

Rrjetet e Sensorëve Wireless

PhD. C. Lamir Shkurti

Shembuj të përdorimit të MKR1000

Përmbajtja

Çka janë dhe pse përdoren mikrokontrollerët	3
MKR1000.....	3
Çka është MKR1000	3
Specifikimet teknike.....	4
Përdorimi i MKR1000 në Arduino Software (IDE).....	7
Shembulli i parë me MKR1000.....	8
Shembuj me MKR1000 dhe SensorKit	10
Ndërprerësi me buton	10
LED-i RGB.....	13
Sensori i dritës.....	15
Sensori i flakës	16
Sensori i lagështisë dhe temperaturës.....	18
Detyrë	20
Përdorimi i Blynk.....	21
Monitorimi i temperaturës dhe lagështisë së ajrit nëpërmjet telefonit.....	21
Dërgimi i të dhënave në bazën e të dhënave nëpërmjet një Ueb API nga pajisja MKR1000	31
Krijimi i bazës së të dhënave në MySQL	31
Marrja e të dhënave nga baza e të dhënave nëpërmjet një Ueb API dhe komandimi i pajisjes MKR1000 nga të dhënat e marura	37
Përdorimi i MKR 1000 si një Ueb Server	44
Përdorimi i JWT(JSON Web Token) për të siguruar shkëmbimin e të dhënave në mënyrë të sigurt	49

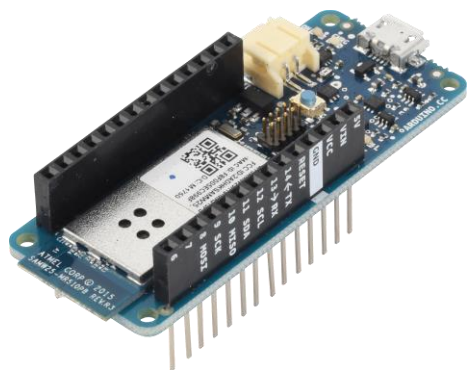
Çka janë dhe pse përdoren mikrokontrollerët

Mikrokontrolleri është një kompjuter i vogël në një qark të integruar. Mikrokontrollërat janë pajisje elektronike të përdorura për të kryer një sërë detyrash automatizuese dhe kontrolluese në pajisje dhe sisteme të ndryshme. Një mikrokontroller përmban një ose më shumë CPU (Processor cores) së bashku me memorien e programueshme dhe pjesët periferike hyrëse dhe dalëse.

Mikrokontrollerët kryesisht përdoren në kontrollimin e automatizimit të produkteve të ndryshme, sikurse sistemet e kontrollit të motorëve automobilistikë, pajisjet mjekësore, telekomandat, pajisjet, mjetet e rrymës, lodra e të tjera.

IoT (Interneti i Gjërave): Në kontekstin e IoT, mikrokontrollërat janë thelbësore. Ata ndihmojnë pajisjet të komunikojnë me njëra-tjetrën dhe të mbledhin dhe dërgojnë të dhëna nëpërmjet internetit. Kjo është e rëndësishme për zhvillimin e smart cities, smart homes dhe aplikacioneve të tjera të ndërlidhura.

MKR1000



Çka është MKR1000

MKR1000 është një pllakë zhvillimore e cila në vete përmban një mikrokontroller e dizajnuar për të ofruar një zgjidhje të lehtë dhe pak të kushtueshme për të gjithë ata që dëshirojnë të merren me sensor, aktuatorë dhe lidhje të lehtë me Wi-Fi pavarësisht eksperiencës së tyre në programimin e mikrokontrollerëve.

Përdorimi i WiFi: MKR1000 është një mikrokontroller i pajisur me një modul WiFi të integruar, që e bën të mundur lidhjen me rrjetet pa tel. Kjo e bën të përshtatshëm për projekte IoT (Interneti i Gjërave), ku lidhja në internet është e rëndësishme.

MKR1000 programohet duke përdorur Arduino Software (IDE – *integrated development environment*). Gjithashtu për ta programuar MKR1000 përdorim gjuhën programuese C dhe C++, sintaksën dhe mënyrën e shkrimit të kësaj gjuhe mund ta gjeni në internet por mënyra më e mirë për ta mësuar është përmes shembujve dhe ushtrimeve të ndryshme.

MKR1000 është bazuar në procesorin ARM Cortex-M0, i cili ofron një performancë të mirë dhe shpejtësi të lartë të procesimit.

MKR1000 është dizajnuar për të punuar një bateri dhe ka efikasitet të lartë të energjisë, duke e bërë të përshtatshëm për aplikacionet në terren dhe në ambientet pa rrjet elektrik të qëndrueshëm.

MKR1000 është pajisur me një portë USB për programim duke e bërë të lehtë për të komunikuar me kompjuterin dhe për të ngarkuar kodin.

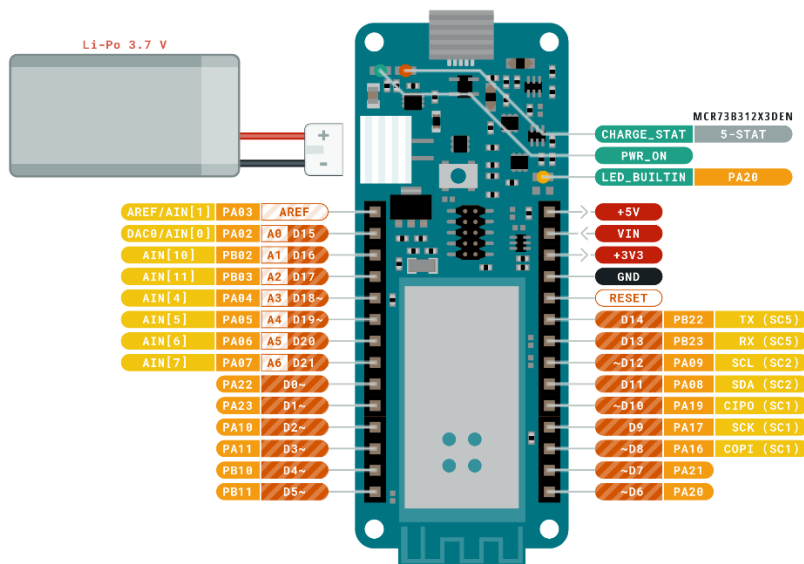
Specifikimet teknike

Microcontroller	SAMD21 Cortex-M0+ 32bit low power ARM MCU
Board Power Supply (USB/VIN)	5V
Supported Battery(*)	Li-Po single cell, 3.7V, 700mAh minimum
Circuit Operating Voltage	3.3V
Digital I/O Pins	8
PWM Pins	12 (0, 1, 2, 3, 4, 5, 6, 7, 8, 10, A3 - or 18 -, A4 -or 19)
UART	1
SPI	1
I2C	1
Analog Input Pins	7 (ADC 8/10/12 bit)
Analog Output Pins	1 (DAC 10 bit)
External Interrupts	8 (0, 1, 4, 5, 6, 7, 8, A1 -or 16-, A2 - or 17)
DC Current per I/O Pin	7 mA
Flash Memory	256 KB
SRAM	32 KB
EEPROM	no

Clock Speed	32.768 kHz (RTC), 48 MHz
LED_BUILTIN	6
Full-Speed USB Device and embedded Host	
Length	61.5 mm
Width	25 mm
Weight	32 gr.



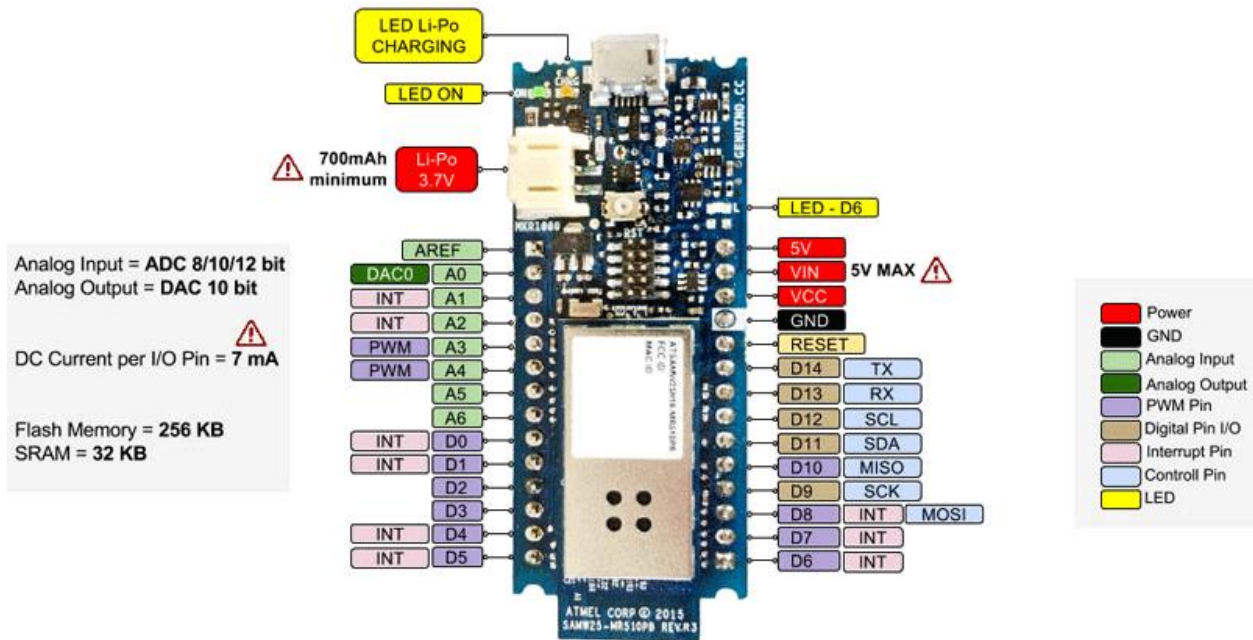
**ARDUINO
MKR WIFI 1000**



Ground	Internal Pin	Digital Pin	Microcontroller's Port
Power	SWD Pin	Analog Pin	
LED	Other Pin	Default	

ARDUINO.CC

This work is licensed under the Creative Commons Attribution-ShareAlike 4.0 International License. To view a copy of this license, visit <http://creativecommons.org/licenses/by-sa/4.0/> or send a letter to Creative Commons, PO Box 1868, Mountain View, CA 94041, USA.



MKR1000 ka një numër të caktuar portesh dhe pinash që mund të përdoren për lidhjen me komponentë të ndryshëm. Këtu janë disa nga pinat kryesore të MKR1000 dhe shpjegime të shkurtra për secilin prej tyre:

VIN: Pini i tensionit hyrës. Ky pini mund të përdoret për të furnizuar tensionin e nevojshëm për MKR1000 nga një burim tensioni të jashtëm.

5V: Pini i tensionit 5V. Ky pini furnizon tensionin 5V, mund të përdoret për të dhënë energji për pajisje të jashtme.

3.3V: Pini i tensionit 3.3V. Ky pini furnizon tensionin 3.3V për pajisjet që operojnë me këtë nivel tensioni.

GND: Pini i përbashkët për të lidhur pajisjet me tokën- pini negativ.

Digital I/O Pins: Ka disa porta I/O (Input/Output) digitale që mund të përdoren për të lexuar ose kontrolluar një nivel tensioni digital.

Analog Input Pins: Ka disa porta për leximin e vlerave analoge nga sensorë analogë.

PWM Pins: Ka porta me shtrirje modulacioni që mund të përdoren për të kontrolluar intensitetin e dritës, shpejtësinë e një motori.

Serial Communication Pins (RX, TX): Pini i pranimit (RX) dhe pini i dërgimit (TX) përdoren për komunikim serial me pajisje të tjera.

SPI, I2C, UART Pins: Ka porta dedikuar për protokollet e komunikimit të ndryshme si SPI, I2C, dhe UART, që mund të përdoren për lidhjen me pajisje të ndryshme.

Përdorimi i MKR1000 në Arduino Software (IDE)

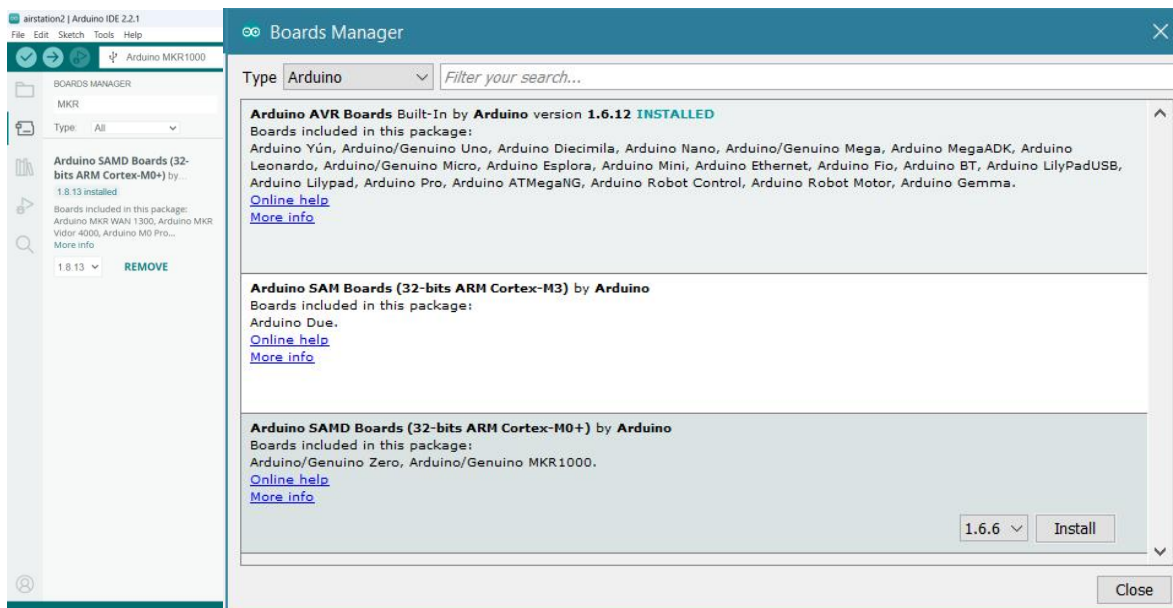
Së pari për ta përdorur MKR1000 në Arduino Software (IDE), duhet të shkarkoni aplikacionin e arduinos për programim (Arduino Software). Dhe për ta bërë këtë së pari shkoni në faqen arduino.cc në internet, shkoni të opsiioni Software dhe në nënmeny zgjidhni Downloads. Pastaj nëse jeni përdorues i sistemit operativ Windows zgjidhni opsiinin e parë në të majtë [Windows Installer, for windows XP and up](#), kjo do të ju drejtoj të një faqe tjetër ku ju duhet të zgjidhni opsiinin “*JUST DOWNLOAD*”, ose nëse ndokush nga ju është i interesuar që të kontribuoj në projektin e arduinos me mjete financiare mund ta zgjedhë opsiinin “*CONTRIBUTE & DOWNLOAD*” dhe të vazhdoj procedurat tjera.

Pasi ta keni shkarkuar file-in vetëm klikoni mbi të dhe Arduino IDE do të jetë e instaluar. Pas këtyre hapave, duhet të vazhdoni duke instaluar libraritë e kërkuara për MKR1000.

Nëse dëshirojmë që ta përdorim MKR1000 ne Arduino IDE së pari duhet të instalojmë Atmel SAMD Core në të. Kjo është një procedurë e thjeshtë :

>Tools menu -> Boards -> Boards Manager

Vetëm kërkoni për MKR1000 ose Atmel SAMD Core dhe rezultati do të jetë i njëjtë. Vetëm klikoni butonin instalo dhe gjithçka do të jetë në rregull.



