Курсов проект по предмета

„Трансформация с Интернет на обектите“

Изготвил: Стефан Ламбов, ФН: 2321446

Съдържание

[I. Тема на проекта. 3](#_Toc153134153)

[II. Използвани материали. 3](#_Toc153134154)

[III. Основна идея на програмата. 3](#_Toc153134155)

[IV. Описание на портовете. 4](#_Toc153134156)

[V. Свързване на приложението с онлайн платформата Arduino Cloud. 5](#_Toc153134157)

[VI. Описание на кода на програмата: 17](#_Toc153134158)

[VII. Заключение. 26](#_Toc153134159)

# Тема на проекта.

Темата на проекта е „Контрол на вентилатор чрез датчик за температура и влажност и включване на лампичка при засичане на движение чрез сензор за движение.

# Използвани материали.

За целта са използват следните материали:

1. Микроконтролери [Arduino UNO R4 WiFi](https://www.robotev.com/product_info.php?cPath=1_21&products_id=701&osCsid=dnetkksngbi5heq5sr2qmigj43)(основен) и [Arduino UNO R3](https://www.olx.bg/d/ad/arduino-uno-r3-atmel-atmega328-savmestim-s-arduino-CID632-ID9j9cM.html)(помощен, само за допълнителни портове плюс(5V) и минус(GND)).
2. Датчик за температура и влажност [Arduino Clone DHT22](https://www.emag.bg/senzor-za-vlazhnost-i-temperatura-arduino-clone-dht22-cl170/pd/DD006JBBM/).
3. [Едноканално реле 5V](https://www.olx.bg/d/ad/ednokanalno-rele-5v-10a-1ch-ll-CID632-ID9j9fT.html?reason=seller_listing%7Colx_shop_basic).
4. Сензор за движение [Arduino Clone HC-SR505 Mini](https://www.emag.bg/pir-senzor-arduino-clone-hc-sr505-mini-cl247/pd/DXWN6JBBM/).
5. [Вентилатор за компютър аксиален с втулка 5V](https://www.emag.bg/ventilator-za-kompjutyr-aksialen-s-vtulka-5v-0-11a-40h40h10mm-19815/pd/D7N17GBBM/).
6. Лед лампичка.
7. Кабели за свързване.

За изработване на приложението се използва безплатната онлайн платформа за IOT: Arduino Cloud, която може да се използва при наличие на микроконтролер Arduino с WiFi.

Програмата е написана на езика C++. Средата за разработка е Arduino IDE.

# Основна идея на програмата.

Основната идея на програмата е представена в следните стъпки:

* При засичане на движение, светва лампичката и се извеждат следните съобщения:

LED turning ON!

motion detected at 59 sec

LED turned OFF!

motion ended at 69 sec

* При температура в целзий >= 28 градуса се активира вентилаторчето и се деактивира сензорът за движение. Извеждат се следните съобщения:

Humidity: 100.00 % Temperature: 28.00 \*C 82.40 \*F Heat index: 36.38 \*C 97.48 \*F

FAN turning ON!

LED turned OFF!

* При температура в целзий >= 30 градуса да се активират вентилаторчето и лампичката и се деактивира сензорът за движение. Извеждат се следните съобщения:

Humidity: 100.00 % Temperature: 30.30 \*C 86.54 \*F Heat index: 45.68 \*C 114.22 \*F

FAN turning ON!

LED turning ON!

# Описание на портовете.

Описание на използваните портове на Arduino UNO R4 за контрол на устройствата:

