

# Step-By-Step Guide Weerstation

MAAK JE EIGEN WEERSTATION VANAF 0  
KYANO DE MAERTELAERE

# INHOUD

Stap 1 – benodigheden .....	2
Stap 2 – fysiek weerstation .....	3
Stap 2.1 – breadboard.....	3
Stap 2.2 – BME280 sensor .....	3
Stap 2.3 – Rain gauge .....	4
Stap 2.4 – anemometer.....	4
stap 2.5 – Wind vane.....	5
Stap 3 – raspberry pi setup .....	6
Stap 4 – code .....	6
stap 4.1 – Librarys .....	6
stap 4.2 – setup & variabelen.....	7
stap 4.3 – rainsensor .....	8
stap 4.4 – wind speed .....	8
stap 4.5 – firebase setup .....	9
stap 4.6 – data ontvangen van database .....	9
stap 4.7 - main program .....	10
stap 4.8 – Shifting variables.....	11
stap 4.9 – sending data to database .....	12
Stap 5 – firebase setup.....	13
stap 5.1 – aanmaken database .....	13
stap 5.2 – Real time database setup .....	15
stap 5.3 – config toevoegen aan code .....	16
stap 6 – website.....	17
stap 6.1 – html pagina.....	17
Stap 6.2 – javascript .....	18
Stap 7 – fysiek weerstation .....	19
stap 7.1 – Electrical junction box .....	19
stap 7.2 – kleine electrical junction box.....	19
stap 7.3 – alles bevestigen aan elkaar.....	19

# STAP 1 – BENODIGDHEDEN

De eerste en makkelijkste stap is natuurlijk om alle onderdelen te bestellen. De lijst kan je vinden op mijn website:

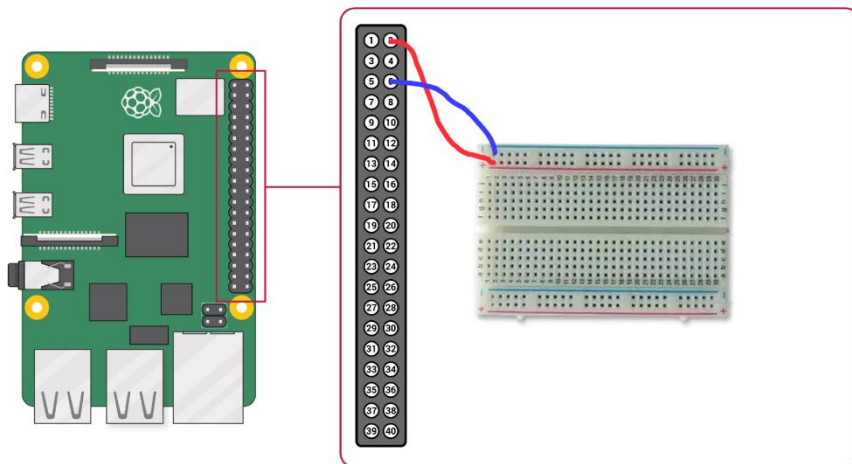
- <https://kyanodm.be/benodigdheden.html>

## STAP 2 – FYSIEK WEERSTATION

Nadat je alle onderdelen ontvangen hebt is de tweede stap om het fysiek weerstation in elkaar te zetten, dit houdt in om alle sensors te verbinden met de Raspberry pi.

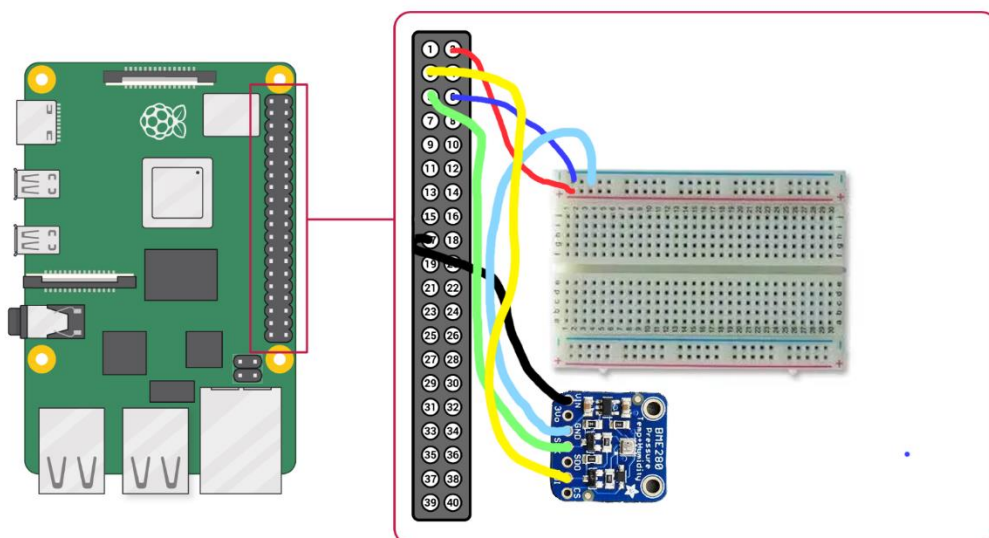
### STAP 2.1 – BREADBOARD

De eerste stap is om een breadboard te verbinden met je Raspberry pi, dit is zeer simpel zoals je kan zien.