Nëse keni probleme me instalim të drivera-ve, d.m.th. sistemi operativ i juaji nuk e njeh MKR1000 atëherë shkoni te faqja :

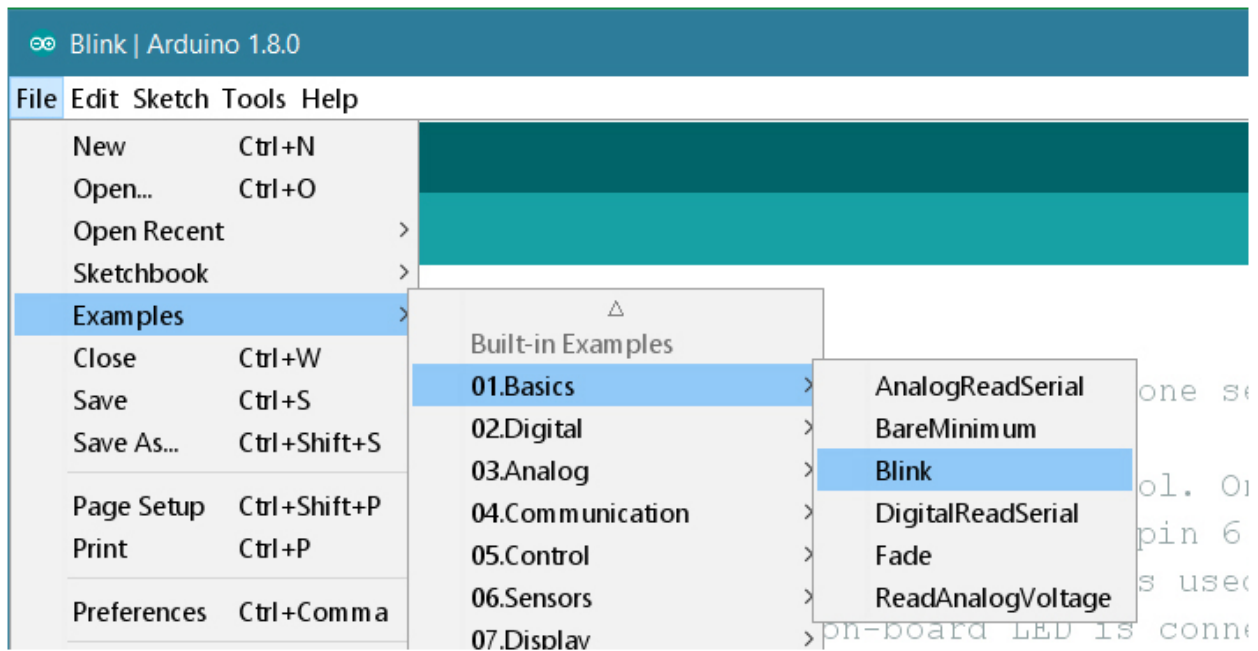
<https://www.arduino.cc/en/Guide/MKR1000>

te kapitulli për instalim të driver-ve i gjeni të gjitha instruksionet se si të procedoni.

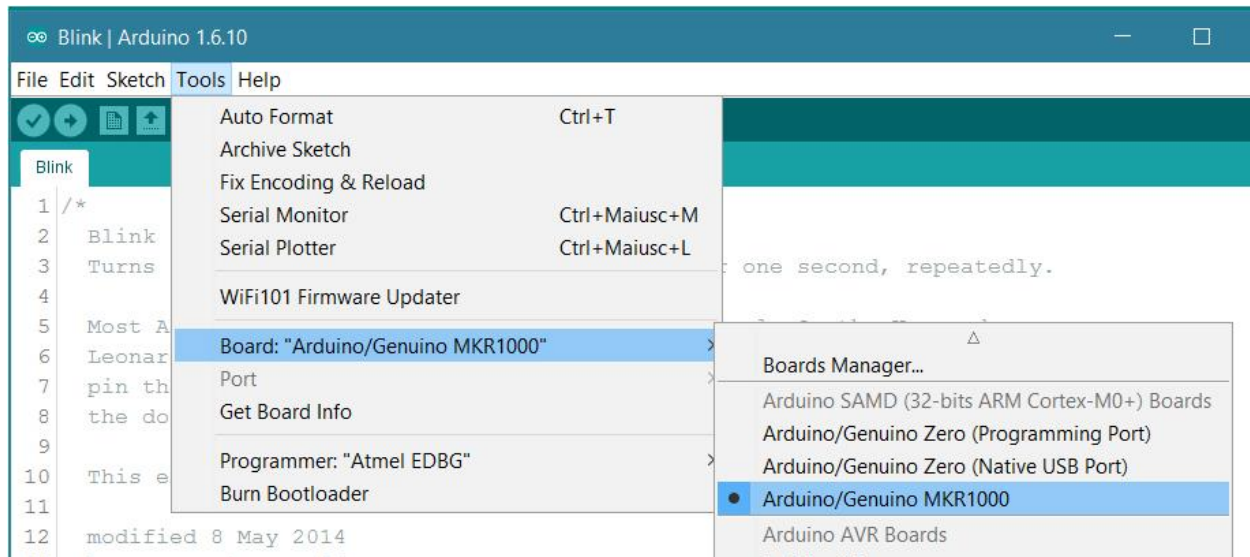
Shembulli i parë me MKR1000

Së pari do të përdorim një shembull të thjeshtë për blinkimin e një LED-i i cili mund të gjendet i gatshëm pasi ta hapni IDE për Arduino.

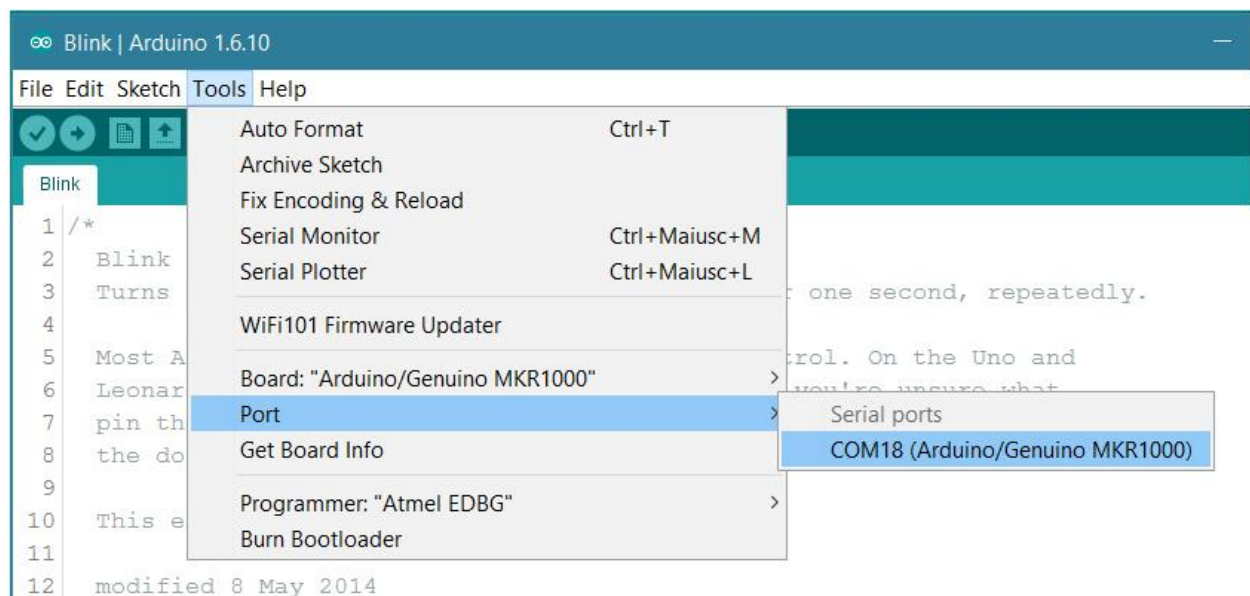
Shkoni në **File > Examples > 01.Basics > Blink**



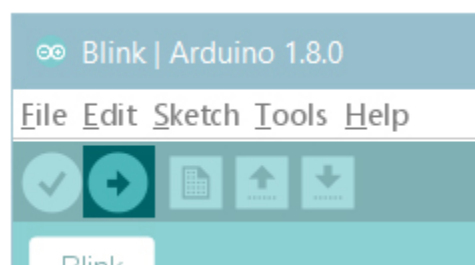
Pas këtij veprimi duhet të zgjedhim board-in me të cilin jemi duke punuar. Shkojmë te **Tools > Board** dhe zgjedhim MKR1000.



Pasi të zgjedhim board-in duhet të zgjedhim portin me të cilin do të komunikojmë përmes USB me MKR1000. Shkojmë te **Tools > Port** > dhe zgjedhim portin i cili përveç numrit të portit përmban edhe emrin e board-it ton MKR1000.



Tani çka na mbetet është që ta ngarkojmë kodin tonë në board duke klikuar butonin *Upload*.



Pasi të prekim këtë buton mund të shihni LED-at e TX dhe RX që blinkojnë dhe na tregojnë që board-i dhe kompjuteri janë duke komunikuar. Pasi ngarkimi i kodit të jetë përfunduar mesazhi *Done uploading* do të paraqitet në status bar.

Disa sekonda pasi ngarkimi i kodit të përfundoj ju mund të shihni duke blinkuar (ndalet dhe ndezët) LED-in i cili është në pllakë i shënjuar me shigjetë të kuqe në foton më poshtë. Nëse kjo ndodhë atëherë ju keni arritur me sukses të bëni projektin e parë me MKR1000.



Shembuj me MKR1000 dhe SensorKit

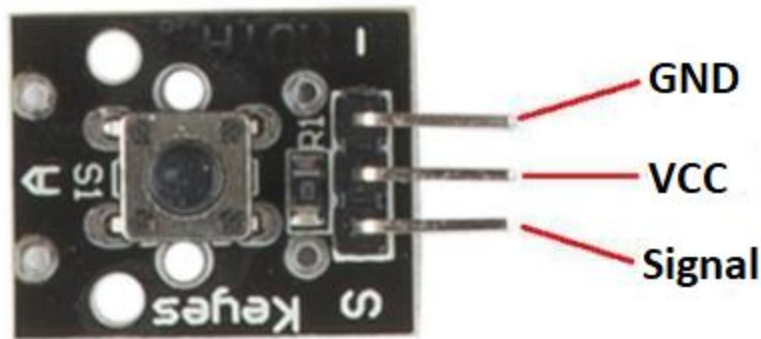
Duhet pasur kujdes me lidhjen e të gjitha moduleve në MKR1000 sepse përndryshe nga Arduino UNO **pinat që kontrollojmë në MKR1000 punojnë me tension 3.3V**. Prandaj duhet pasur kujdes që lidhjet të bëhen në baze të këshillave që janë dhënë më poshtë në mënyrë që mos të vije deri te dëmtimi i pllakës.

Ndërprerësi me buton

Moduli me buton përdoret për të dhënë një komandë MKR1000, përmes shtypjes së butonit. Duhet pasur kujdes që moduli i butonit të lidhet me MKR1000 sipas fotos së treguar më poshtë. Pini i sinjalit (Signal) duhet të lidhet në MKR1000 në pinin të cilin kemi vendosur ta kontrollojmë në kodin që kemi shkruar.

D.m.th. për ta shfrytëzuar kodin e mëposhtëm lidhim

- GND nga moduli në GND të MKR1000
- VCC nga moduli në VCC të MKR1000
- Signal nga moduli në pinin 3 të MKR1000



Në rastin tonë kemi përdorur pinin e tretë të MKR1000 por ju mund të eksperimentoni me secilin prej tyre, gjithashtu kemi përdorur edhe LED-in i cili është i vendosur në pllakën e MKR1000, sikur në shembullin e mëhershëm të MKR1000.

Një sqarim i thjeshtë i kodit rresht pas rreshti do të ishte kështu:

- `int led = LED_BUILTIN;` deklarimi i një variable në të cilën do ta ruajmë numrin e pinin me të cilin do ta kontrollojmë LED-in.
- `int buttonpin = 3;` deklarimi i një variable në të cilën do ta ruajmë numrin e pinin përmes të cilit do ta lexojmë gjendjen e butonit.
- `int val;` deklarimi i një variable ku do ta ruajmë gjendjen e butonit.
- `void setup()`
 - {
 - `ky funksion përdoret për inicializimin e komponentëve që dëshirojmë ti përdorim gjatë projektit. Ekzekutohet vetëm një herë pasi të fillon ekzekutimin e kodit mikrokontrolleri dhe pasi të përfundohet i gjithë kodi që ka brenda kalon në funksionin void loop();`
 - }
- `pinMode(variabila1,variabila2);` ky funksion përdoret për të inicializuar modën e punës së pinin që dëshirojmë ta përdorim, dëshirojmë ta përdorim pinin e caktuar të MKR1000 si hyrje apo si dalje. Po ashtu shohim se funksioni pranon dy parametra, parametri i parë `variabila1` është pini që dëshirojmë ta inicializojmë, dhe `variabila2` është modi në të cilin dëshirojmë ta inicializojmë INPUT apo OUTPUT.
- `void loop()`
 - {
 - `ky funksion është pjesa ku mikrokontrolleri ekzekuton kodin në mënyrë të përsëritshme. D.m.th. fillon nga rreshti i parë i funksionit shkon te secili rresht njëri pas tjetrit dhe pasi ta ketë ekzekutuar rreshtin fundit fillon prapë nga rreshti i parë.`
 - }
- `digitalRead(variabila1);` ky është një funksion të cilin e përdorim për ta lexuar gjendjen e pinin të caktuar. Parametri `variabila1` është numri i pinin të cilin dëshirojmë ta lexojmë. Ky funksion kthen si përgjigje 1 apo 0, TRUE ose FALSE varësisht nga gjendja e pinin, 3.3 Volt apo 0 Volt
- `if(kushti)`
 - {

nëse plotësohet kushti që kemi vendosur brenda kllapave të vogla atëherë do të ekzekutohet kodi që është brenda kllapave gjarpërore, përndryshe kodi vazhdon në else

```
}  
else  
{
```

Nëse nuk plotësohet kushti brenda if ekzekutimi i kodit vazhdon këtu.

```
}
```

- `digitalWrite(variabila1,variabila2)`; ky funksion përdoret për të caktuar gjendjen e pinit në MKR1000, HIGH – gjendja e pinit do të jetë 1 logjik, dhe LOW gjendja e pinit do të jetë 0 logjike.