PIN 2 -> Датчик за температура и влажност

PIN 3 -> Сензор за движение

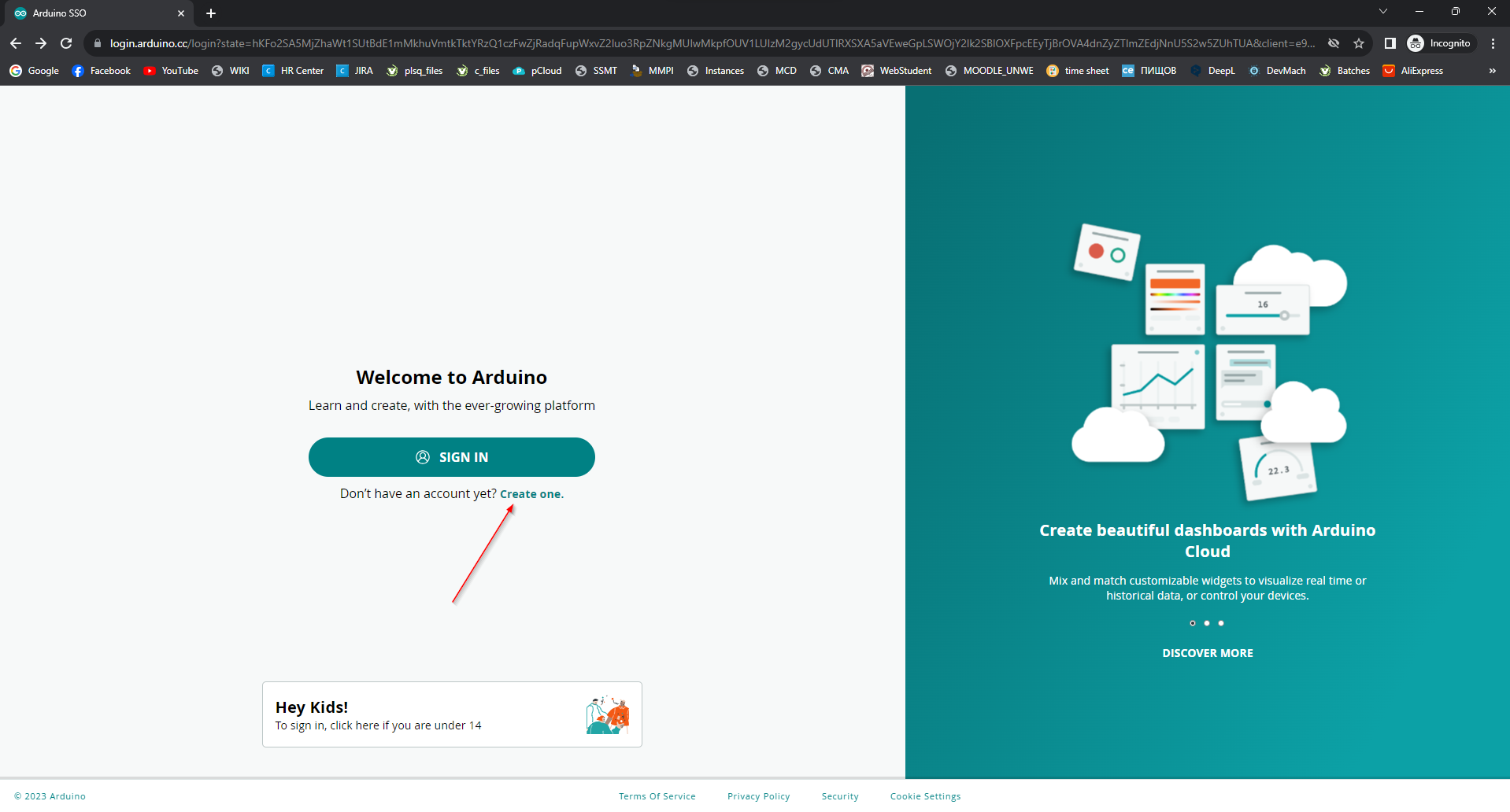
PIN 11 -> Вентилатор 5V

PIN 13 -> Лед лампичка

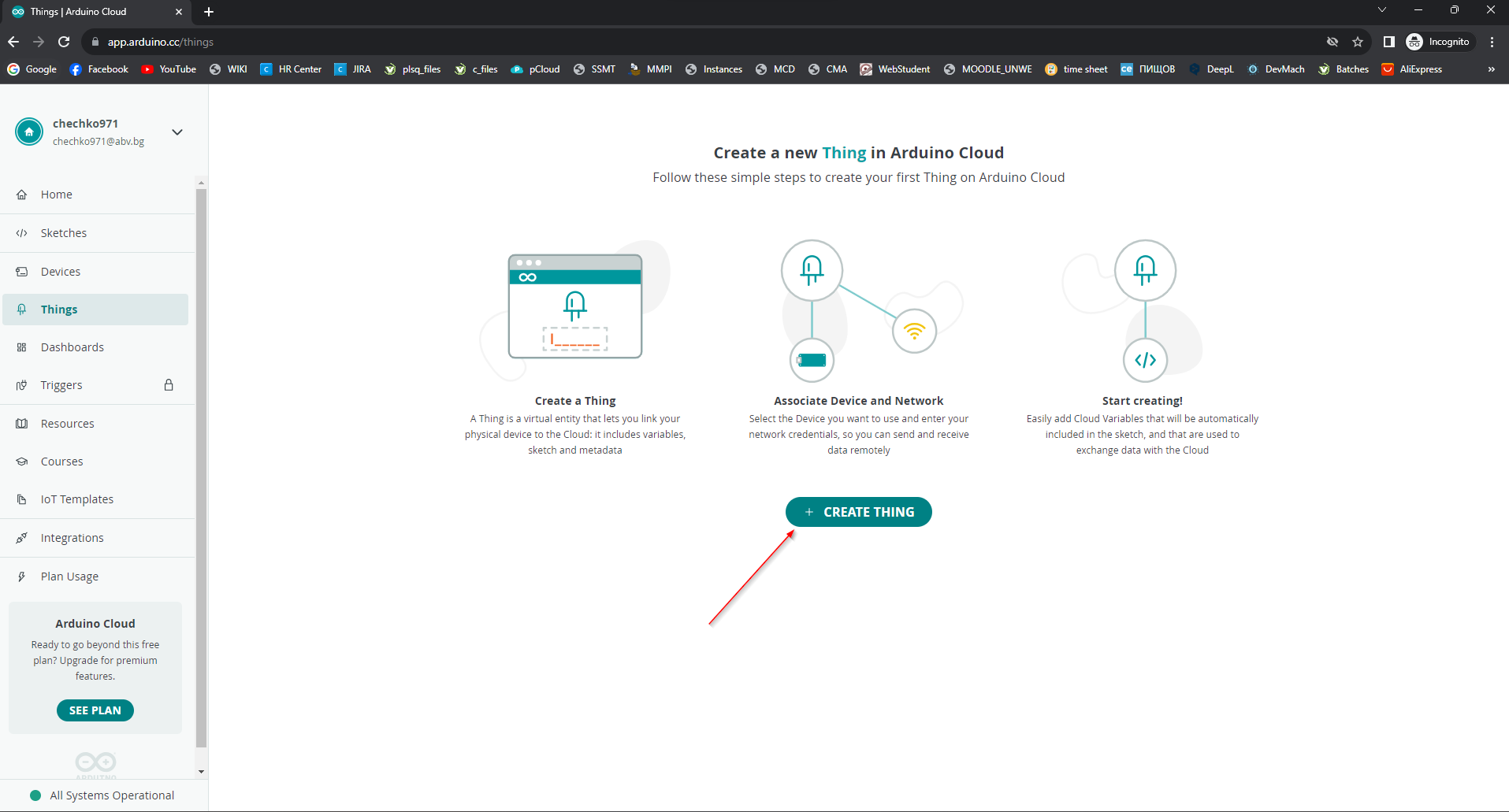
# Свързване на приложението с онлайн платформата Arduino Cloud.

Свързване на приложението с онлайн платформата Arduino Cloud става по следния начин:

1. Създаване на потребителски профил.



1. Създаване на „нещо“: <https://app.arduino.cc/things> с име „UNWE IOT PROJECT“.



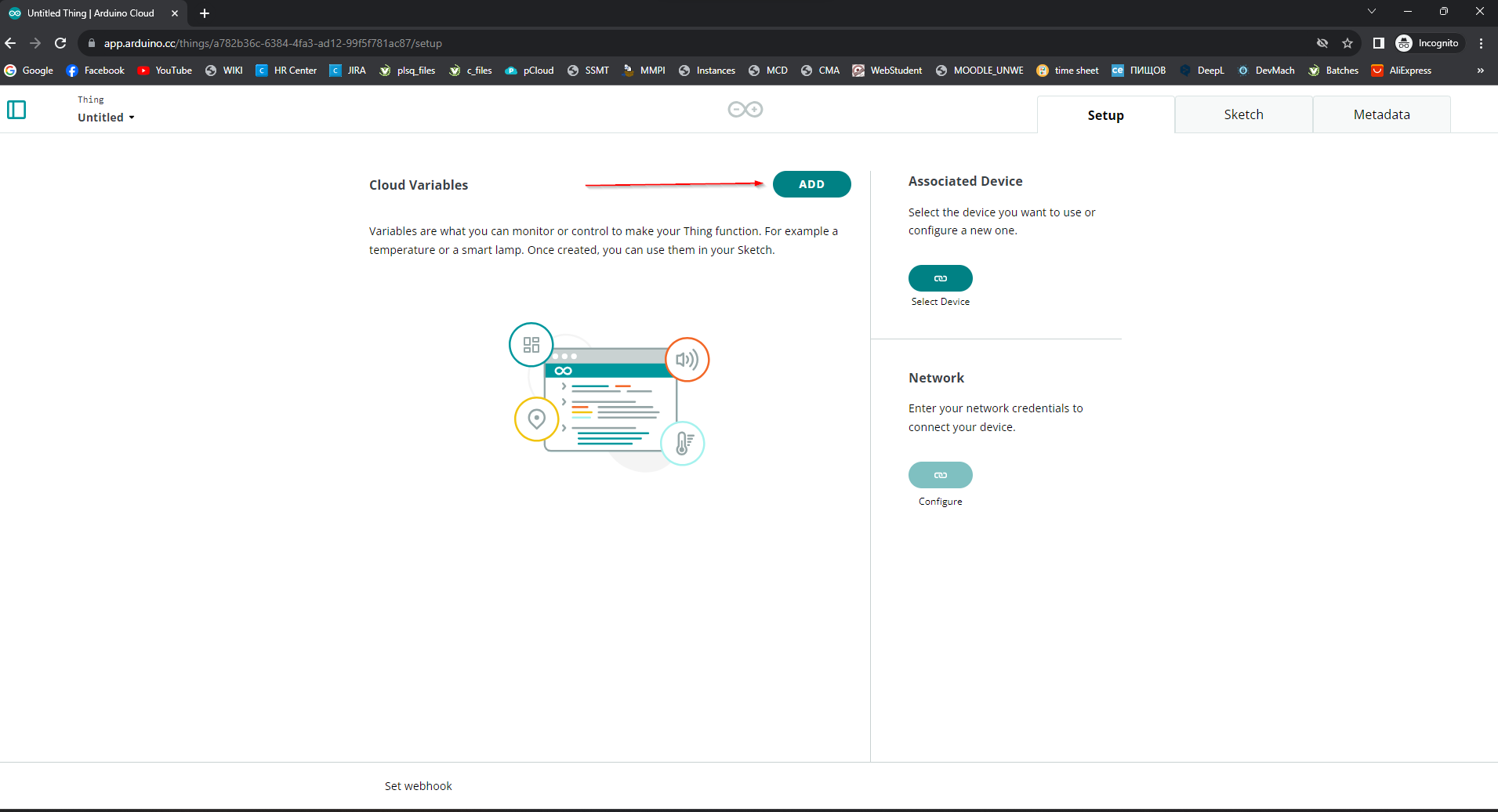
1. Създаване на следните променливи:

bool bulbControl; -> за включване и изключване на лампичката

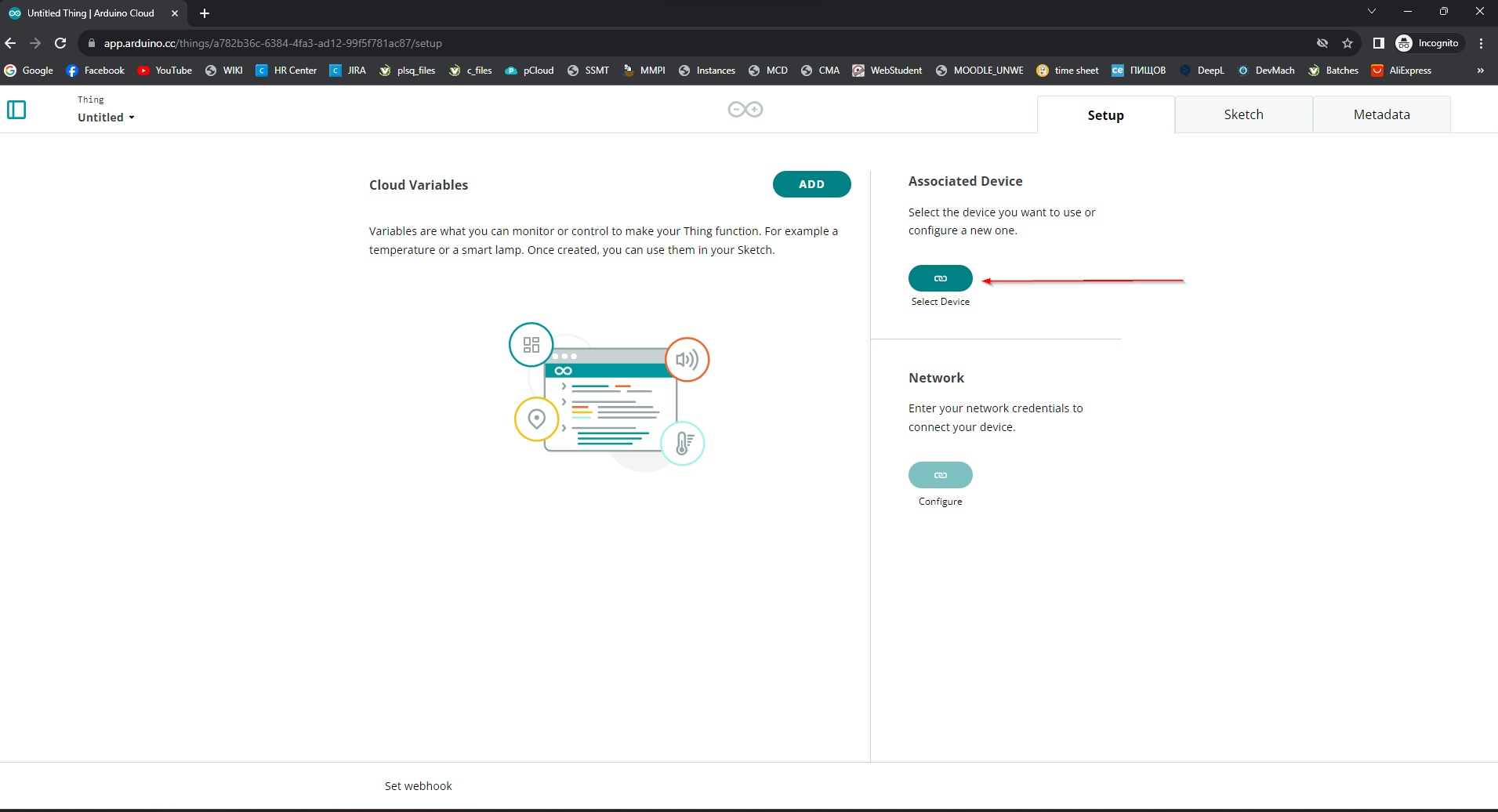
bool fanControl; -> за включване и изключване на вентилатора

bool showTemperature; -> за извеждане на съобщение за температурата и влажността в момента.

float tempCelsius; -> съхраняване на температурата в Целзии и генериране на графика по периоди.



1. Свързване на микроконтролера Arduino UNO R4. Генерира се Secret Key на микроконтролера, който трябва да пазим.



1. Въвеждане на данните на WiFi и Secret Key за връзка с интернет.

A screenshot of a computer

Description automatically generated

1. След това минаваме към поле Sketch, където поставяме кода на програмата.