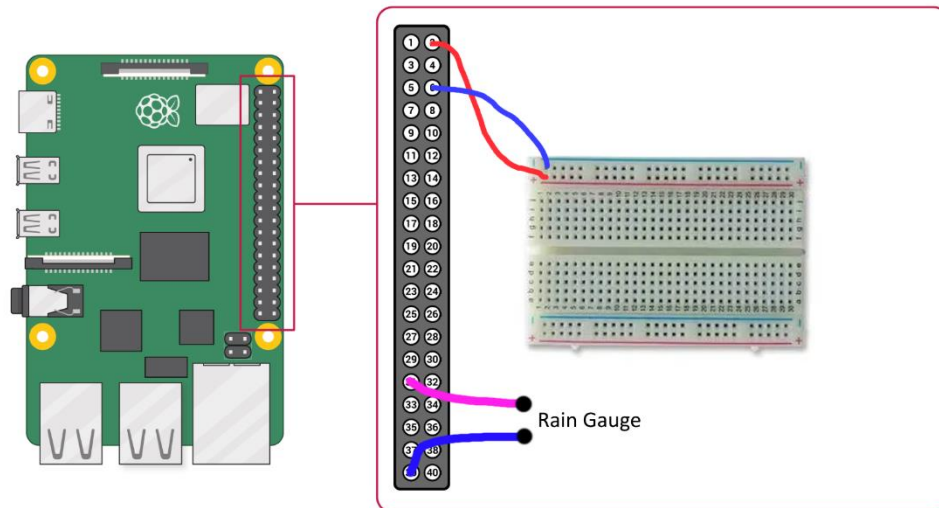
### STAP 2.2 – BME280 SENSOR

Daarna voeg je de BME280 sensor toe, het is belangrijk dat je de juiste pin van de BME sensor met de juiste pin op de Raspberry pi verbind, Als je dezelfde bme280 sensor als mij hebt gekocht heb je ook een rij van pinnetjes gekregen, deze moet je hiervoor nog solderen om de bme280 sensor, zodat je de kabels makkelijk kan verbinden



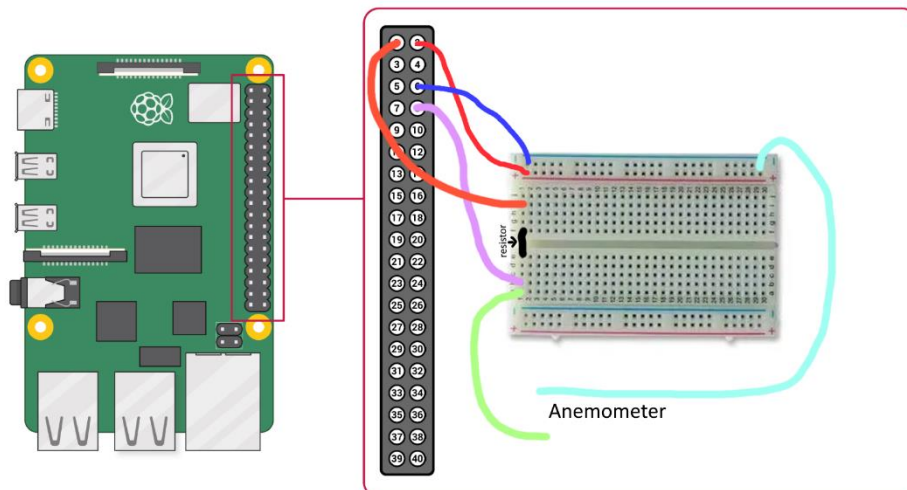
## STAP 2.3 – RAIN GAUGE

De volgende stap is om de regen sensor toe te voegen, dit is zeer makkelijk, deze moet je gewoon aan de Raspberry pi verbinden en dit zal werken.



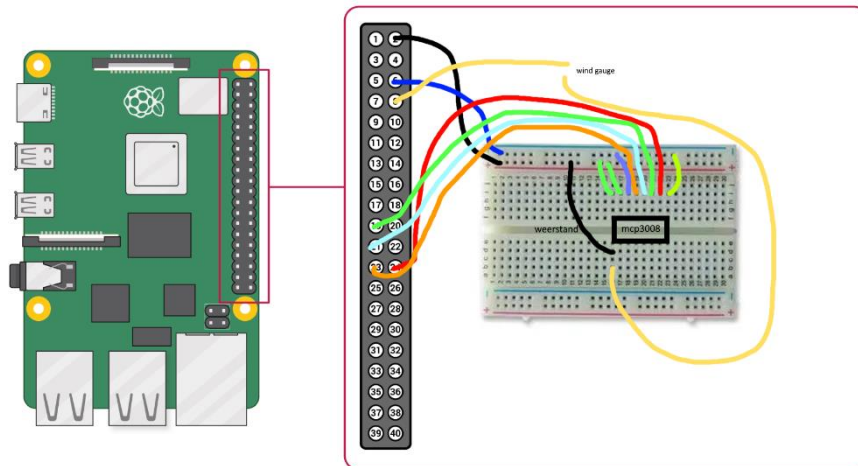
## STAP 2.4 – ANEMOMETER

Vervolgens voegen we de anemometer toe, deze is al iets ingewikkelder, maar zolang je alles juist verbind zal alles werken



## STAP 2.5 – WIND VANE

Voor de windrichting zal je ook de MCP3008 nodig hebben, dit zorgt ervoor dat een analoge input omgezet word naar een digitale input.



## STAP 3 – RASPBERRY PI SETUP

Stap 3 is om je Raspberry pi klaar te maken voor gebruik, als eerste heb je een OS nodig op je Raspberry pi. Ik heb Raspberry Pi OS gebruikt. Ik heb hier ook een guide voor gemaakt:

- <https://kyanodm.be/Os.html>

Als dit is gelukt zal je een paar librarys moeten download, je kan gewoon de volgende commando's in je command prompt uitvoeren:

- git clone <https://github.com/RaspberryPiFoundation/weather-station>
- sudo pip3 install RPi.bme280

Omdat je de eerste command hebt uitgevoerd is er een weather-station mapje aangemaakt met de benodigde librarys, in dat mapje maak een file aan genaamd **weatherstation.py**.