ButtonSwitch_module

```
int led = LED_BUILTIN; //Define the LED pin  
int buttonpin = 3; //Define the push button pin  
int val; //Define a numeric variable  
  
void setup()  
{  
  pinMode(led,OUTPUT);  
  pinMode(buttonpin,INPUT);  
}  
  
void loop()  
{  
  val = digitalRead(buttonpin); // check the state of the button  
  if(val==HIGH) // if button is pressed, turn LED on  
  {  
    digitalWrite(led,HIGH);  
  }  
  else  
  {  
    digitalWrite(led,LOW);  
  }  
}
```

LED-i RGB

LED përdoret si shkurtes për Light Emitting Diode dhe RGB për Red, Green, Blue



RGB LED-i përdoret për të krijuar ngjyra të ndryshme përmes tri ngjyrave bazë, dhe kontrollohet përmes tre pinave të MKR1000. Për tu përshtatur me kodin më poshtë i cili ndez tetë ngjyrat e ndryshme që mund të krijohen duhet lidhur këta pina:

- GREEN nga moduli te pini 2 në MKR1000
- RED nga moduli te pini 1 në MKR1000
- BLUE nga moduli te pini 3 në MKR1000

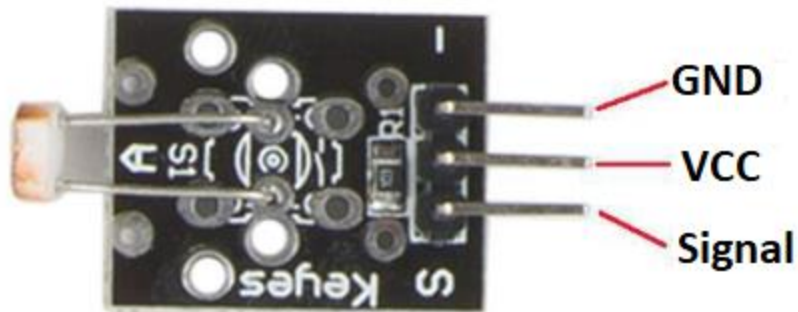
Dallimi mes dy moduleve më lartë është vetëm nga lloji i komponentës që është përdorur si RGB LED.

Gjithashtu për të bërë test shembullin mund ta shfrytëzoni kodin më poshtë, i cili çdo një sekondë shfaqë një nga tetë ngjyrat që mund të krijohen me RGB LED.

```
int red_light_pin= 1;
int green_light_pin = 2;
int blue_light_pin = 3;
void setup() {
  pinMode(red_light_pin, OUTPUT);
  pinMode(green_light_pin, OUTPUT);
  pinMode(blue_light_pin, OUTPUT);
}
```

```
void loop() {  
  RGB_color(255, 0, 0); // Red  
  delay(1000);  
  RGB_color(0, 255, 0); // Green  
  delay(1000);  
  RGB_color(0, 0, 255); // Blue  
  delay(1000);  
  RGB_color(255, 255, 125); // Raspberry  
  delay(1000);  
  RGB_color(0, 255, 255); // Cyan  
  delay(1000);  
  RGB_color(255, 0, 255); // Magenta  
  delay(1000);  
  RGB_color(255, 255, 0); // Yellow  
  delay(1000);  
  RGB_color(255, 255, 255); // White  
  delay(1000);  
}  
  
void RGB_color(int red_light_value, int green_light_value, int blue_light_value)  
{  
  analogWrite(red_light_pin, red_light_value);  
  analogWrite(green_light_pin, green_light_value);  
  analogWrite(blue_light_pin, blue_light_value);  
}
```

Sensori i dritës



Në këtë sensor tensioni analog në pinin SIGNAL ndryshon varësisht nga ndriçimi i jashtëm. Ne do ta matim këtë tension duke shfrytëzuar pinin **A0 të MKR1000** dhe do ta shfaqim në komunikimin serik i cili është i mundur në IDE e Arduinos.

Për ta hapur serial monitorin duhet të shkoni te **Tools > Serial Monitor**, dhe të rregulloni shpejtësinë e komunikimit nëse nuk është 9600 siç është specifikuar në kodë.

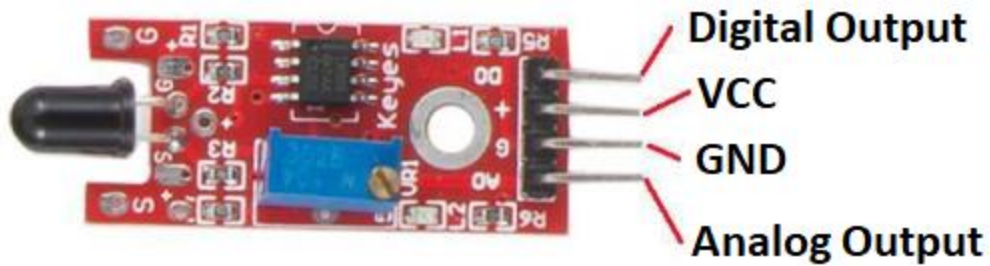
PhotoResistor

```
int sensorPin = A0; // select the analog input pin for the photoresistor

void setup() {
  Serial.begin(9600);
}

void loop() {
  Serial.println(analogRead(sensorPin));
  delay(200);
}
```

Sensori i flakës



Shembulli me sensor të flakës mund të bëhet duke shfrytëzuar kodin e mëposhtëm dhe duke lidhur pinat për VCC, GND në vendet përkatëse si dhe **Analog Output në A0 të MKR1000**, ndërsa Digital Output nuk kemi nevojë ta lidhim. Gjithashtu duhet të hapim edhe dritaren për komunikim serik sikur në shembullin e mëhershëm për të lexuar mesazhet e printuara me funksionet për printim.

```
// lowest and highest sensor readings:
```

```
const int sensorMin = 0; // sensor minimum
```

```
const int sensorMax = 1024; // sensor maximum
```

```
void setup() {
```

```
  // initialize serial communication @ 9600 baud:
```

```
  Serial.begin(9600);
```

```
}
```

```
void loop() {
```

```
  // read the sensor on analog A0:
```

```
  int sensorReading = analogRead(A0);
```

```
  // map the sensor range (four options):
```

```
  // ex: 'long int map(long int, long int, long int, long int, long int)'
```

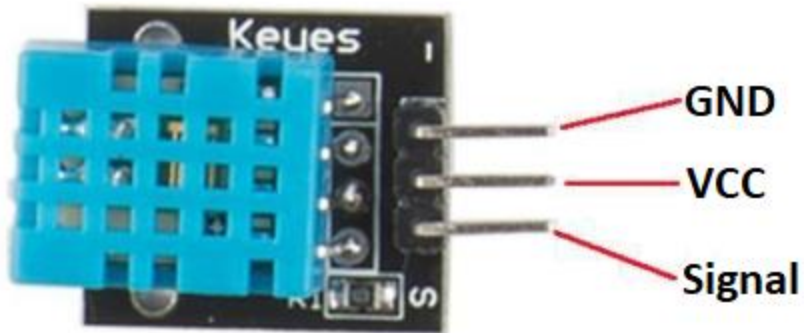
```
  int range = map(sensorReading, sensorMin, sensorMax, 0, 3);
```

```
  // range value:
```

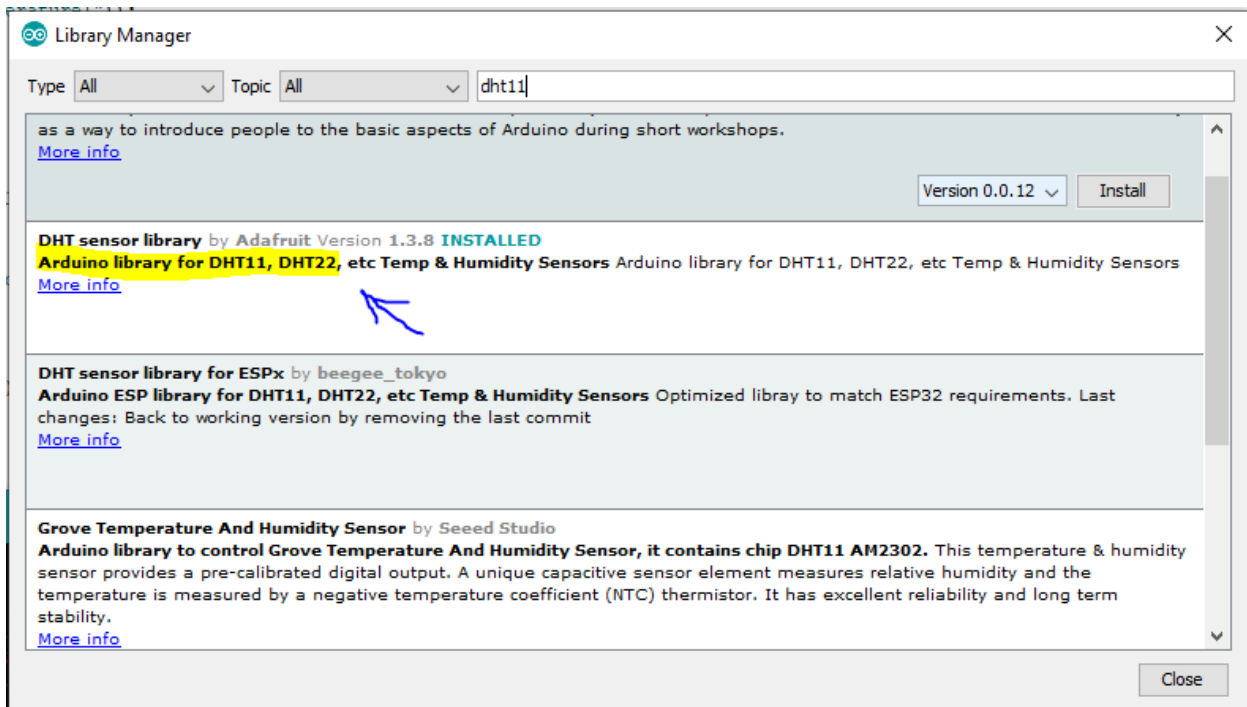


```
switch (range) {  
  case 0: // A fire closer than 1.5 feet away.  
    Serial.println("*** Close Fire ***");  
    break;  
  case 1: // A fire between 1-3 feet away.  
    Serial.println("*** Distant Fire ***");  
    break;  
  case 2: // No fire detected.  
    Serial.println("No Fire");  
    break;  
}  
delay(1); // delay between reads  
}
```

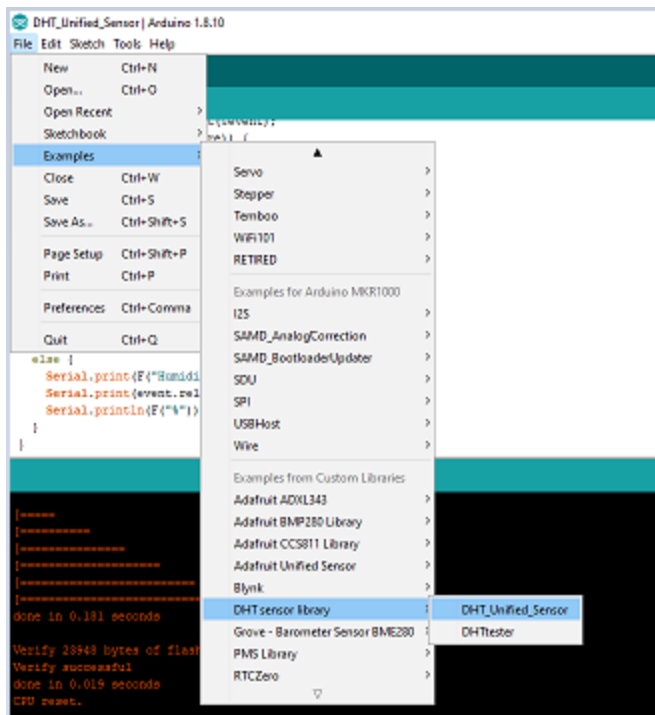
Sensori i lagështisë dhe temperaturës



Për ta përdorur këtë modul së pari lidhim pinat e tij siç është e specifikuar më lartë. Pastaj duhet ta instalojmë librarinë e gatshme për DHT11 emri i modulit. Për ta instaluar librarinë e sensorit shkojmë te **Tools > Library Manager**, kërkohet për **DHT11** dhe instalojmë librarinë përkatëse.



Pasi ta instalojmë librarinë tani kemi shembujt përkatës për ta përdorur sensorin, dhe mund ta hapim njërin nga shembujt duke shkuar te : **File > Example > DHT sensor library > DHT_unified_sensor**.



Pasi ti ndjekim hapat e shënuar më lartë, hapet një dritare e arduinos së bashku me kodin për ta kontrolluar sensorin. Tani duhet ti bëjmë edhe disa ndryshime përfundimtare, e para duhet ti komentojmë të gjitha llojet e sensorëve që nuk jemi duke i përdorur, në mënyrë që kodi të funksionoj vetëm për DHT11, dhe e dyta të shënojmë në kod se ku e kemi lidhur pinin **Signal** të modullit të DHT11.

```
#include <Adafruit_Sensor.h>
#include <DHT.h>
#include <DHT_U.h>

#define DHTPIN 2 // Digital pin connected to the DHT sensor ←
// Feather HUZZAH ESP8266 note: use pins 3, 4, 5, 12, 13 or 14 --
// Pin 15 can work but DHT must be disconnected during program upload.

// Uncomment the type of sensor in use:
#define DHTTYPE DHT11 // DHT 11
// #define DHTTYPE DHT22 // DHT 22 (AM2302) ←
// #define DHTTYPE DHT21 // DHT 21 (AM2301)

// See guide for details on sensor wiring and usage:
// https://learn.adafruit.com/dht/overview

DHT_Unified dht(DHTPIN, DHTTYPE);
```

Detyrë

- 1) Duhet të krijoni një projekt IoT me MKR1000, varet se çka dëshironi të krijoni, dhe çka dëshironi të shfaqni me projektin tuaj. Kjo pjesë duhet të përmbajë lidhjet harduerike të së paku tre moduleve-sensorëve së bashku në MKR1000 dhe kodin për kontrollimin e së paku tre moduleve së bashku. Aplikacioni gjithashtu duhet ta ketë pjesën e konfigurimit dhe monitorimit të të dhënave nga webi.
- 2) Pastaj pasi ta përfundoni pjesën e parë ju duhet që gjithçka që keni bërë më lartë ta dokumentoni në një dokument Word të shkruar. Dokumentimi i punës përfshinë një shpjegim të thjeshtë se çka keni dëshiruar të tregoni me projektin, foto të lidhjeve të sensorëve, kodin që keni shkruar, mënyrën e paraqitjes së rezultateve dhe gjithçka tjetër që mendoni që do ta prezantonte më së miri projektin tuaj.

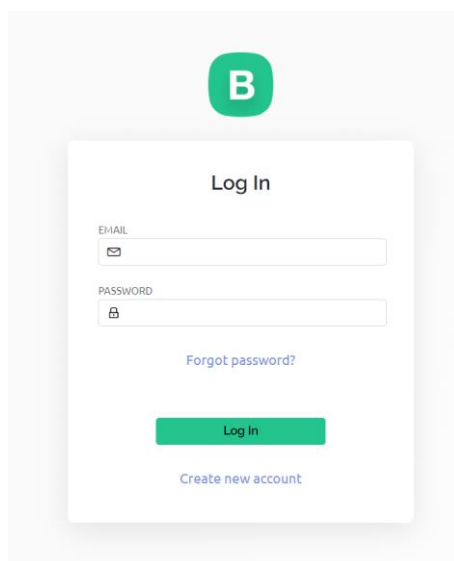
Përdorimi i Blynk

Monitorimi i temperaturës dhe lagështisë së ajrit nëpërmjet telefonit

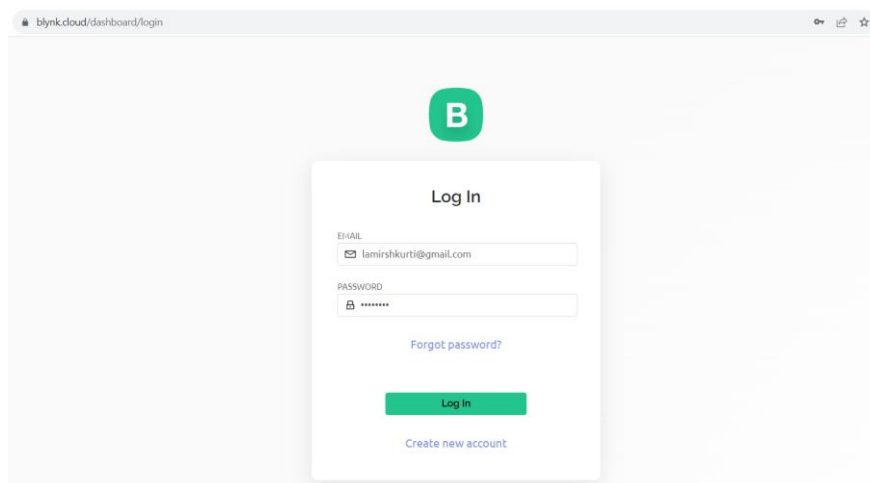
Blynk është një platformë për zhvilluesit e pajisjeve IoT (Internet of Things). Platforma lejon zhvilluesit të krijojnë aplikacione mobile për të kontrolluar dhe monitoruar pajisjet e tyre të lidhura në internet. Blynk ofron një platformë të lehtë për përdorim dhe një aplikacion mobil për të kontrolluar pajisjet. Përmes Blynk, mund të krijoni ndërfaqe të përshtatshme për pajisjet tuaja të lidhura, të shpërndani komandat dhe të merrni informacion mbi statusin e tyre. Për të përdorur platformën, zhvilluesit përdorin bibliotekat dhe API-të e ofruara nga Blynk për të lidhur pajisjet e tyre me serverat e Blynk.

