen waar ik tegen aan liep toen het beter ging begon ik het ook leuker te vinden.

# Links en commands die me hielpen tijdens dit project.

<https://tecadmin.net/install-laravel-on-debian-9-stretch/>

sudo apt install mysql-server php-mysql

mariadb-server doen want mysql-server kan de pi niet vinden.

Voor de leds gebruikte ik php-gpio

https://github.com/ronanguilloux/php-gpio

toen ik route 404 error terug gaf gebruikte ik "php artisan route:clear"

voor de database problemen Laravel SQLSTATE[HY000] [1698] Access denied for user 'root'@'localhost' (1698)

"php artisan config:cache"

"php artisan config:clear"

"php artisan cache:clear"