## STAP 4 – CODE

De volgende stap is om code te schrijven voor je weerstation:

### STAP 4.1 – LIBRARYS

Als eerste importen we alle nodige librarys, librarys zijn voorgeprogrammeerde functies en procedures

```
import RPi.GPIO as GPIO
import time
import datetime
import math
from gpiozero import Button
import statistics
import smbus2
import csv
import math
import pyrebase
from time import sleep
import board
from adafruit_bme280 import basic as adafruit_bme280
from gpiozero import MCP3008
```

## STAP 4.2 – SETUP & VARIABLEN

Bij deze stap voegen we een setup toe voor onze anemometer, en voegen we ook alle variabelen toe voor ons programma. (Tip: klik ok het tekst vak en doe ctrl + a, om alles te selecteren, je kan niet alles zien omdat de code langer is dan het tekst vak.)

```
GPIO.setmode(GPIO.BCM)
GPIO.setup(14, GPIO.IN)
pin = 14

store_speeds = []
radius_cm = 4.5
wind_interval = 5
wind_count = 0
CM_IN_A_KM = 100000.0
SECS_IN_AN_HOUR = 3600.0
ADJUSTMENT = 1.18
max_wind_speed = 0
counter = 0

val0 = 0
val1 = 0
val2 = 0
val3 = 0
val4 = 0
val5 = 0
val6 = 0
val7 = 0
val8 = 0
val9 = 0
val10 = 0
val11 = 0
val12 = 0
val13 = 0
val14 = 0
val15 = 0
val16 = 0
val17 = 0
```



## STAP 4.3 – RAINSENSOR

Vervolgens voegen we de code toe voor onze regensensor, deze code is heel makkelijk aangezien deze sensor werkt met een knop.

```
def bucket_tipped():
    global count
    global rainfall_rounded
    count += 1
    rain_fall = count * BUCKET_SIZE
    rainfall_rounded = round(rain_fall, 2)
    print(rainfall_rounded)

def reset_rain_fall():
    global count
    count = 0

def my_callback(channel):
    global wind_count
    wind_count = wind_count + 1
```

## STAP 4.4 – WIND SPEED

Hierna voegen we de code toe van onze wind snelheid sensor, deze code is een grote formule omdat het bereken van de windsnelheid door vele factoren beïnvloed word.

```
def my_callback(channel):
    global wind_count
    wind_count = wind_count + 1

GPIO.add_event_detect(pin, GPIO.RISING, callback=my_callback)

wind_speed_sensor = Button(5)
wind_speed_sensor.when_activated = my_callback

def calculate_speed(time_sec):
    global wind_count
    circumference_cm = (2 * math.pi) * radius_cm
    rotations = wind_count / 20
```

## STAP 4.5 – FIREBASE SETUP

Vervolgens gaan we de firebase config toevoegen, dit zorgt ervoor dat de data naar een database word verstuurd. YOUR\_KEY, moet je veranderen door de gegevens van jou database, in het volgende hoofdstuk zal ik je tonen hoe je dit moet doen.

```
config = {
    "apiKey": " YOUR_KEY ",
    "authDomain": " YOUR_KEY ",
    "databaseURL": " YOUR_KEY ",
    "storageBucket": " YOUR_KEY ",
}

firebase = pyrebase.initialize_app(config)
db = firebase.database()
```

## STAP 4.6 – DATA ONTVANGEN VAN DATABASE

De volgende code zal de data requesten van de database, zodat het programma altijd kan verder doen waar hij gebleven was, als we dit niet zouden hebben zou het programma elke keer starten met 0, nu kan het blijven verder doen. (Tip: klik ok het tekst vak en doe ctrl + a, om alles te selecteren, je kan niet alles zien omdat de code langer is dan het tekst vak.)

```
tim0 = firebase.database().child("graph/time/0").get().val()
tim1 = firebase.database().child("graph/time/1").get().val()
tim2 = firebase.database().child("graph/time/2").get().val()
tim3 = firebase.database().child("graph/time/3").get().val()
tim4 = firebase.database().child("graph/time/4").get().val()
tim5 = firebase.database().child("graph/time/5").get().val()
tim6 = firebase.database().child("graph/time/6").get().val()
tim7 = firebase.database().child("graph/time/7").get().val()
tim8 = firebase.database().child("graph/time/8").get().val()
tim9 = firebase.database().child("graph/time/9").get().val()

val0 = firebase.database().child("graph/temp/0").get().val()
val1 = firebase.database().child("graph/temp/1").get().val()
val2 = firebase.database().child("graph/temp/2").get().val()
val3 = firebase.database().child("graph/temp/3").get().val()
val4 = firebase.database().child("graph/temp/4").get().val()
val5 = firebase.database().child("graph/temp/5").get().val()
```

## STAP 4.7 - MAIN PROGRAM

Het volgende deel code gaat de data van de bme280 sensor halen, printen en in een variabel steken. Het gaat ook de windsnelheid berekenen en printen in de console. Het zal ook de windrichting bepalen van de windvane. Als laatste zal het ook de regenval bepalen. (Tip: klik ok het tekst vak en doe ctrl + a, om alles te selecteren, je kan niet alles zien omdat de code langer is dan het tekst vak.)

```
while True:
    start_time = time.time()

    i2c = board.I2C()
    bme280 = adafruit_bme280.Adafruit_BME280_I2C(i2c)
    humidity = round(bme280.humidity, 2)
    pressure = round(bme280.pressure, 2)
    ambient_temperature = round(bme280.temperature, 2)
    print(humidity, pressure, ambient_temperature)
    time.sleep(5)

    final_speed = calculate_speed(wind_interval)
    store_speeds.append(final_speed)

    wind_speed = statistics.mean(store_speeds)
    global wind_rounded
    wind_rounded = round(wind_speed, 2)

    if wind_rounded > max_wind_speed:
        max_wind_speed = wind_rounded

    print("Current wind speed:", wind_rounded, "km/h")
    wind_count = 0
    store_speeds = []

    wind = round adc.value * 3.3, 1)
    values.append(wind)

    if wind == 1.5 or wind == 1.3 or wind == 0.8:
        direction = "North"
    elif wind == 2.6:
```