Më posht do të gjeni një shembull se si mund ta përdorim këtë platformë

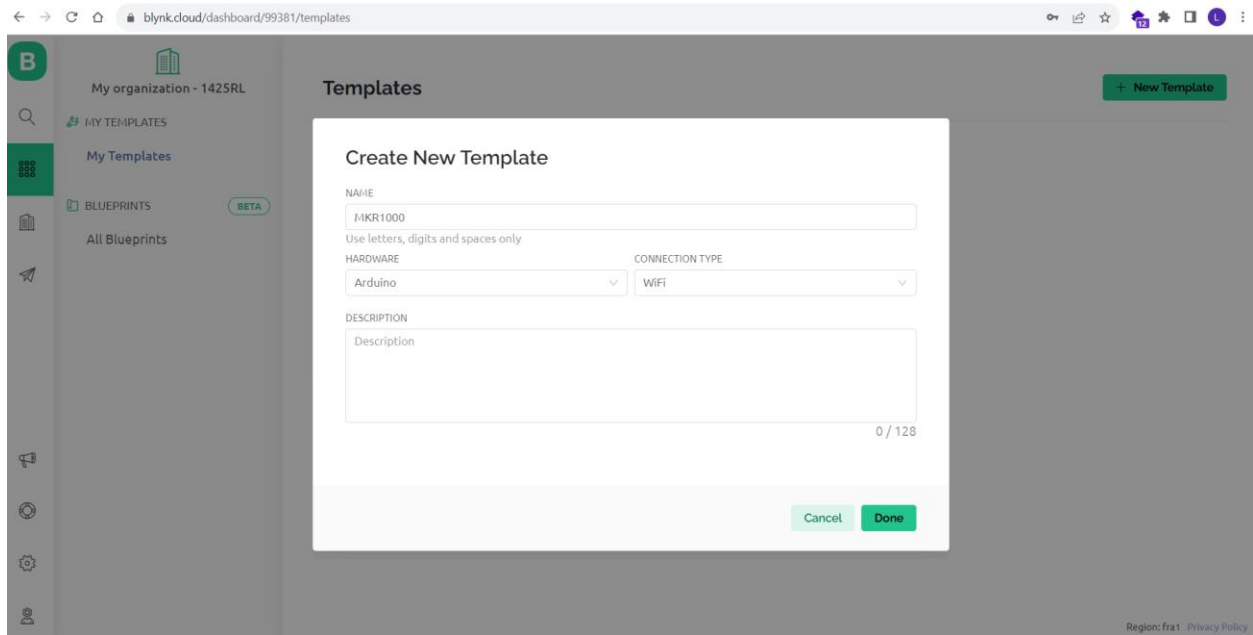
Hapeni <https://blynk.cloud/> dhe krijoni një llogari duke klikuar Create new account.



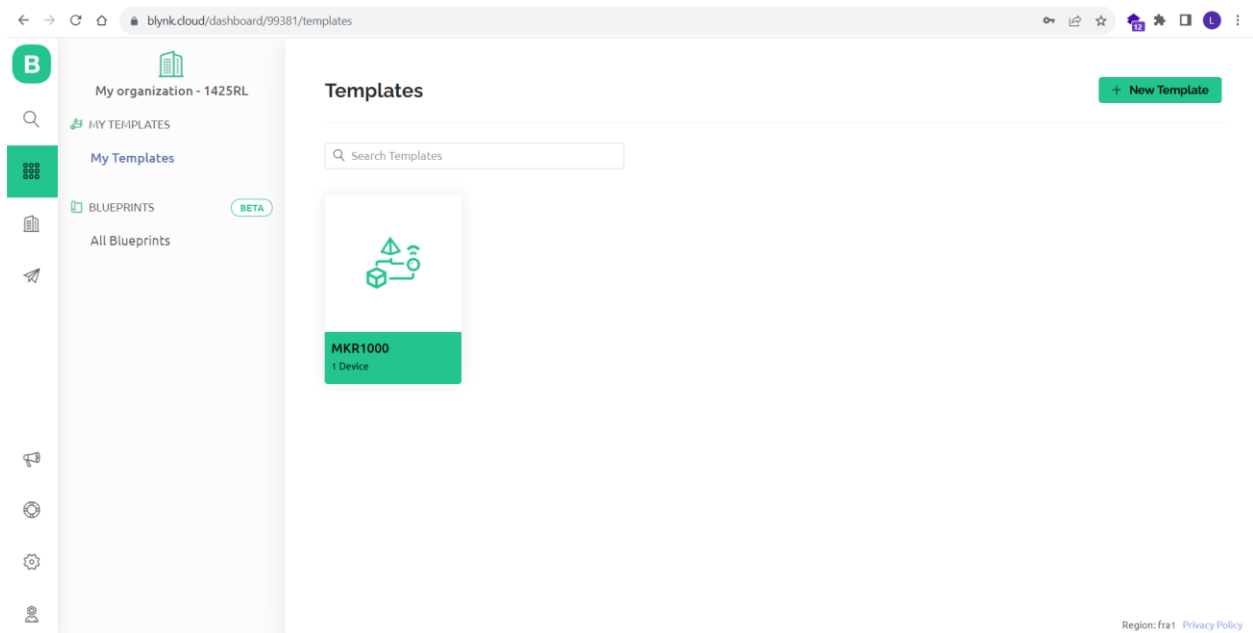
Pasi ta keni krijuar një llogari ne Blynk dhe keni konfirmuar emailin , kyçemi në llogarinë e hapur duke e dhënë emailin dhe passwordin.



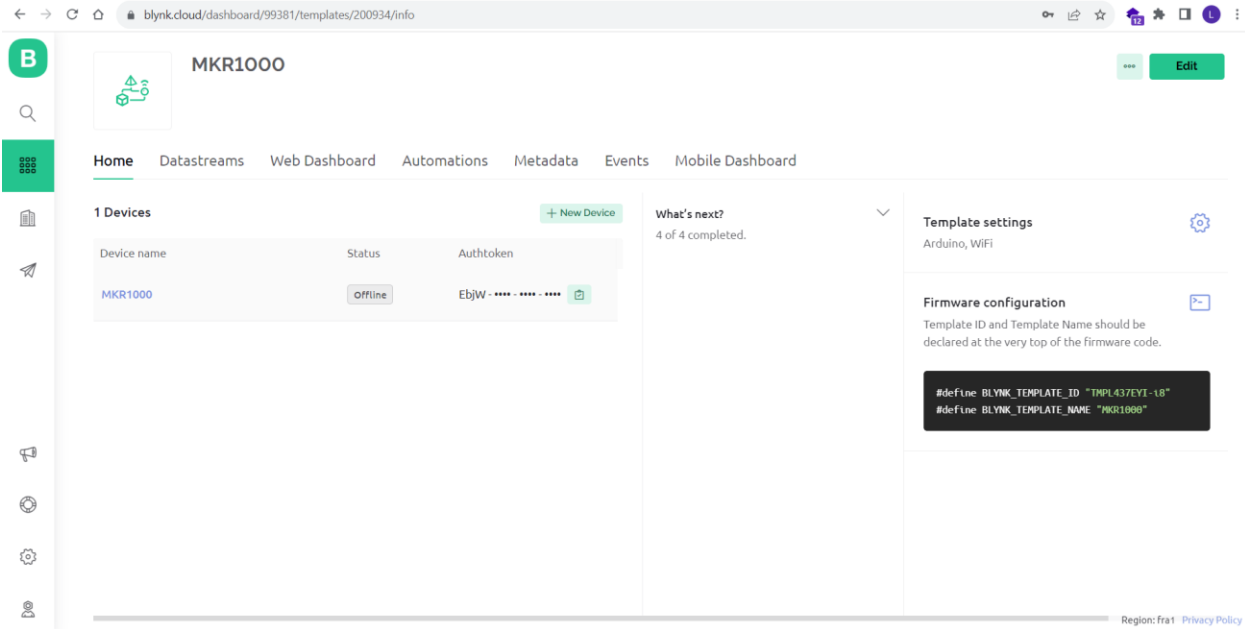
Pasi të jemi kyçur krijojmë një pajisje duke klikuar tek butoni New Template. Mbushim fushat si Name, Hardware dhe Connection Type dhe klikojmë Done.



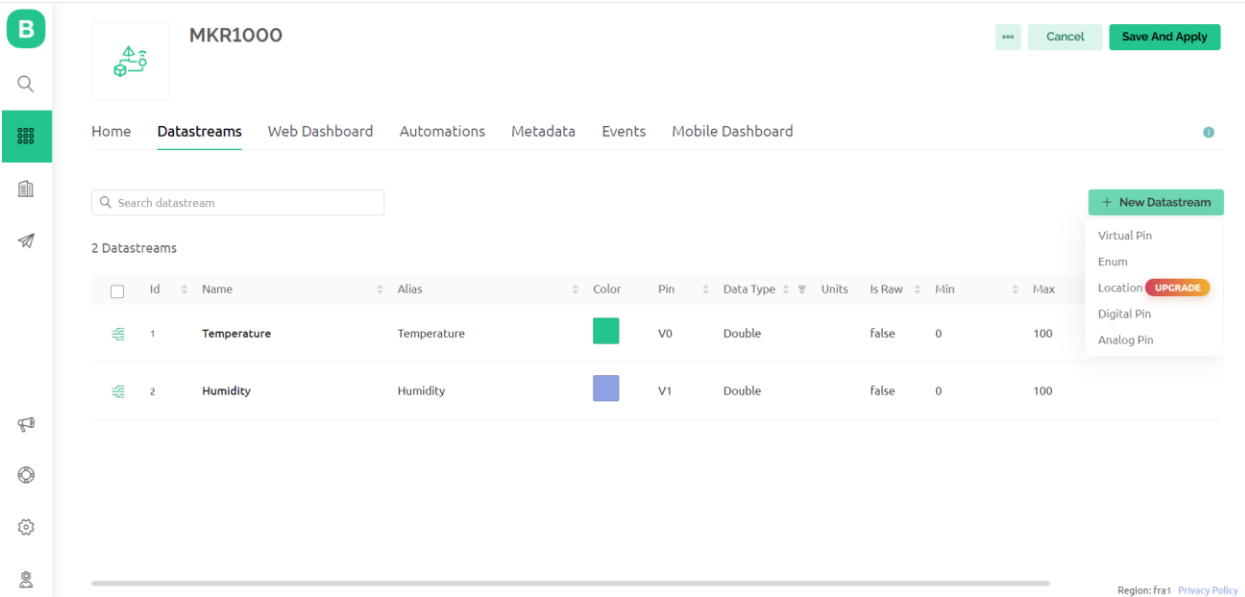
Pasi është krijuar pajisja do të shfaqet si ne figurën e mëposhtme



Pasi të klikojmë tek pajisja e krijuar do të shfaqet figura si më poshtë:



Tek regjistri Datastreams krijojmë dy datastream duke klikuar në butonin New Data stream -> Virtual pin sikurse në figurën e mëposhtme.

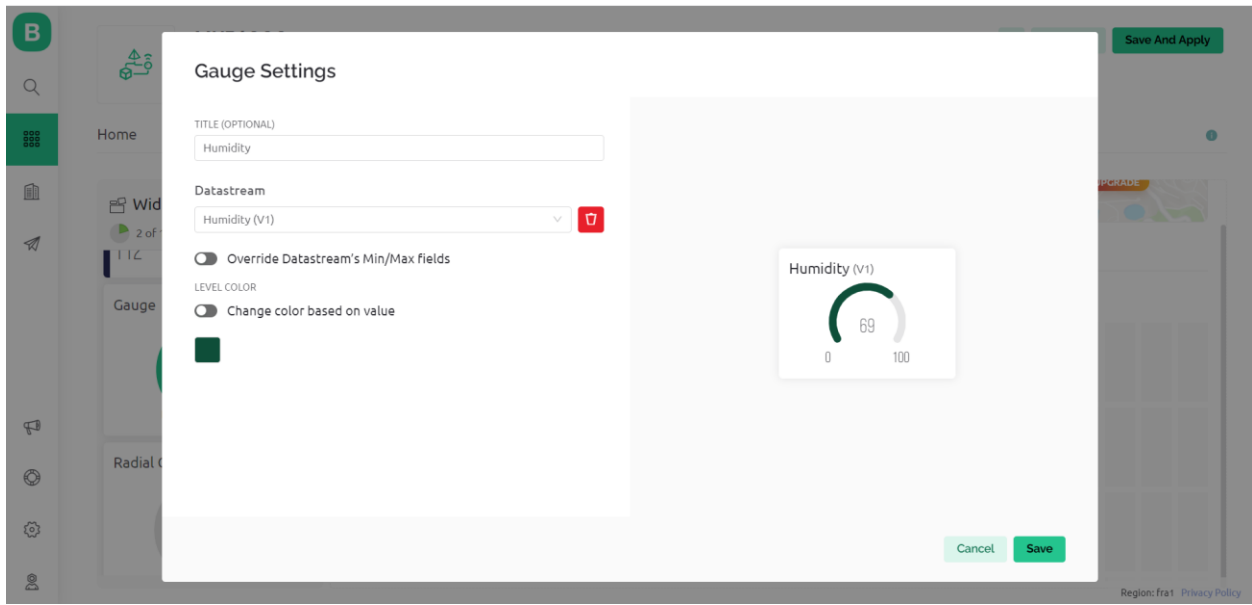


Pasi kemi krijuar datastreams dhe i kemi ruajtur ato, klikojmë tek regjistri Web Dashboard në anën e djathtë tek pjesa Widget Box zgjedhim dy Gauge si në figurë:

The screenshot shows the MKR1000 dashboard interface. At the top, there is a search bar and a 'Save And Apply' button. The navigation menu includes Home, Datastreams, Web Dashboard (selected), Automations, Metadata, Events, and Mobile Dashboard. The main content area features a 'Widget Box' on the left with a 'Gauge' widget showing a value of 42 and a 'Radial Gauge' widget also showing 42. The central area displays 'Device name' as 'Online' with fields for 'Device Owner' and 'Company Name'. Below this, there is a 'Dashboard' section with a time range selector set to 'Last Hour' and two gauge widgets: 'Temperature (V0)' showing 98 and 'Humidity (V1)' showing 69. A map is visible in the top right corner with a 'Show map' toggle and an 'UPGRADE' button. The bottom right corner indicates 'Region: fra1' and 'Privacy Policy'.