A screenshot of a computer

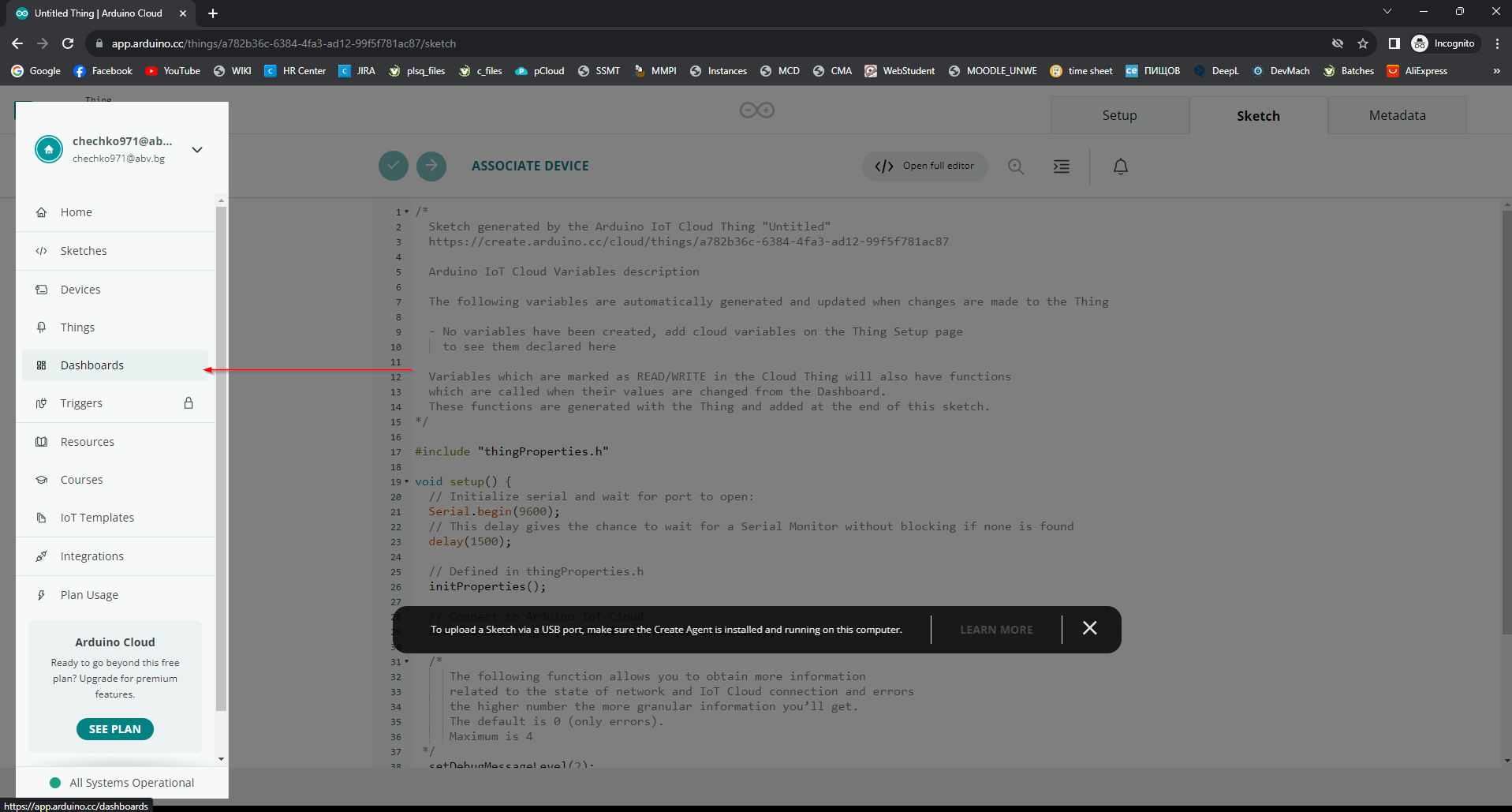
Description automatically generated

1. Всяка променлива си е създала собствена фунция, която трябва да кажем какво действие искаме да се извъшва при промяна на стойностите на съответната променлива.

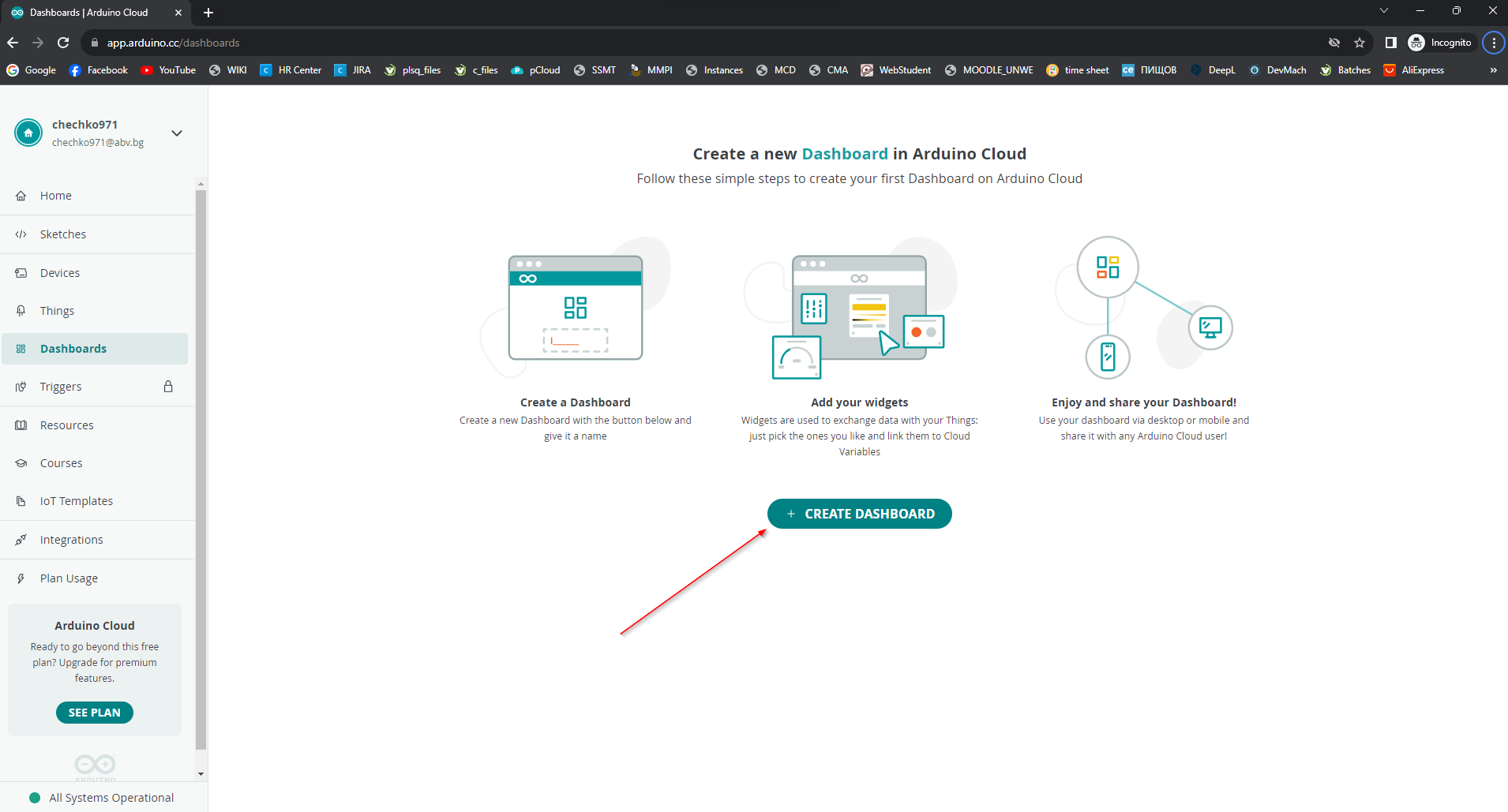
A screenshot of a computer

Description automatically generated

1. Минаваме към меню Dashboards.