## STAP 4.8 – SHIFTING VARIABLES

In dit deel van de code worden om het uur de variables doorgeschoven, ik heb dit gedaan omdat ik op mijn GIP website grafieken heb met 10 variabelen waar je de afgelopen 10 uur op kan zien wat er gebeurt is. En dit systeem maakt dit zeer gemakkelijk. (Tip: klik ok het tekst vak en doe ctrl + a, om alles te selecteren, je kan niet alles zien omdat de code langer is dan het tekst vak.)

```
counter +=1

if counter == 636:
    counter = 0
    tim0 = tim1
    tim1 = tim2
    tim2 = tim3
    tim3 = tim4
    tim4 = tim5
    tim5 = tim6
    tim6 = tim7
    tim7 = tim8
    tim8 = tim9
    tim9 = datetime.datetime.now().strftime("%H:%M")

    val0 = val1
    val1 = val2
    val2 = val3
    val3 = val4
    val4 = val5
    val5 = val6
    val6 = val7
    val7 = val8
    val8 = val9
    val9 = ambient_temperature

    val10 = val11
    val11 = val12
    val12 = val13
    val13 = val14
    val14 = val15
    val15 = val16
    val16 = val17
```

## STAP 4.9 – SENDING DATA TO DATABASE

De laatste stap is om de data te sturen naar de database. Dit is ook steeds dezelfde code, maar gewoon de variabele en de locatie is anders. (Tip: klik ok het tekst vak en doe ctrl + a, om alles te selecteren, je kan niet alles zien omdat de code langer is dan het tekst vak.)

```
db.child('graph').update({'temp/0': val0})
db.child('graph').update({'temp/1': val1})
db.child('graph').update({'temp/2': val2})
db.child('graph').update({'temp/3': val3})
db.child('graph').update({'temp/4': val4})
db.child('graph').update({'temp/5': val5})
db.child('graph').update({'temp/6': val6})
db.child('graph').update({'temp/7': val7})
db.child('graph').update({'temp/8': val8})
db.child('graph').update({'temp/9': ambient_temperature})

db.child('graph').update({'hum/0': val10})
db.child('graph').update({'hum/1': val11})
db.child('graph').update({'hum/2': val12})
db.child('graph').update({'hum/3': val13})
db.child('graph').update({'hum/4': val14})
db.child('graph').update({'hum/5': val15})
db.child('graph').update({'hum/6': val16})
db.child('graph').update({'hum/7': val17})
db.child('graph').update({'hum/8': val18})
db.child('graph').update({'hum/9': humidity})

db.child('graph').update({'pres/0': val20})
db.child('graph').update({'pres/1': val21})
db.child('graph').update({'pres/2': val22})
```

Zorg ervoor dat de code hierboven nog in de main program staat, de volgende code moet er buiten:

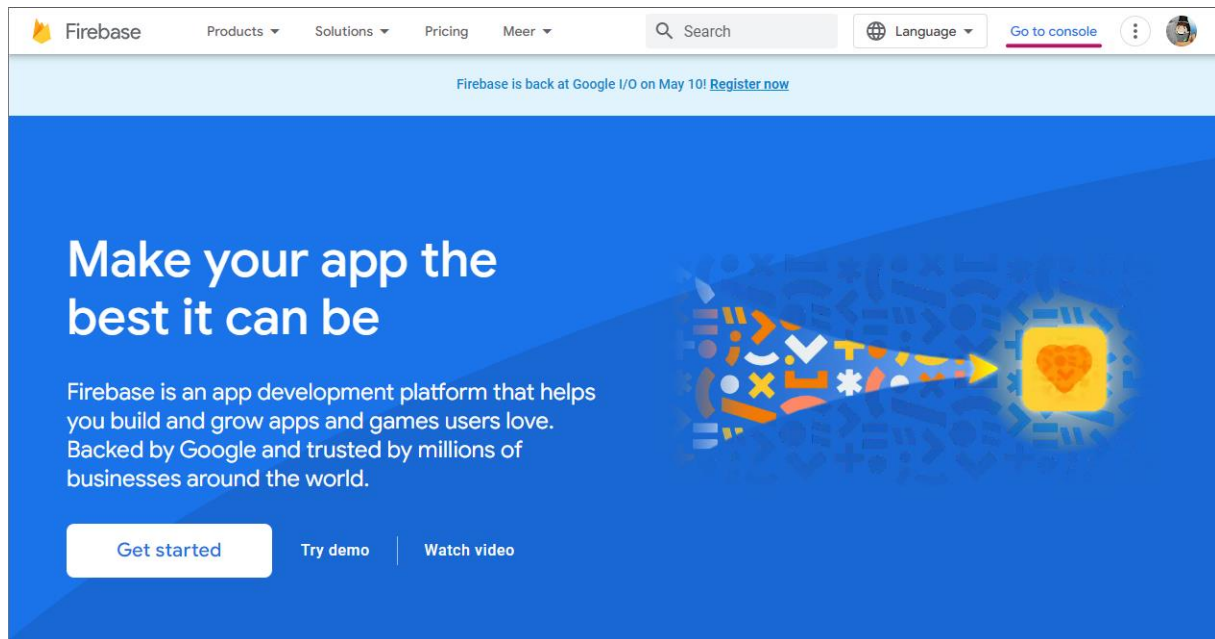
```
db.child('bme').update({'humidity': humidity})
db.child('bme').update({'temperature': ambient_temperature})
db.child('bme').update({'pressure': pressure})
db.child('bme').update({'max_wind_speed': wind_rounded})
db.child('bme').update({'wind_direction': direction})
db.child('bme').update({'water': rainfall_rounded})
```