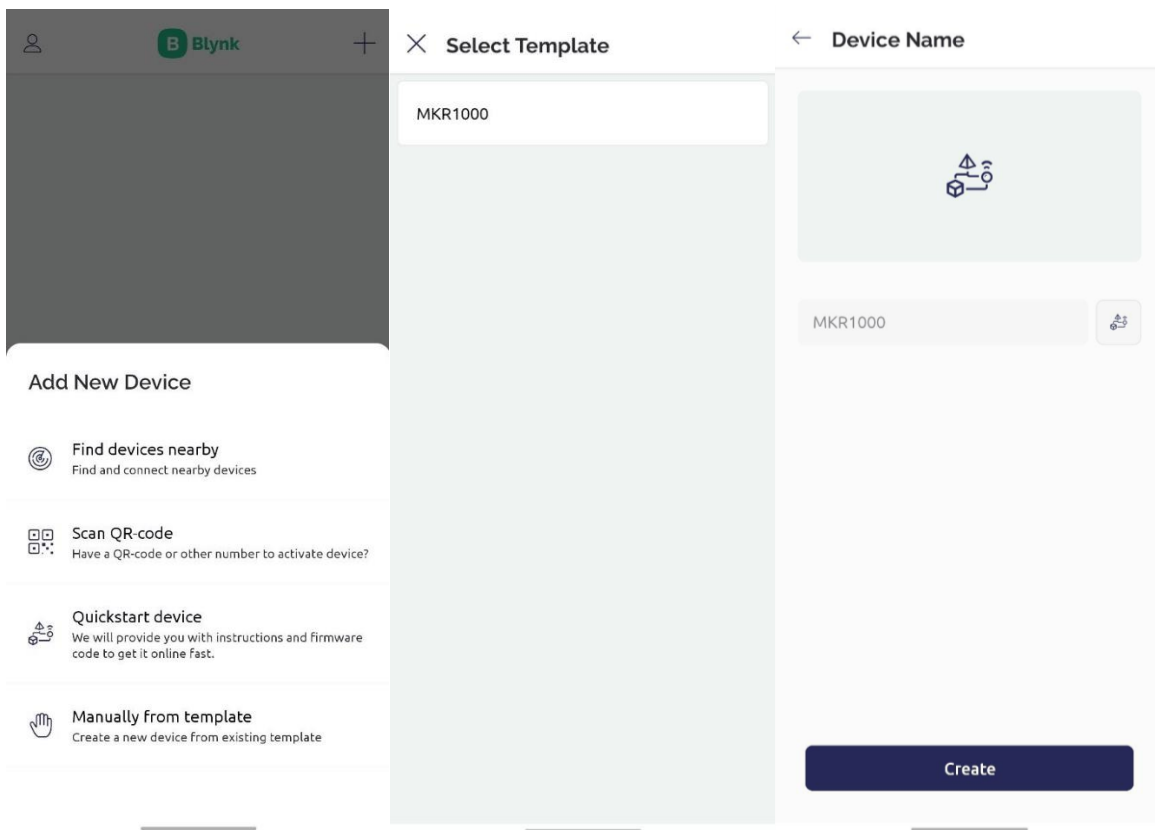
Gauge Temperature lidhet me datastream-in që e krijuam më lartë pra me V0 kurse Humidity lidhet me datastream-in V1 si në figurat e mëposhtme:

The screenshot shows the 'Gauge Settings' dialog box. The 'TITLE (OPTIONAL)' field is set to 'Temperature'. The 'Datastream' dropdown is set to 'Temperature (V0)'. There is a checkbox for 'Override Datastream's Min/Max fields' which is currently unchecked. Under 'LEVEL COLOR', there is a checkbox for 'Change color based on value' which is checked, and a green color swatch is shown. A preview of the gauge widget is displayed on the right, showing the value 98. The dialog box has 'Cancel' and 'Save' buttons at the bottom right. The background shows the same dashboard as the previous image.

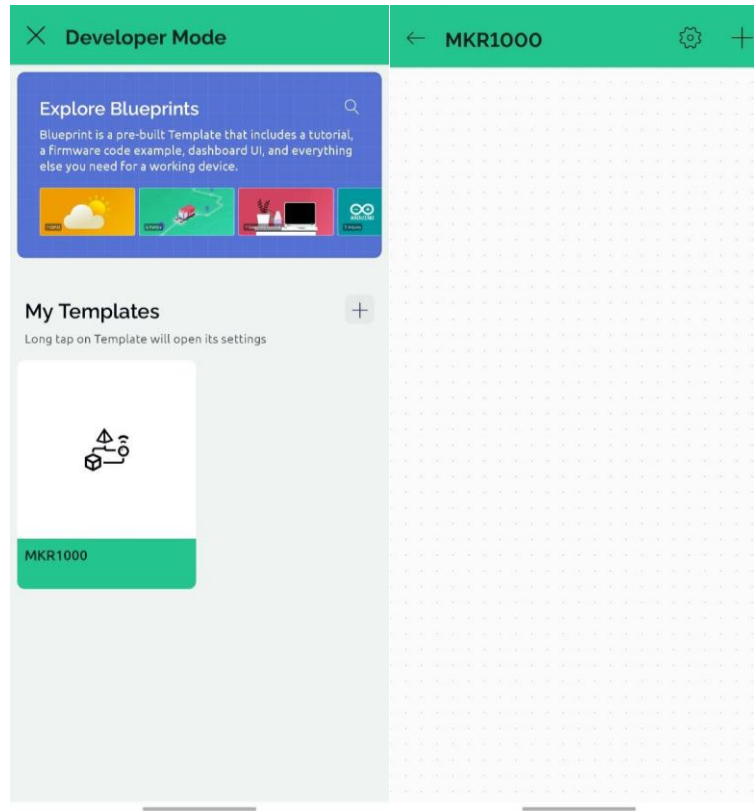


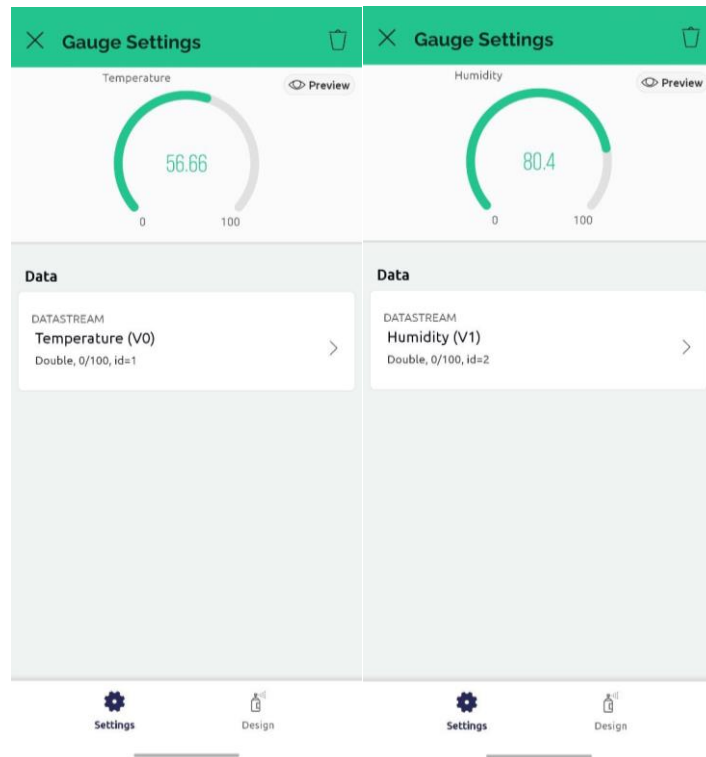
Dhe ne fund klikojmë tek butoni Save And Apply.

Tani instalohet aplikacioni në telefonin mobil. Pasi ta hapim aplikacionin klikojmë në + në këndin e sipërm të djathtë pastaj klikojmë Manually from template hapet një dritare e re ku do të na shfaqet template-i që e kemi krijuar ne Blynk pasi e zgjedhim klikojmë butonin create si më poshtë

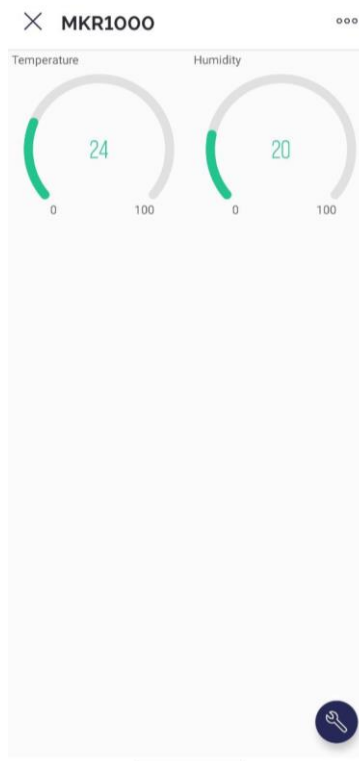


Pastaj klikojmë tek template MKR100 që sapo e krijuam dhe shtojmë me + elementet(kontrollat) të cilat do ti lidhim me Datastream që i krijuam në web tek llogaria në Blynk pra pasi e shtojmë kontrollën Gauge tek settings e kësaj kontrollë e zgjedhim datastream Temperature(V0) dhe tek kontrollat e dytë veprojmë njejt vetëm se e zgjedhim datastream Humidity(V1) si në figurat e mëposhtme

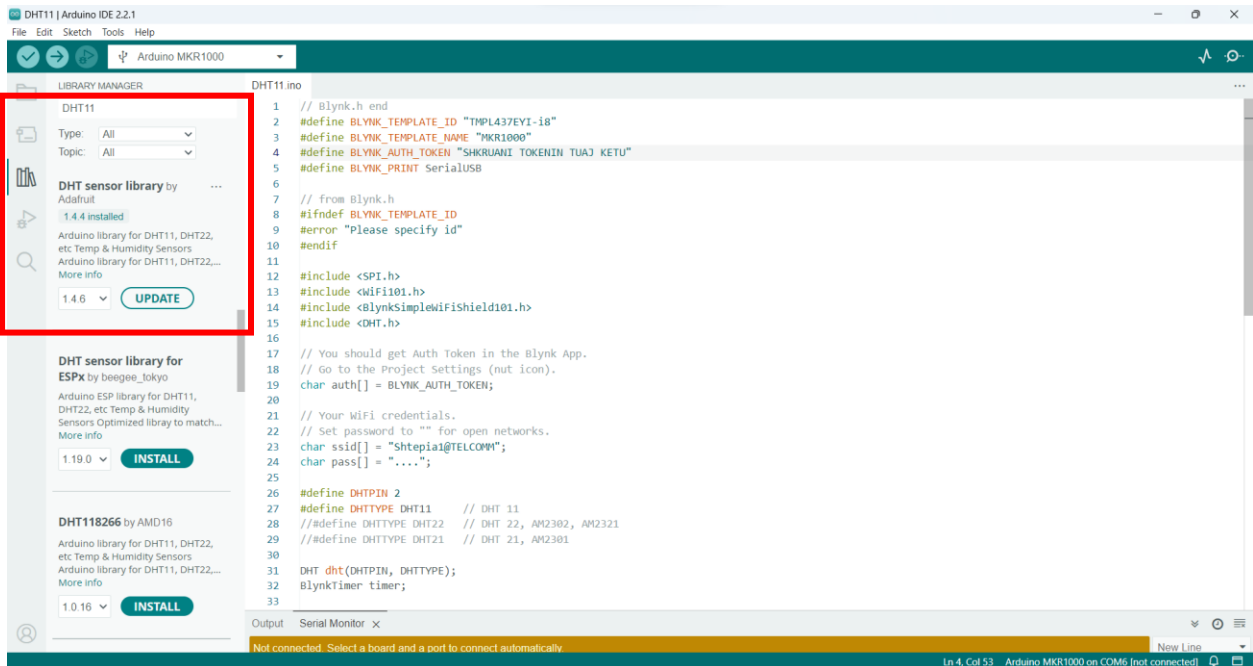
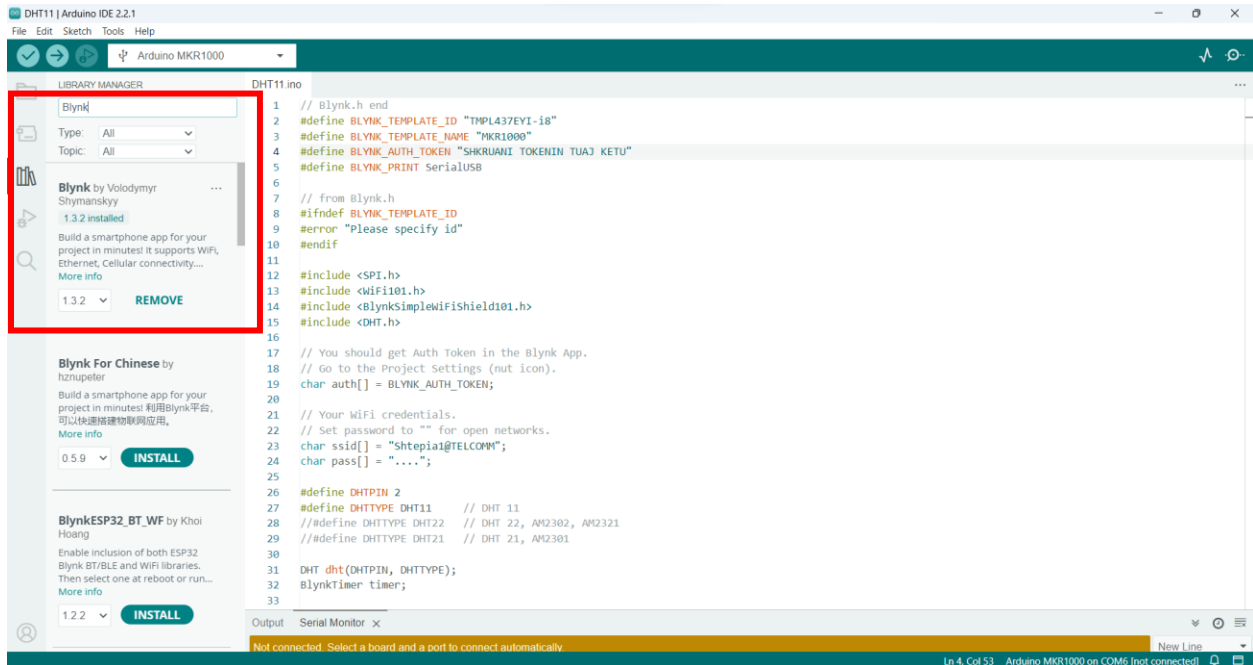




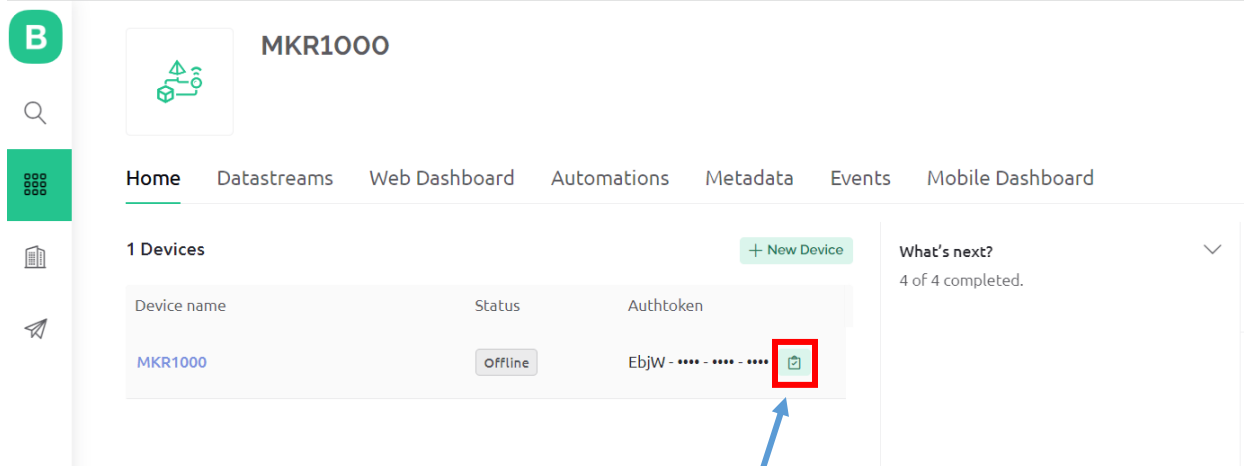
dhe në fund pasi ta kemi ngarkuar kodin e më poshtëm në arduino mkr1000 vlerat do të mirën nga sensorët dhe do të dërgohen tek platforma Blynk të cilat pastaj mund ti shikojm nga aplikacioni që e krijuam si në figurën e mëposhtme



Për të funksionuar kodi i mëposhtëm në Arduino MKR1000 duhet instaluar librarinë Blynk dhe DHT11. Pas instalimit të librarive duhet ngarkuar kodin tek pajisja.