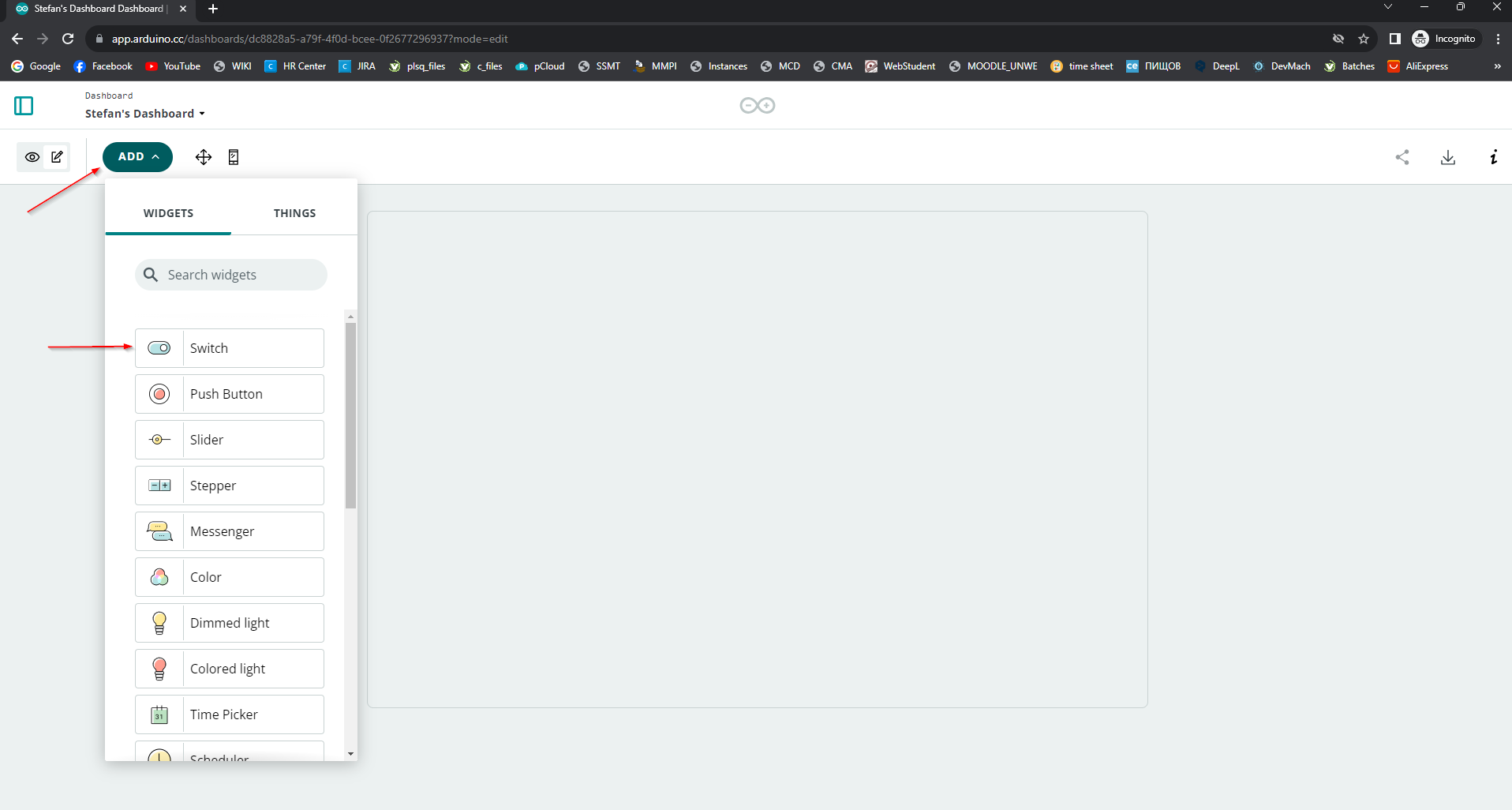


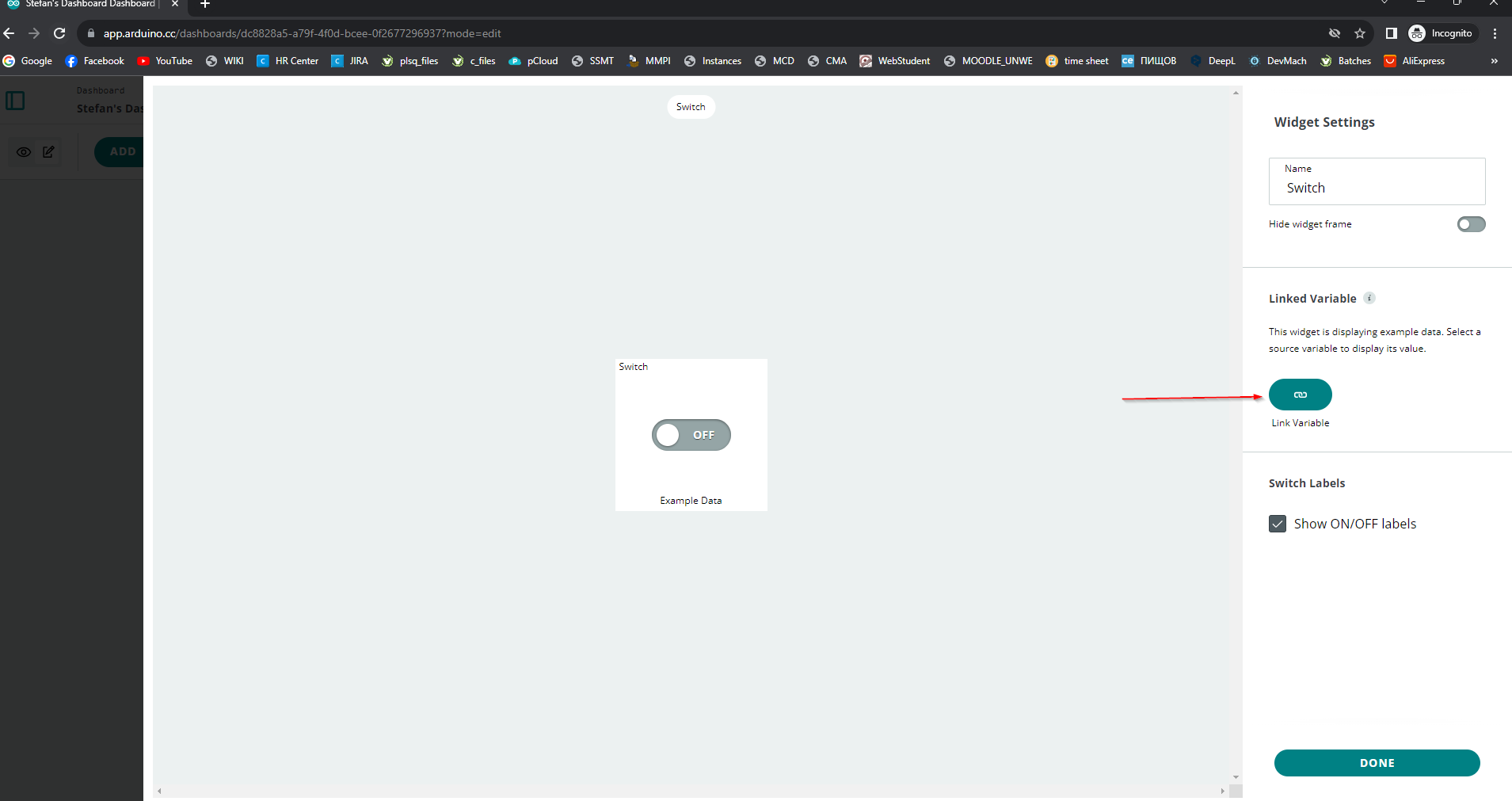
Създаваме нов Dashboard: Stefan’s Dashboard.