# STAP 5 – FIREBASE SETUP

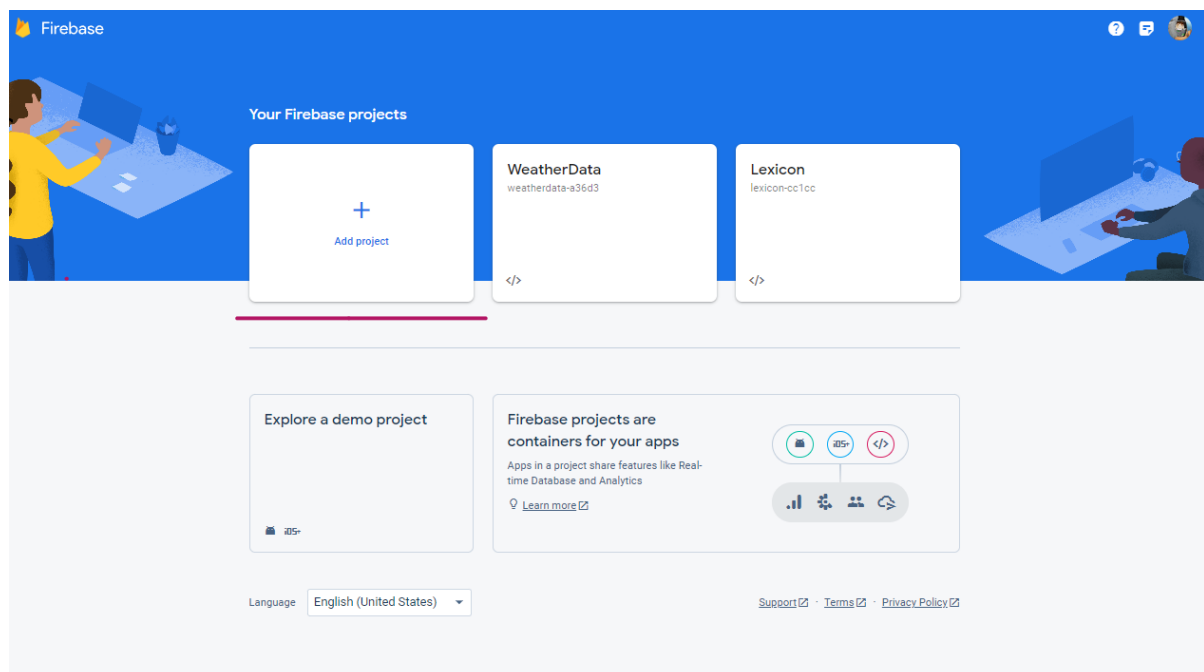
## STAP 5.1 – AANMAKEN DATABASE

De volgende stap is om een Firebase real time database aan te maken. Als eerste navigeer je naar de Firebase website: <https://firebase.google.com/>

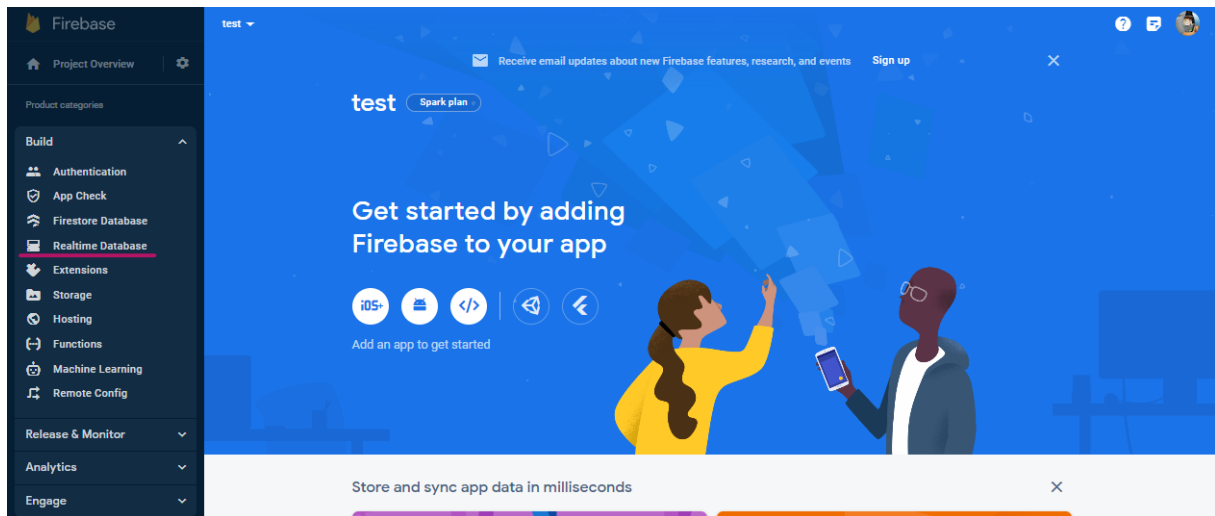
Vervolgens druk je rechtsboven op “Go to console”.



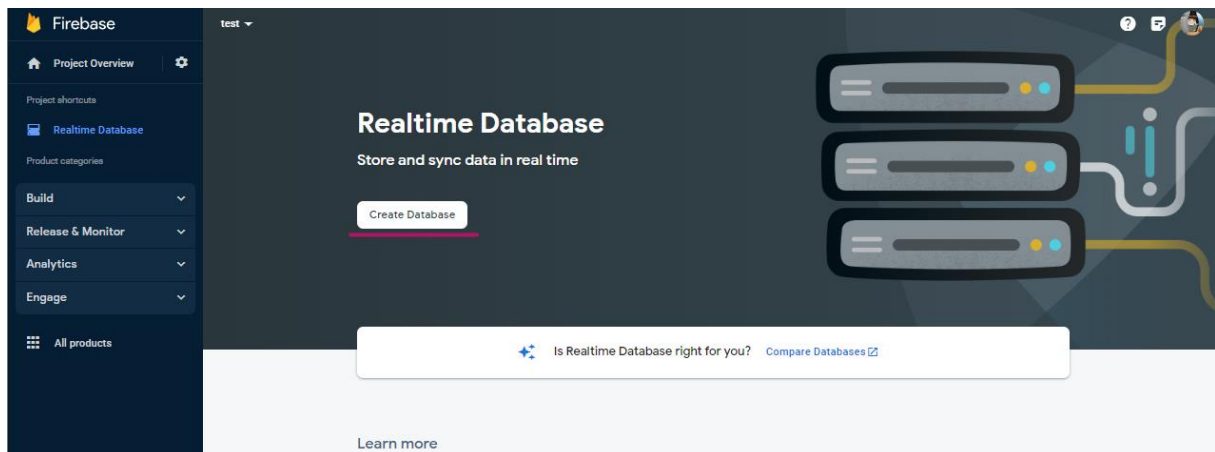
Daarna druk je op “Add project”, en volg je de stappen.