Para se ta ngarkoni kodin merni tokenin e juaj nga pajisja qe e keni krijuar në platformën blynk, si dhe specifikoni emrin dhe passwordin e WIFI-së që do ta shfrytëzoj pajisja.



```
// Blynk.h end
#define BLYNK_TEMPLATE_ID "TMPL437EYI-i8"
#define BLYNK_TEMPLATE_NAME "MKR1000"
#define BLYNK_AUTH_TOKEN "KOPJONI TOKENIN TUAJ KETU"
#define BLYNK_PRINT SerialUSB

// from Blynk.h
#ifndef BLYNK_TEMPLATE_ID
#error "Please specify id"
#endif

#include <SPI.h>
#include <WiFi101.h>
#include <BlynkSimpleWiFiShield101.h>
#include <DHT.h>

// You should get Auth Token in the Blynk App.
// Go to the Project Settings (nut icon).
char auth[] = BLYNK_AUTH_TOKEN;

// Your WiFi credentials.
// Set password to "" for open networks.
char ssid[] = "KREN";
char pass[] = "krenkosova";

#define DHTPIN 2
#define DHTTYPE DHT11 // DHT 11
// #define DHTTYPE DHT22 // DHT 22, AM2302, AM2321
// #define DHTTYPE DHT21 // DHT 21, AM2301
```

```

DHT dht(DHTPIN, DHTTYPE);
BlynkTimer timer;

void sendSensor()
{
  float h = dht.readHumidity();
  float t = dht.readTemperature(); // or dht.readTemperature(true) for Fahrenheit

  if (isnan(h) || isnan(t)) {
    Serial.println("Failed to read from DHT sensor!");
    return;
  }
  // You can send any value at any time.
  // Please don't send more that 10 values per second.
  Blynk.virtualWrite(V0, t);
  Blynk.virtualWrite(V1, h);
}

void setup()
{
  // Debug console
  SerialUSB.begin(9600);

  Blynk.begin(auth, ssid, pass);
  // You can also specify server:
  //Blynk.begin(auth, ssid, pass, "blynk-cloud.com", 80);
  //Blynk.begin(auth, ssid, pass, IPAddress(192,168,1,100), 8080);

  dht.begin();

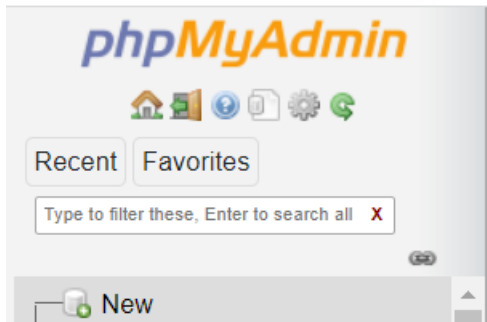
  // Setup a function to be called every second
  timer.setInterval(1000L, sendSensor);
}

void loop()
{
  Blynk.run();
  // You can inject your own code or combine it with other sketches.
  // Check other examples on how to communicate with Blynk. Remember
  // to avoid delay() function!
  timer.run();
}

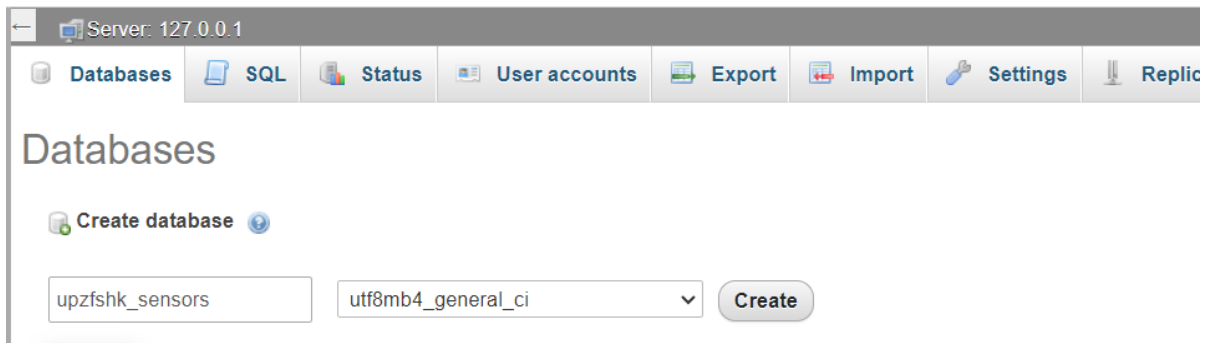
```

Dërgimi i të dhënave në bazën e të dhënave nëpërmjet një Ueb API nga pajisja MKR1000
Krijimi i bazës së të dhënave në MySQL

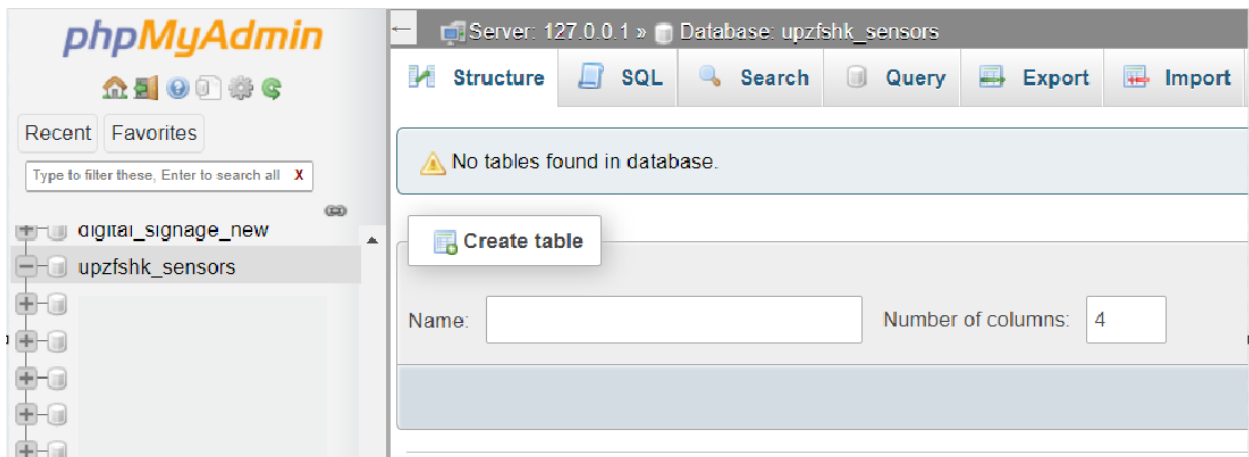
Pasi ta hapim phpMyAdmin në anën e majtë tek paneli klikojmë New sikurse në figurën e mëposhtme



Pasi të klikojmë na shfaqet pjesa ku duhet shkruajmë emrin e bazës së të dhënave që do ta krijojmë.

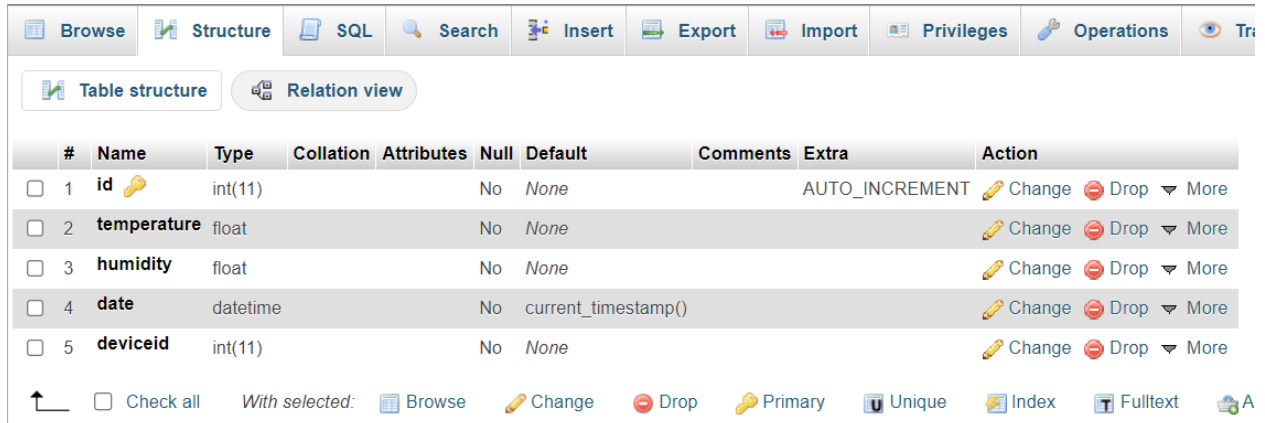


Pasi ta shkruajmë emrin e bazës së të dhënave klikojmë Create dhe do të shohim bazën e të dhënave të krijuar.



Tani në anën e djathtë poshtë create table tek Name shkruajmë emrin e tabelës që do ta krijojmë psh dataset dhe klikojmë butonin Go.

Pasi krijohet tabela dataset i shtojmë fushat sikurse në figurën e mëposhtme, pra tabela do ta ketë këtë strukturë:



#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/>	1	id			No	None		AUTO_INCREMENT	Change Drop More
<input type="checkbox"/>	2	temperature			No	None			Change Drop More
<input type="checkbox"/>	3	humidity			No	None			Change Drop More
<input type="checkbox"/>	4	date			No	current_timestamp()			Change Drop More
<input type="checkbox"/>	5	deviceid			No	None			Change Drop More

Pasi krijojmë bazën e të dhënave dhe tabelën dataset e krijojmë Ueb API duke krijuar fajllin index.php i cili do të ketë këtë përmbajtje:

```
<?php

// Lidhja me bazën e të dhënave
$host = "localhost";
$username = "shkruani emrin e userit në databazë";
$password = "shkruani passwordin e userit";
$database = "shkruani emrin e databazës";

$connection = new mysqli($host, $username, $password, $database);

// Kontrolloni lidhjen
if ($connection->connect_error) {
    die("Lidhja dështoi: " . $connection->connect_error);
}

$temperature = isset($_REQUEST['temperature']) ? $_REQUEST['temperature'] : null;
$humidity = isset($_REQUEST['humidity']) ? $_REQUEST['humidity'] : null;
$deviceid = isset($_REQUEST['deviceid']) ? $_REQUEST['deviceid'] : null;

// Merrni të dhënat nga POST request
$data = json_decode(file_get_contents("php://input"), true);

// Kryerja e validimeve
if (isset($temperature) && isset($humidity) && isset($deviceid)) {
    //if (isset($data['temperature']) && isset($data['humidity']) &&
    //isset($data['deviceid'])) {
```



```

    $query = "INSERT INTO dataset (temperature, humidity, deviceid) VALUES (?, ?,
?)";
    $statement = $connection->prepare($query);
    // $statement->bind_param("ddi", $data['temperature'], $data['humidity'],
    $data['deviceid']);
    $statement->bind_param("ddi", $temperature, $humidity, $deviceid);

    // Kryeni query
    if ($statement->execute()) {
        echo "Të dhënat janë vedosur me sukses!";
    } else {
        echo "Gabim gjatë futjes së të dhënave: " . $statement->error;
    }

    // Mbyllni deklaratën dhe lidhjen me bazën e të dhënave
    $statement->close();
    $connection->close();
} else {
    echo "Të dhënat nuk janë në formatin e duhur.";
}
?>

```

Tani do të tregojmë mënyrën e dërgimit të të dhënave nga pajisja MKR100 tek baza e të dhënave.

Hapim Arduino IDE dhe shkruajmë kodin e mëposhtëm: Mos harroni ta shkruani SSID dhe Passwordin e wifi-së në të cilin do të lidhet pajisja, gjithashtu edhe serverin ku do të dërgohen të dhënat i cili mund te jete localhost apo një server në internet.

```

#include <WiFi101.h> // for MKR1000

#include "DHT.h"
#include <Wire.h>

#define DHTPIN 2
#define DHTTYPE DHT11
DHT dht(DHTPIN, DHTTYPE);

char ssid[] =      "KREN";           // network to join
char pass[] =      "krenkosova";     // password for wifi network

int status = WL_IDLE_STATUS;

char server[] = "upz-fshk.online";   // server URL

String postData;

```

```

String postVariable = "depth=";

WiFiClient client;

void setup() {

    // Start serial connection for feedback
    Serial.begin(9600);
    dht.begin();

    // Connect to WiFi
    while (status != WL_CONNECTED) {
        Serial.print("Attempting to connect to: ");
        Serial.println(ssid);
        status = WiFi.begin(ssid, pass);
        delay(10000);           // wait 10 sec for connection
    }

    printWiFiStatus();
}

void loop() {
    // Readings
    int depth = 500;
    int bottleId = 11;

    postData = postVariable + depth;

    // Connect to php script at port 80
    if (client.connect(server, 80)) {
        Serial.println("Connected to server.");

        int temp = dht.readTemperature();
        int hum = dht.readHumidity();

        client.println("POST
/index.php?temperature="+String(temp)+"&humidity="+String(hum)+"&deviceid=1
HTTP/1.1");
        client.print("Host: ");
        client.println(server);
        client.println("User-Agent: ArduinoWiFi/1.1");
        client.println("Connection: close");
        client.println("Content-Type: application/x-www-form-urlencoded;");
        client.print("Content-Length: ");
        String postEntity = String("");