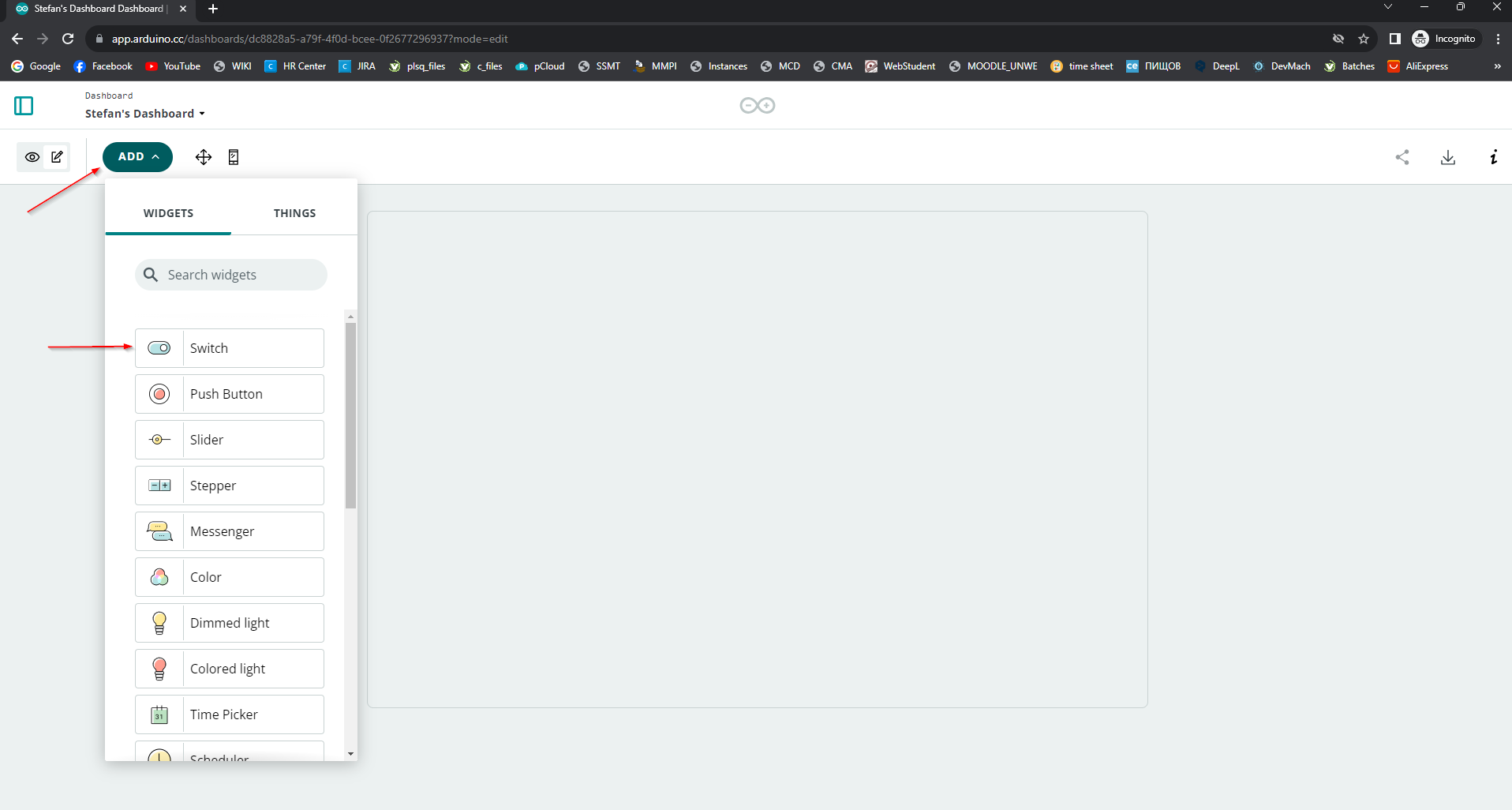
В него добавяме следните неща:

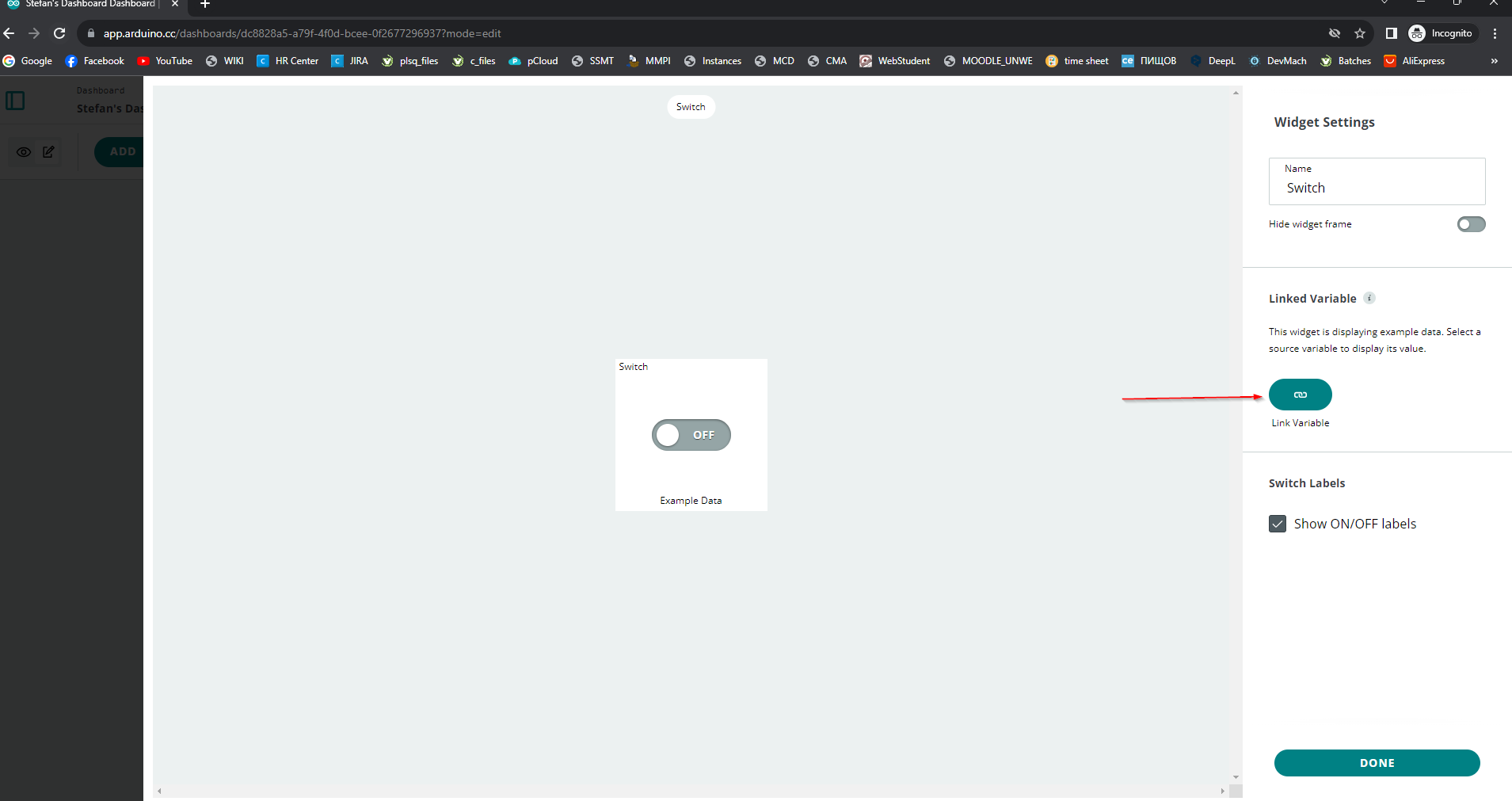
-> Switch бутон за контрол на лампичката, който ще вържем към променливата bulbControl.