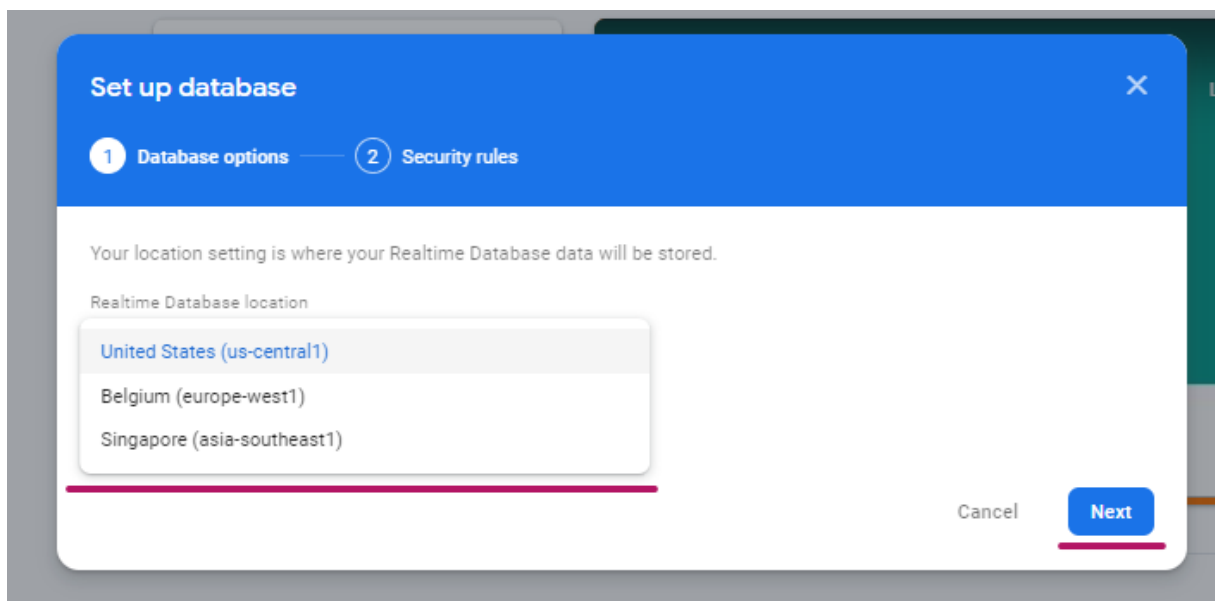
Als Firebase klaar is met het maken van je database druk je vervolgens recht bovenaan op “Realtime Database”.



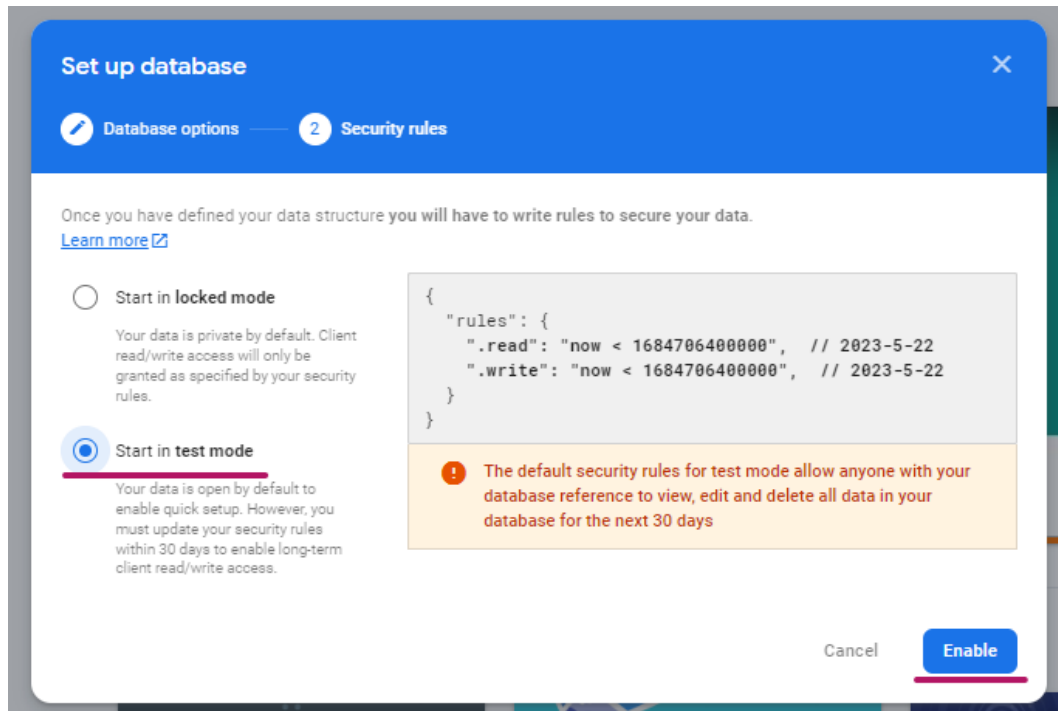
Daarna druk je op de knop “Create Database”.



Kies de dichtstbijzijnde locatie en druk “Next”.

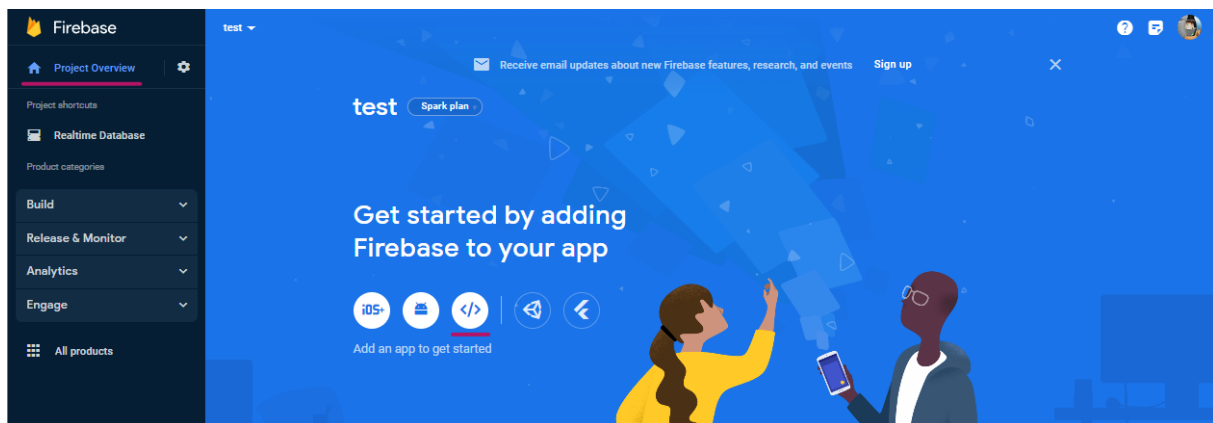


Druk op “Start in **test mode**” en druk “Enable”.