```

```

    client.println(postEntity.length());
    client.println();
    client.println(postEntity);

    Serial.println("Data are saved.");
    delay(5000);

} else {
    Serial.println("Failed to connect to server.");
}

if (client.connected()) {
    client.stop();
}
Serial.println(postData);

delay(3000);
}

void printWiFiStatus() {

    // print the SSID of the network you're attached to:
    Serial.print("SSID: ");
    Serial.println(WiFi.SSID());

    // print your board's IP address:
    IPAddress ip = WiFi.localIP();
    Serial.print("IP Address: ");
    Serial.println(ip);

    // print your subnet mask:
    IPAddress subnet = WiFi.subnetMask();
    Serial.print("NETMASK: ");
    Serial.println();

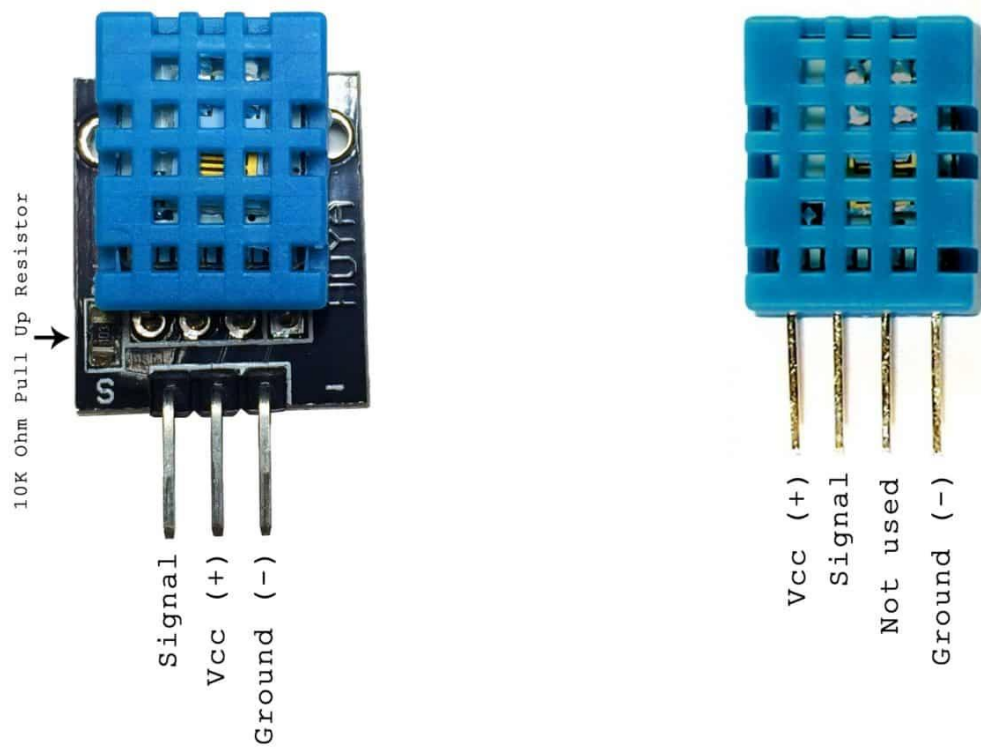
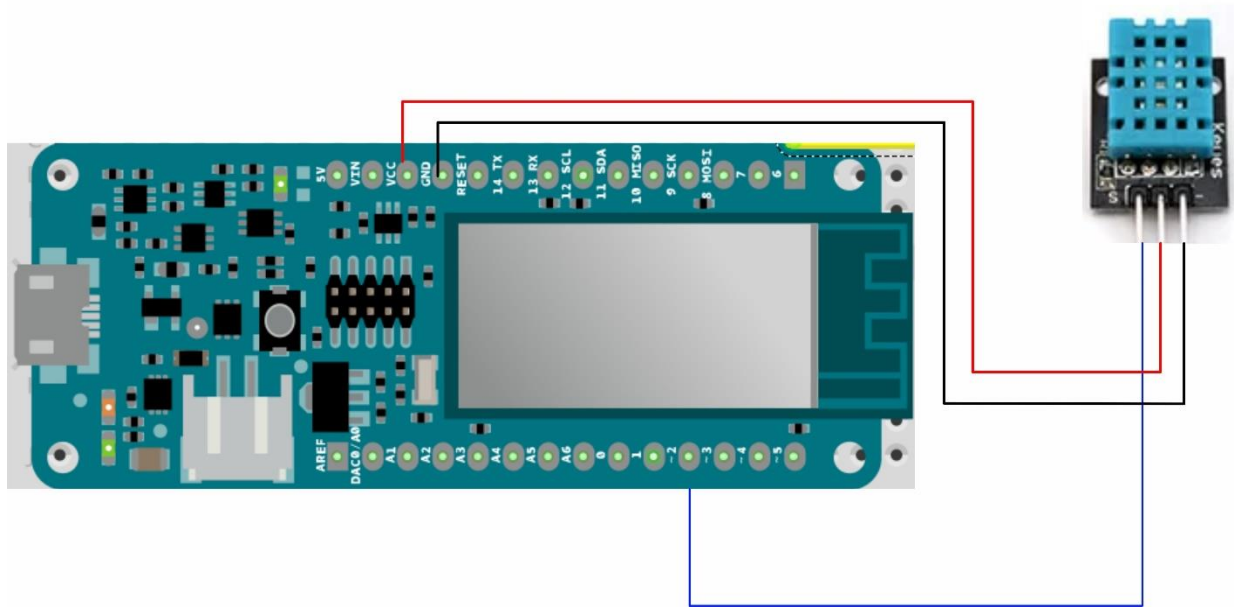
    // print your gateway address:
    IPAddress gateway = WiFi.gatewayIP();
    Serial.print("GATEWAY: ");
    Serial.println(gateway);

    // print the received signal strength:
    long rssi = WiFi.RSSI();
    Serial.print("signal strength (RSSI):");
    Serial.print(rssi);
    Serial.println(" dBm");
}

```

}

Kodin e ngarkojmë tek Arduino MKR1000 i cili do të ketë të lidhur sensorin DHT11 për matjen e temperaturës dhe lagështisë së ajrit.



Pasi kodi ngarkohet tek pajisja MKR1000. Të dhënat e marura nga sensori do të regjistrohen në bazën e të dhënave tek tabela dataset.

Marrja e të dhënave nga baza e të dhënave nëpërmjet një Ueb API dhe komandimi i pajisjes MKR1000 nga të dhënat e marura

Pasi ta hapim phpMyAdmin në anën e majtë tek databaza kemi krijuar tabelën led sikurse në figurën e mëposhtme.

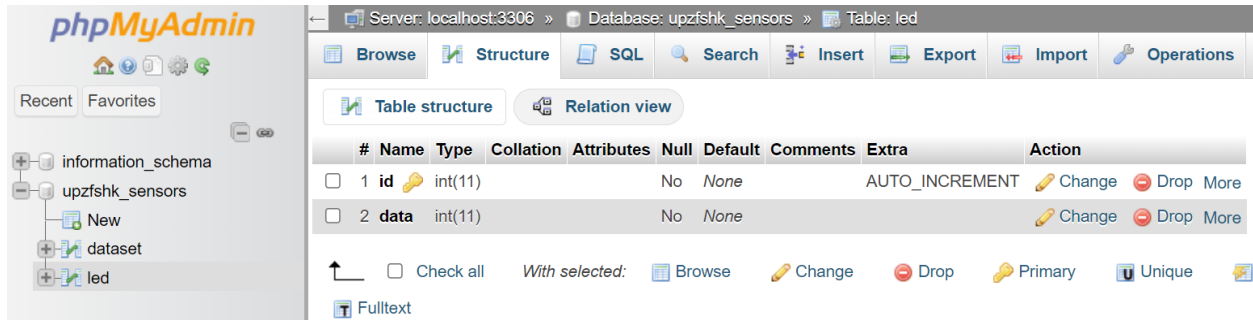


Tabela ka dy fusha id dhe data. Në data do të ruajmë vlerën led-it 1 ose 0 e cila korrespondon 1 me ndezur dhe 0 e fikur.

Pasi krijojmë tabelën led e krijojmë Ueb API në fajllin index.php ekzistues duke shtuar dy funksionalitete të reja për marrjen e vlerës së data nga tabela led dhe shfaqjen e saj si json dhe funksionaliteti i dytë i cili bënë update vlerën e data të led-it. Fajlli index.php do të ketë këtë përmbajtje:

```
<?php

// Lidhja me bazën e të dhënave
$host = "localhost";
$username = "user";
$password = "password";
$databse = "upzfshk_sensors";

$connection = new mysqli($host, $username, $password, $databse);

// Kontrolloni lidhjen
if ($connection->connect_error) {
    die("Lidhja dështoi: " . $connection->connect_error);
}

$temperature = isset($_REQUEST['temperature']) ? $_REQUEST['temperature'] : null;
$humidity = isset($_REQUEST['humidity']) ? $_REQUEST['humidity'] : null;
$deviceid = isset($_REQUEST['deviceid']) ? $_REQUEST['deviceid'] : null;

$ledValue = isset($_REQUEST['led']) ? $_REQUEST['led'] : null;

$ledInsertValue = isset($_REQUEST['ledValue']) ? $_REQUEST['ledValue'] : null;
```

```

// Merrni të dhënat nga POST request
$data = json_decode(file_get_contents("php://input"), true);

// Kryerja e validimeve
if (isset($temperature) && isset($humidity) && isset($deviceid)) {
//if (isset($data['temperature']) && isset($data['humidity']) &&
isset($data['deviceid'])) {

    $query = "INSERT INTO dataset (temperature, humidity, deviceid) VALUES (?, ?,
?)" ;
    $statement = $connection->prepare($query);
//    $statement->bind_param("ddi", $data['temperature'], $data['humidity'],
$data['deviceid']);
    $statement->bind_param("ddi", $temperature, $humidity, $deviceid);

    // Kryeni query
    if ($statement->execute()) {
        echo "Të dhënat janë vedosur me sukses!";
    } else {
        echo "Gabim gjatë futjes së të dhënave: " . $statement->error;
    }

    // Mbyllni deklaratën dhe lidhjen me bazën e të dhënave
    $statement->close();
    $connection->close();
} else if(isset($ledValue)){

    $q = mysqli_query($connection,"select * from led ORDER BY `id` ASC");

    $data['rows'] = array();
    while($r = mysqli_fetch_array($q))
    {
        $arr = array();
        $arr["id"] = $r['id'];
        $arr["data"] = $r['data'];

        array_push($data["rows"], $arr);
    }
    $veri = json_encode($data["rows"]);
    echo $veri;
    //echo $data["rows"][0]["data"];//$veri;
}

```

```

// Kryerja e validimeve
if (isset($ledInsertValue)) {

    $query = "UPDATE led SET data = ?";
    $statement = $connection->prepare($query);

    $statement->bind_param("i", $ledInsertValue);

    // Kryeni query
    if ($statement->execute()) {
        echo "Të dhënat janë vedosur me sukses!";
    } else {
        echo "Gabim gjatë futjes së të dhënave: " . $statement->error;
    }

    // Mbyllni deklaratën dhe lidhjen me bazën e të dhënave
    $statement->close();
    $connection->close();
}
else {
    echo "Të dhënat nuk janë në formatin e duhur.";
}
?>

```

Gjithashtu kemi krijuar edhe një fajll led.html i cili bënë update vlerën e data tek tabela led. Fajlli led.html ka ketë përmbajtje:

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Forma e Sensorit</title>
</head>
<body>

    <h2>Zgjedhni ON per ta leshuar led-in ndersa Off per ta ndalur:</h2>

    <form action="index.php" method="post">
        <input type="radio" id="on" name="ledValue" value="1">
        <label for="on">On</label>

        <br>

```

```

        <input type="radio" id="off" name="ledValue" value="0">
        <label for="off">Off</label>

        <br><br>

        <input type="submit" value="Submit">
    </form>

</body>
</html>

```

Tani do të tregojmë mënyrën e marrjes së të të dhënave nga databaza dhe ndezjen apo fikjen e ledit me pajisjen MKR100 duke u bazuar në vlerat e marura nga databaza.

Hapim Arduino IDE dhe shkruajmë kodin e mëposhtëm: Mos harroni ta shkruani SSID dhe Passwordin e wifi-së në të cilin do të lidhet pajisja, gjithashtu edhe serverin ku do të dërgohen të dhënat i cili mund të jete localhost apo një server në internet.

```

#include <WiFi101.h> // for MKR1000

char ssid[] = "KREN"; // your network SSID (name)
char pass[] = "krenkosova"; // your network password (use for WPA, or use as
key for WEP)

int ledPin = 2;

int status = WL_IDLE_STATUS;

//IPAddress server(74,125,232,128); // numeric IP for Google (no DNS)
char server[] = "upz-fshk.online"; // name address for Google (using DNS)
String teksti = "";

WiFiClient client;

void setup() {

    //Initialize serial and wait for port to open:

    Serial.begin(9600);
    pinMode(ledPin, OUTPUT);
    while (!Serial) {

        ; // wait for serial port to connect. Needed for native USB port only

```



```
}

// attempt to connect to Wifi network:

while (status != WL_CONNECTED) {

    Serial.print("Attempting to connect to SSID: ");

    Serial.println(ssid);

    status = WiFi.begin(ssid, pass);

    // wait 10 seconds for connection:
    delay(10000);

}

Serial.println("Connected to wifi");

printWifiStatus();
lidhuPerseri();

}

void loop() {

    while (client.available()) {

        char c = client.read();
        String txt = String(c);
        teksti += txt;
        // Serial.print(c);
        Serial.write(c);

    }

    if (!client.connected()) {

        Serial.println();

        // client.stop();
        Serial.println(teksti);

    }

}
```

```

// Gjej fillimin dhe fundin e pjesës JSON
int json_start = teksti.indexOf("[");
int json_end = teksti.indexOf("]", json_start) + 1;

// Nxerrni pjesën JSON nga përgjigja HTTP
String json_data = teksti.substring(json_start, json_end);

// Printoni pjesën JSON në Serial Monitor
Serial.print("JSON data: ");
Serial.println(json_data);

int data_start = json_data.indexOf("\"data\":") + 8;
int data_end = json_data.indexOf("\"", data_start);
String data_value = json_data.substring(data_start, data_end);

// Printoni vlerën e "data" në Serial Monitor
Serial.print("Data value: ");
Serial.println(data_value);
if (data_value == "1") {
    digitalWrite(ledPin, HIGH);
}
if (data_value == "0") {
    digitalWrite(ledPin, LOW);
}

// do nothing forevermore:
teksti = "";

// while (true);
delay(3000);

lidhuPerseri();

}
}

void lidhuPerseri(){
    Serial.println("\nStarting connection to server...");

    // if you get a connection, report back via serial:

    if (client.connect(server, 80)) {

        Serial.println("connected to server");
    }
}

```

```
// Make a HTTP request:

client.println("GET /index.php?led HTTP/1.1");

client.print("Host: ");
client.println(server);

client.println("Connection: close");

client.println();

}
}

void printWifiStatus() {

// print the SSID of the network you're attached to:

Serial.print("SSID: ");

Serial.println(WiFi.SSID());

// print your board's IP address:

IPAddress ip = WiFi.localIP();

Serial.print("IP Address: ");

Serial.println(ip);

// print the received signal strength:

long rssi = WiFi.RSSI();

Serial.print("signal strength (RSSI):");

Serial.print(rssi);

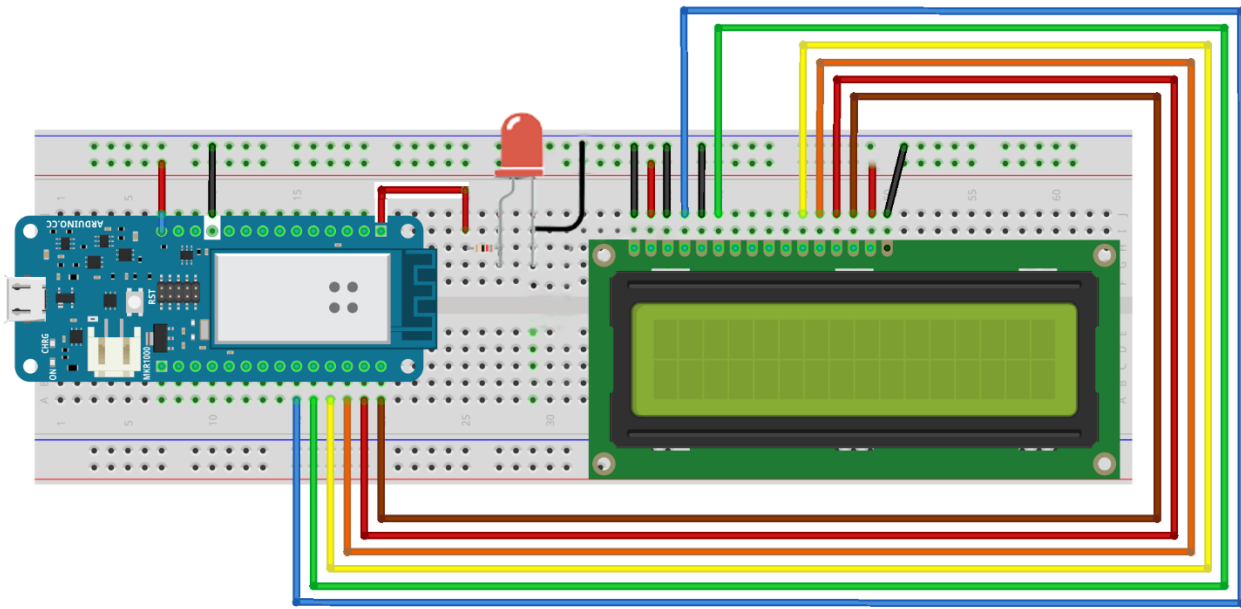
Serial.println(" dBm");
}
```

Përdorimi i MKR 1000 si një Ueb Server

Ueb Serveri do të përdoret si një ndërfaqe për bordin tone MKR100, ku do të krijojmë dy butona për të ndezur ose fikur nga distanca një LED dhe shkruarjen e tekstit në mënyrë dinamike në një LCD ekran nga uebi.

Skema e lidhjes së LCD dhe LED-it:

Sigurohuni që keni instaluar bibliotekat WiFi101 dhe LiquidCrystal



Kodi në Arduino:

```
#include <SPI.h>
#include <WiFi101.h>
#include <LiquidCrystal.h>

LiquidCrystal lcd(0, 1, 2, 3, 4, 5);

char ssid[] = "KREN";           // your network SSID (name) between the " "
char pass[] = "krenkosova";    // your network password between the " "
int keyIndex = 0;              // your network key Index number (needed only
for WEP)
int status = WL_IDLE_STATUS;  //connection status
WiFiServer server(80);        //server socket

WiFiClient client = server.available();
```

```

int ledPin = 6;
String teksti = "";

void setup() {
  Serial.begin(9600);
  pinMode(ledPin, OUTPUT);
  while (!Serial);

  lcd.begin(16, 2); // If you have 20x4 LCD user (20,4)

  enable_WiFi();
  connect_WiFi();

  server.begin();
  printWifiStatus();
}

void loop() {
  client = server.available();

  if (client) {
    printWEB();
  }
}

void printWifiStatus() {
  // print the SSID of the network you're attached to:
  Serial.print("SSID: ");
  Serial.println(WiFi.SSID());

  // print your board's IP address:
  IPAddress ip = WiFi.localIP();
  Serial.print("IP Address: ");
  Serial.println(ip);

  // print the received signal strength:
  long rssi = WiFi.RSSI();
  Serial.print("signal strength (RSSI):");
  Serial.print(rssi);
  Serial.println(" dBm");

  Serial.print("To see this page in action, open a browser to http://");
  Serial.println(ip);
}

```

```

void enable_WiFi() {

    String fv = WiFi.firmwareVersion();
    if (fv < "1.0.0") {
        Serial.println("Please upgrade the firmware");
    }
}

void connect_WiFi() {
    // attempt to connect to Wifi network:
    while (status != WL_CONNECTED) {
        Serial.print("Attempting to connect to SSID: ");
        Serial.println(ssid);
        // Connect to WPA/WPA2 network. Change this line if using open or WEP
network:
        status = WiFi.begin(ssid, pass);

        // wait 10 seconds for connection:
        delay(10000);
    }
}

void printWEB() {

    if (client) {
        Serial.println("new client");
        String currentLine = "";
        while (client.connected()) {
            if (client.available()) {
                char c = client.read();
                String txt = String(c);
                teksti += txt;
                Serial.write(c);
                if (c == '\n') {
                    // if you get a client,
                    // print a message out the serial port
                    // make a String to hold incoming data from the client
                    // loop while the client's connected
                    // if there's bytes to read from the client,
                    // read a byte, then
                    // print it out the serial monitor
                    // if the byte is a newline character

                    // if the current line is blank, you got two newline characters in a row.
                    // that's the end of the client HTTP request, so send a response:
                    if (currentLine.length() == 0) {

                        // HTTP headers always start with a response code (e.g. HTTP/1.1 200 OK)
                        // and a content-type so the client knows what's coming, then a blank line:
                        client.println("HTTP/1.1 200 OK");
                        client.println("Content-type:text/html");
                        client.println();
                    }
                }
            }
        }
    }
}

```

```

        //create the buttons
        client.print("Click <a href=\"/H\">here</a> turn the LED on<br>");
        client.print("Click <a href=\"/L\">here</a> turn the LED
off<br><br>");
        // Add input field
        client.print("<form action=\"/text\" method=\"get\">");
        client.print("Text: <input type=\"text\" name=\"inputText\"><br>");
        client.print("<input type=\"submit\" value=\"Submit\">");
        client.print("</form>");

        // The HTTP response ends with another blank line:
        client.println();
        // break out of the while loop:
        break;
    }
    else {        // if you got a newline, then clear currentLine:
        currentLine = "";
    }
}
else if (c != '\r') {    // if you got anything else but a carriage return character,
    currentLine += c;    // add it to the end of the currentLine
}
Serial.println(currentLine);
if (currentLine.endsWith("GET /H")) {
    digitalWrite(ledPin, HIGH);
}
if (currentLine.endsWith("GET /L")) {
    digitalWrite(ledPin, LOW);
}

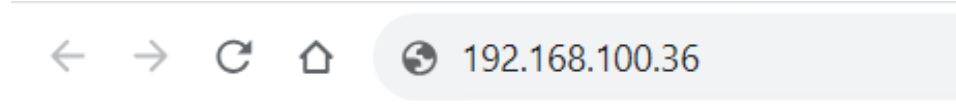
// Check if the request includes text input
if (currentLine.indexOf("GET /text?inputText=") != -1) {
    int startPos = currentLine.indexOf("GET /text?inputText=") + 20;
    int endPos = currentLine.indexOf("HTTP");
    String inputText = currentLine.substring(startPos, endPos);
    Serial.println("Input Text: " + inputText);

    // Display the input text on the LCD
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Teksti i shkruar: ");
    lcd.setCursor(0, 1);
    lcd.print(inputText);
}

```

```
    }  
  }  
  Serial.println("lamir -" + teksti);  
  // close the connection:  
  client.stop();  
  Serial.println("client disconnected");  
}  
}
```

Pasi të kemi ngarkuar me sukses kodin në MKR1000, hapni Serial Monitorin dhe pas lidhjes me wifi do të ju shfaqet IP-adresa e pajisjes. Merrni IP-adresën dhe vendosni në browser dhe do të ju shfaqet dritarja ku mund të ndizni dhe fikni LED-in dhe shkruani tekstin i cili do të shfaqet në display.



Click [here](#) turn the LED on
Click [here](#) turn the LED off

Text:

Përdorimi i JWT(JSON Web Token) për të siguruar shkëmbimin e të dhënave në mënyrë të sigurt

Një bibliotekë e thjeshtë për të koduar dhe deshifruar JSON Web Tokens (JWT) në PHP mund ta gjeni në këtë link: <https://github.com/firebase/php-jwt>

Për ta përdorur këtë librari tek fajlli ku kemi punuar më heret index.php shtojmë këtë kod:

```
require "php-jwt/src/JWT.php";
```

```
require "php-jwt/src/Key.php";
```

Së pari gjenerojmë një token për ta shfrytëzuar më pastaj nga Arduino MKR1000 për ti dërtuar të dhënat së bashku me tokenin që do ta gjenerojmë:

```
<?php
require 'php-jwt/src/JWT.php';
require 'php-jwt/src/Key.php';

use Firebase\JWT\JWT;
use Firebase\JWT\Key;

$secretKey = 'sekreti-lamir';

$tokenData = [
    'id' => '123321'
];
// $tokenData = [
//     'data' => [
//         'id' => '123321'
//         // 'data' => time(),
//         // 'iat': 1703288812,
//         // 'exp': 1703383200
//     ],
// ];
// ];
// $key = base64_decode($secretKey);

$jwt = JWT::encode($tokenData, $secretKey, 'HS256');

$decoded = JWT::decode($jwt, new Key($secretKey, 'HS256'));
print_r($decoded);

echo json_encode(['token' => $jwt]);
?>
```

Tani këtë token e shfrytëzohet në kodin e shkruar në arduino:

```
#include <WiFi101.h> // for MKR1000

#include "DHT.h"
#include <Wire.h>

#define DHTPIN 2
#define DHTTYPE DHT11
DHT dht(DHTPIN, DHTTYPE);

char ssid[] =      "KREN";           // network to join
char pass[] =      "krenkosova";     // password for wifi network

int status = WL_IDLE_STATUS;

char server[] = "upz-fshk.online";   // server URL

String token =
"eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpZCI6IjEyMzMyMSJ9.dASvDkCc-
JeOJjU0bZWgN_IyX1LDUZJRMnLK3Fhya0k";

String postData;
String postVariable = "depth=";

WiFiClient client;

void setup() {

  // Start serial connection for feedback
  Serial.begin(9600);
  dht.begin();

  // Connect to WiFi
  while (status != WL_CONNECTED) {
    Serial.print("Attempting to connect to: ");
    Serial.println(ssid);
    status = WiFi.begin(ssid, pass);
    delay(10000);           // wait 10 sec for connection
  }

  printWiFiStatus();
}
```

```

void loop() {

  // Readings
  int depth = 500;
  int bottleId = 11;

  postData = postVariable + depth;

  // Connect to php script at port 80
  if (client.connect(server, 80)) {
    Serial.println("Connected to server.");

    int temp = dht.readTemperature();
    int hum = dht.readHumidity();

    client.println("POST
/index.php?temperature="+String(temp)+"&humidity="+String(hum)+"&deviceid=1&token
="+String(token)+" HTTP/1.1");
    client.print("Host: ");
    client.println(server);
    client.println("User-Agent: ArduinoWiFi/1.1");
    client.println("Connection: close");
    client.println("Content-Type: application/x-www-form-urlencoded;");
    client.print("Content-Length: ");
    String postEntity = String("");
    client.println(postEntity.length());
    client.println();
    client.println(postEntity);

    Serial.println("Data are saved.");
    delay(5000);

  } else {
    Serial.println("Failed to connect to server.");
  }

  if (client.connected()) {
    client.stop();
  }
  Serial.println(postData);

  delay(3000);
}

void printWiFiStatus() {

```

```

// print the SSID of the network you're attached to:
Serial.print("SSID: ");
Serial.println(WiFi.SSID());

// print your board's IP address:
IPAddress ip = WiFi.localIP();
Serial.print("IP Address: ");
Serial.println(ip);

// print your subnet mask:
IPAddress subnet = WiFi.subnetMask();
Serial.print("NETMASK: ");
Serial.println();

// print your gateway address:
IPAddress gateway = WiFi.gatewayIP();
Serial.print("GATEWAY: ");
Serial.println(gateway);

// print the received signal strength:
long rssi = WiFi.RSSI();
Serial.print("signal strength (RSSI):");
Serial.print(rssi);
Serial.println(" dBm");
}

```

Ndërsa fajlli index.php duhet të jetë si më poshtë:

```

<?php

require "php-jwt/src/JWT.php";
require "php-jwt/src/Key.php";

use Firebase\JWT\JWT;
use Firebase\JWT\Key;

// Lidhja me bazën e të dhënave
$host = "localhost";
$username = "";
$password = "";
$databse = "upzfsk_sensors";

$connection = new mysqli($host, $username, $password, $databse);

```

```

// Kontrolloni lidhjen
if ($connection->connect_error) {
    die("Lidhja dështoi: " . $connection->connect_error);
}

$secretKey = "sekreti-lamir";
$token = $_REQUEST["token"];
$sekreti_id = "123321";

$temperature = isset($_REQUEST["temperature"]) ? $_REQUEST["temperature"] : null;
$humidity = isset($_REQUEST["humidity"]) ? $_REQUEST["humidity"] : null;
$deviceid = isset($_REQUEST["deviceid"]) ? $_REQUEST["deviceid"] : null;

$ledValue = isset($_REQUEST["led"]) ? $_REQUEST["led"] : null;
$ledInsertValue = isset($_REQUEST["ledValue"]) ? $_REQUEST["ledValue"] : null;

// Merrni të dhënat nga POST request
// $data = json_decode(file_get_contents("php://input"), true);

// Kryerja e validimeve
if (isset($temperature) && isset($humidity) && isset($deviceid)) {
    //if (isset($data['temperature']) && isset($data['humidity']) &&
    //isset($data['deviceid'])) {
    try {
        //echo $token . " " . $secretKey;
        // Verifikoni tokenin duke përdorur çelësin e fshehur
        $decoded = JWT::decode($token, new Key($secretKey, "HS256"));

        $dataToInsert = $decoded->id;
        //echo "lsh-" . $dataToInsert;

        if ($dataToInsert == $sekreti_id) {
            $query =
                "INSERT INTO dataset (temperature, humidity, deviceid) VALUES (?,
?, ?)";

            $statement = $connection->prepare($query);
            $statement->bind_param("ddi", $temperature, $humidity, $deviceid);

            // Kryeni query
            if ($statement->execute()) {
                echo "Të dhënat janë vedosur me sukses!";
            } else {
                echo "Gabim gjatë futjes së të dhënave: " . $statement->error;
            }
        }
    }
}

```

```

        $statement->close();
        $connection->close();
    }
} catch (Exception $e) {
    // Nëse ndodh një gabim gjatë verifikimit, tokeni është i pavlefshëm
    echo json_encode([
        "message" =>
            "Tokeni i pavlefshëm ose ka ndodhur një gabim gjatë
verifikimit.",
    ]);
}
} elseif (isset($ledValue)) {
    $q = mysqli_query($connection, "select * from led ORDER BY `id` ASC");

    $data["rows"] = [];
    while ($r = mysqli_fetch_array($q)) {
        $arr = [];
        $arr["id"] = $r["id"];
        $arr["data"] = $r["data"];

        array_push($data["rows"], $arr);
    }
    $veri = json_encode($data["rows"]);
    echo $veri;
    //echo $data["rows"][0]["data");//$veri;
}
// Kryerja e validimeve
if (isset($ledInsertValue)) {
    $query = "UPDATE led SET data = ?";
    $statement = $connection->prepare($query);

    $statement->bind_param("i", $ledInsertValue);

    // Kryeni query
    if ($statement->execute()) {
        echo "Të dhënat janë vedosur me sukses!";
    } else {
        echo "Gabim gjatë futjes së të dhënave: " . $statement->error;
    }

    // Mbyllni deklaratën dhe lidhjen me bazën e të dhënave
    $statement->close();
    $connection->close();
} else {

```

```
echo "Të dhënat nuk janë në formatin e duhur.";  
}  
?>
```