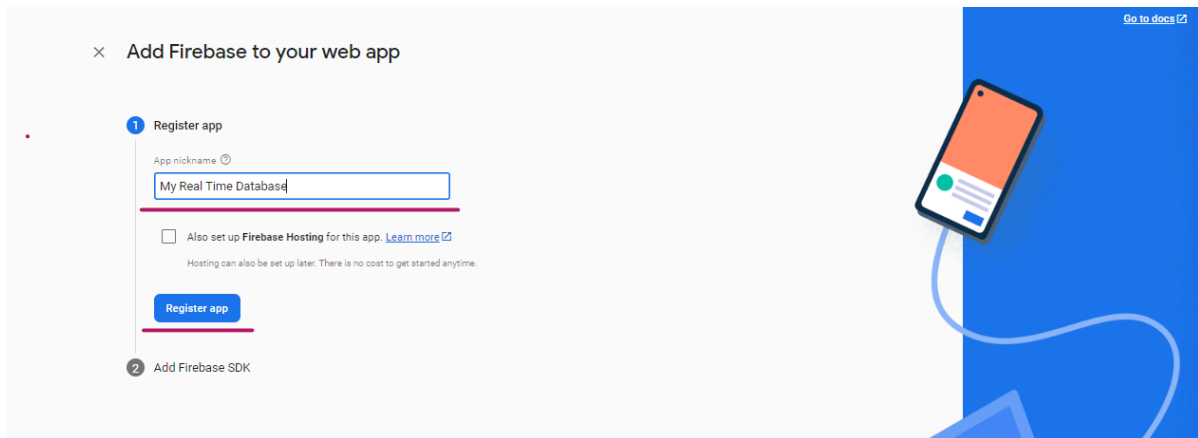
## STAP 5.2 – REAL TIME DATABASE SETUP

Daarna druk je op de knop: “`</>`” op de Project Overview pagina.





Vervolgens kies je een naam en druk je op “Register app”.



× Add Firebase to your web app

1 Register app

App nickname ⓘ

My Real Time Database

☐ Also set up Firebase Hosting for this app. [Learn more](#)

Hosting can also be set up later. There is no cost to get started anytime.

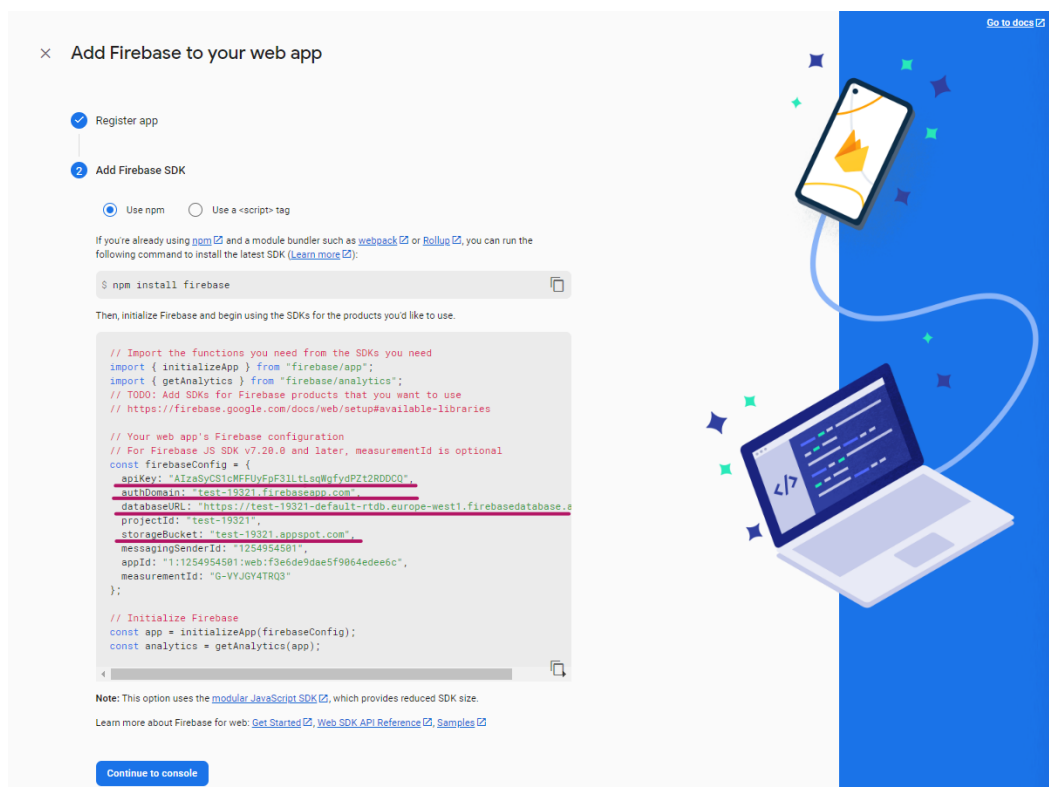
Register app

2 Add Firebase SDK

Go to docs

## STAP 5.3 – CONFIG TOEVOEGEN AAN CODE

Ten slotte bevind je jezelf op “Add Firebase to your app” pagina, hierop zie je de config die je nodig hebt. Je voegt deze gegevens toe aan je programma op je Raspberry pi. En daarna ben je klaar met je Raspberry pi, en je Firebase.



× Add Firebase to your web app

1 Register app

2 Add Firebase SDK

☒ Use npm ☐ Use a <script> tag

If you're already using [npm](#) and a module bundler such as [webpack](#) or [Rollup](#), you can run the following command to install the latest SDK ([Learn more](#)):

```
$ npm install firebase
```

Then, initialize Firebase and begin using the SDKs for the products you'd like to use.

```
// Import the functions you need from the SDKs you need
import { initializeApp } from "firebase/app";
import { getAnalytics } from "firebase/analytics";
// TODO: Add SDKs for Firebase products that you want to use
// https://firebase.google.com/docs/web/setup#available-libraries

// Your web app's Firebase configuration
// For Firebase JS SDK v7.20.0 and later, measurementId is optional
const firebaseConfig = {
  apiKey: "AIzaSyCSicMFUyFpF31ltlsgWafydPzt2RD0CQ",
  authDomain: "test-19321.firebaseio.com",
  databaseURL: "https://test-19321-default-rtdb.europe-west1.firebaseio.com",
  projectId: "test-19321",
  storageBucket: "test-19321.appspot.com",
  messagingSenderId: "1254954581",
  appId: "1:1254954581:web:f3e6de9dae5f9864edee6c",
  measurementId: "G-VYJGY4TRQ3"
};

// Initialize Firebase
const app = initializeApp(firebaseConfig);
const analytics = getAnalytics(app);
```

Note: This option uses the [modular JavaScript SDK](#), which provides reduced SDK size.

Learn more about Firebase for web: [Get Started](#), [Web SDK API Reference](#), [Samples](#)

Continue to console

Go to docs

# STAP 6 – WEBSITE

De laatste stap is om de data uit de Firebase te fetchen en te displayen op je website. Dus nu gaan we een simpele website maken waar de data op verschijnt.

## STAP 6.1 – HTML PAGINA

Als eerste maken we een simpele html pagina, ik gebruik Bootstrap om een simpele div aan te maken die centered staat op de pagina, in de <head> kan je zien da tik een link heb naar bootstrap, maar ook 2 links naar een library van Firebase.

```
<html lang="en">

<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Example Page</title>
  <!-- Firebase -->
  <script src="https://www.gstatic.com/firebasejs/8.10.0/firebase-
app.js"></script>
  <script src="https://www.gstatic.com/firebasejs/8.10.0/firebase-
database.js"></script>
  <!-- Bootstrap -->
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha3/dist/css/bootstrap.min.css" rel="stylesheet"
      integrity="sha384-
KK94CHFLLe+nY2dmCWGMq91rCGa5gtU4mk92HdvYe+M/SXH301p5ILy+dN9+nJOZ"
      crossorigin="anonymous">
  <!-- Eigen script -->
  <script src="js/MyScript"></script>
</head>

<body>
  <div class="row text-center justify-content-center mt-5 ">
    <div class="col-8 bg-primary" id="Temperature"></div>
  </div>
</body>
```

## STAP 6.2 – JAVASCRIPT

De laatste stap is op een javascript file aan te maken met de naam: "MyScript.js". Deze file plaats je ook in een mapje genaamd "js", op deze manier is alles gesorteerd per mapje. In dit script maken we de link naar de Firebase, en plaatsen we de data van de Firebase in de div met het id Temperature. Er zijn bepaalde dingen die je zelf zal moeten aanpassen in dit script, een van de dingen die je zeker moet veranderen is de firebaseConfig, je vervangt YOUR\_KEY door je eigen value van je Firebase, dit zijn de zelfde values die je in je python file van je Raspberry pi hebt moeten invullen. Dan heb je de line:

```
"const fetchedData = database.ref('graph/temp');"

```

'graph/temp' zal je moeten aanpassen naar het pad waar de data is gesaved in jou database.

```
const firebaseConfig = {
  apiKey: " YOUR_KEY ",
  authDomain: " YOUR_KEY ",
  databaseURL: " YOUR_KEY ",
  projectId: " YOUR_KEY ",
  storageBucket: " YOUR_KEY ",
  messagingSenderId: " YOUR_KEY ",
  appId: " YOUR_KEY ",
  measurementId: " YOUR_KEY "
};

if (firebase.apps.length === 0) { // check if Firebase app has
already been initialized
  firebase.initializeApp(firebaseConfig);
}

const database = firebase.database();
const fetchedData = database.ref('graph/temp');

fetchedData.on('value', (snapshot) => {
  const data = snapshot.val();
  const dates = snapshot.val();
  const dateValues = Object.values(dates);

  const temperatureDiv = document.getElementById("Temperature");
  temperatureDiv.innerHTML = dateValues.join(", ");
});

```

## STAP 7 – FYSIEK WEERSTATION

Nu je weerstation volledig werkt, kan je werken om het er fysiek beter uit te laten zien en echt usefull maakt.

### STAP 7.1 – ELECTRICAL JUNCTION BOX

De volgende stap is om de Raspberry pi en je breadboard in een waterdichte doos te steken, hier heb ik niet echt foto's van omdat het best straight forward is.

### STAP 7.2 – KLEINE ELECTRICAL JUNCTION BOX

Daarna kan je de BME280 sensor apart in de kleine box doen, de box moet ook niet volledig afgesloten worden, omdat de sensor anders geen verse lucht kan krijgen, en dit is noodzakelijk omdat hij anders de luchtvochtigheid niet accuraat kan meten, en het zelfde geldt voor temperatuur en luchtdruk.

Als deze 2 stappen klaar zijn kan je de dozen afsluiten zodat alles opgeborgen is.

### STAP 7.3 – ALLES BEVESTIGEN AAN ELKAAR

Nu dat de beide dozen klaar zijn, kan je alles aan mekaar vastmaken, in het pakket van de anemometer zaten 2 metalen staven, deze kan je op elkaar zetten, en daarop kan je een plastieken balk opzetten waarop je dan de anemometer en de wind vane kan op bevestigen. Vervolgens heb ik de dozen met zip ties vastgemaakt op de paal. En dan alles in de statief gezet. Vervolgens kan je de losse kabels nog een beetje vast maken met zip ties en dan ben je klaar met je weersation