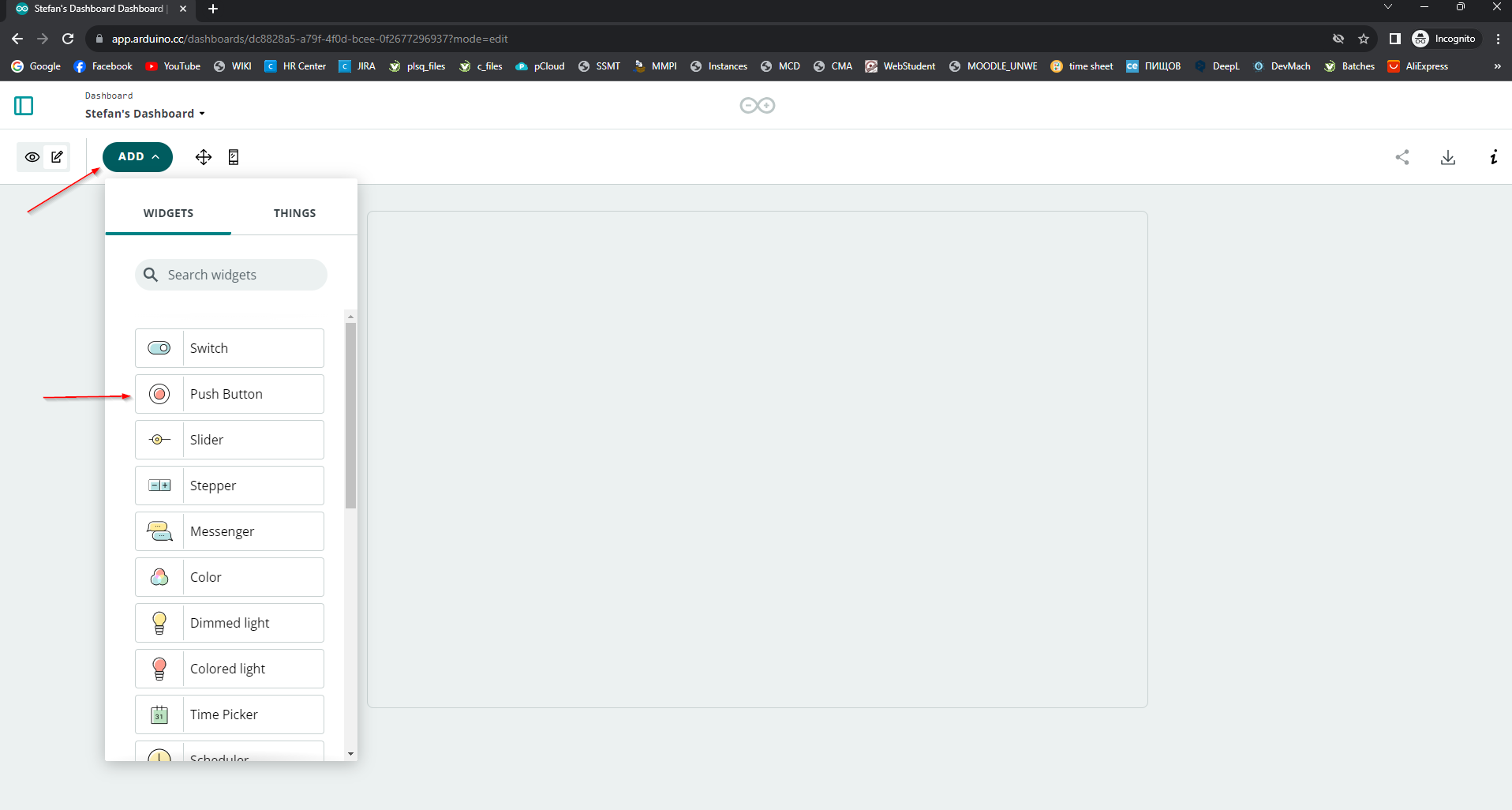


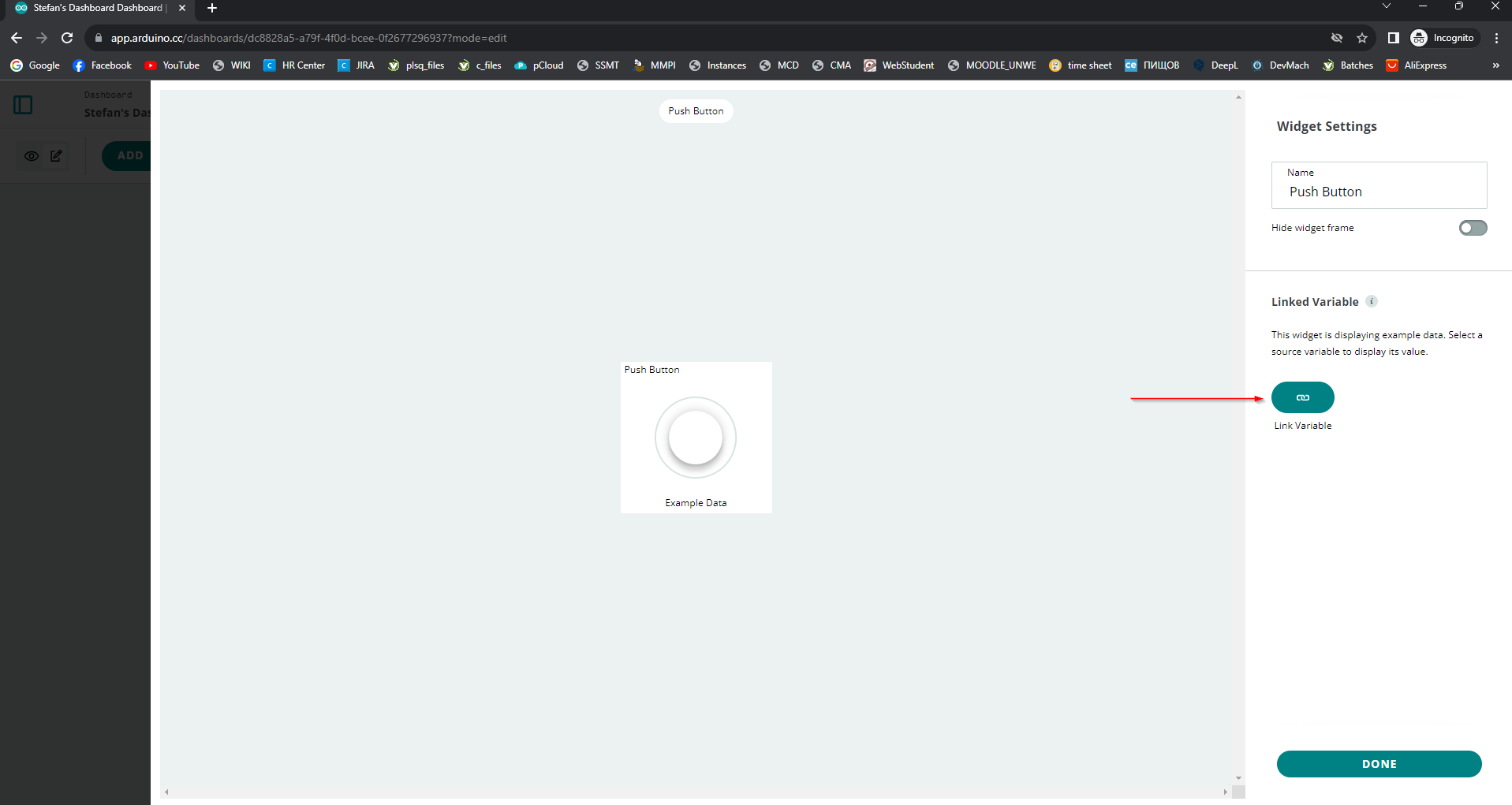
-> Switch бутон за контрол на вентилатора, който ще вържем към променливата fanControl.



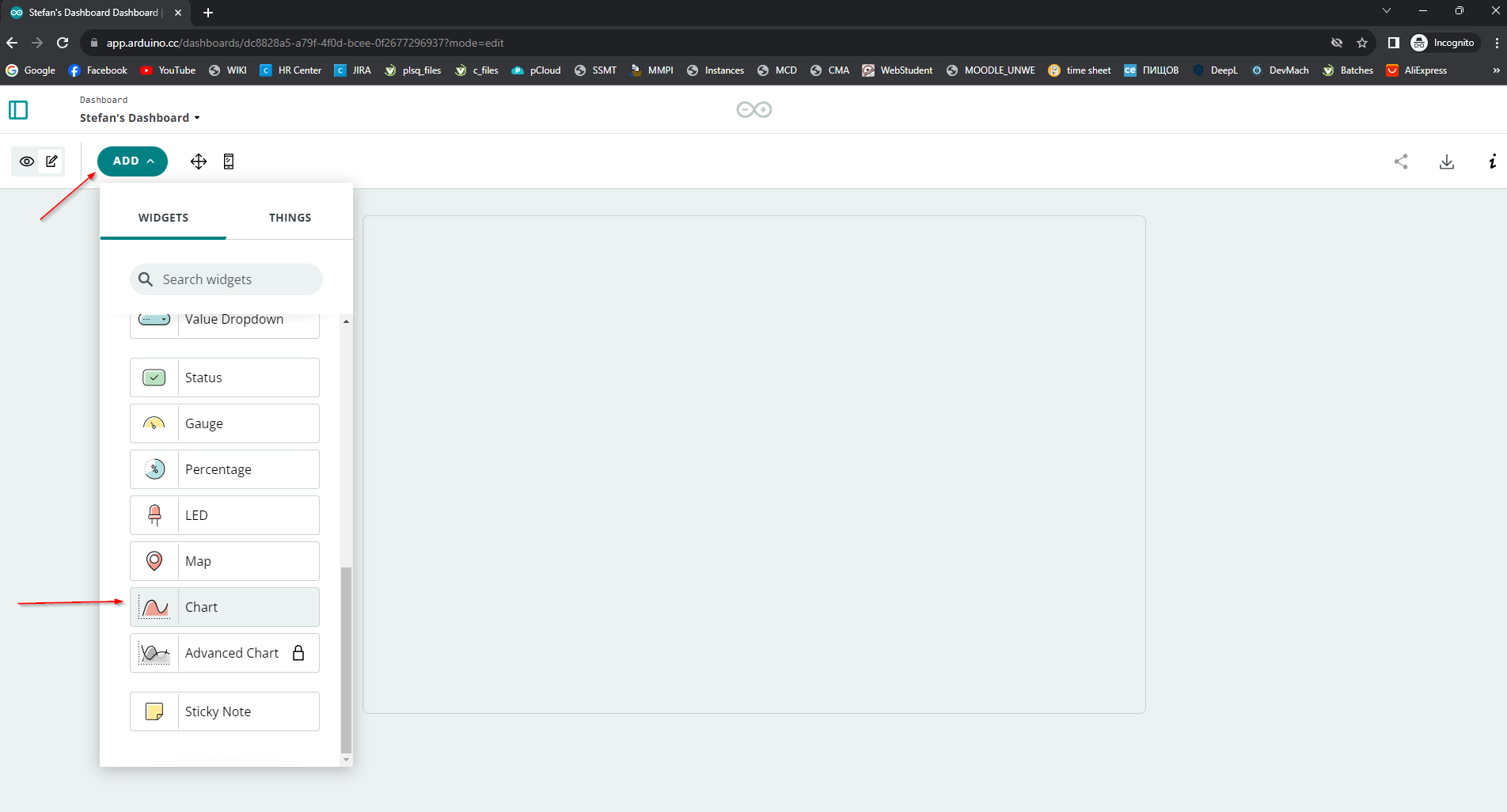


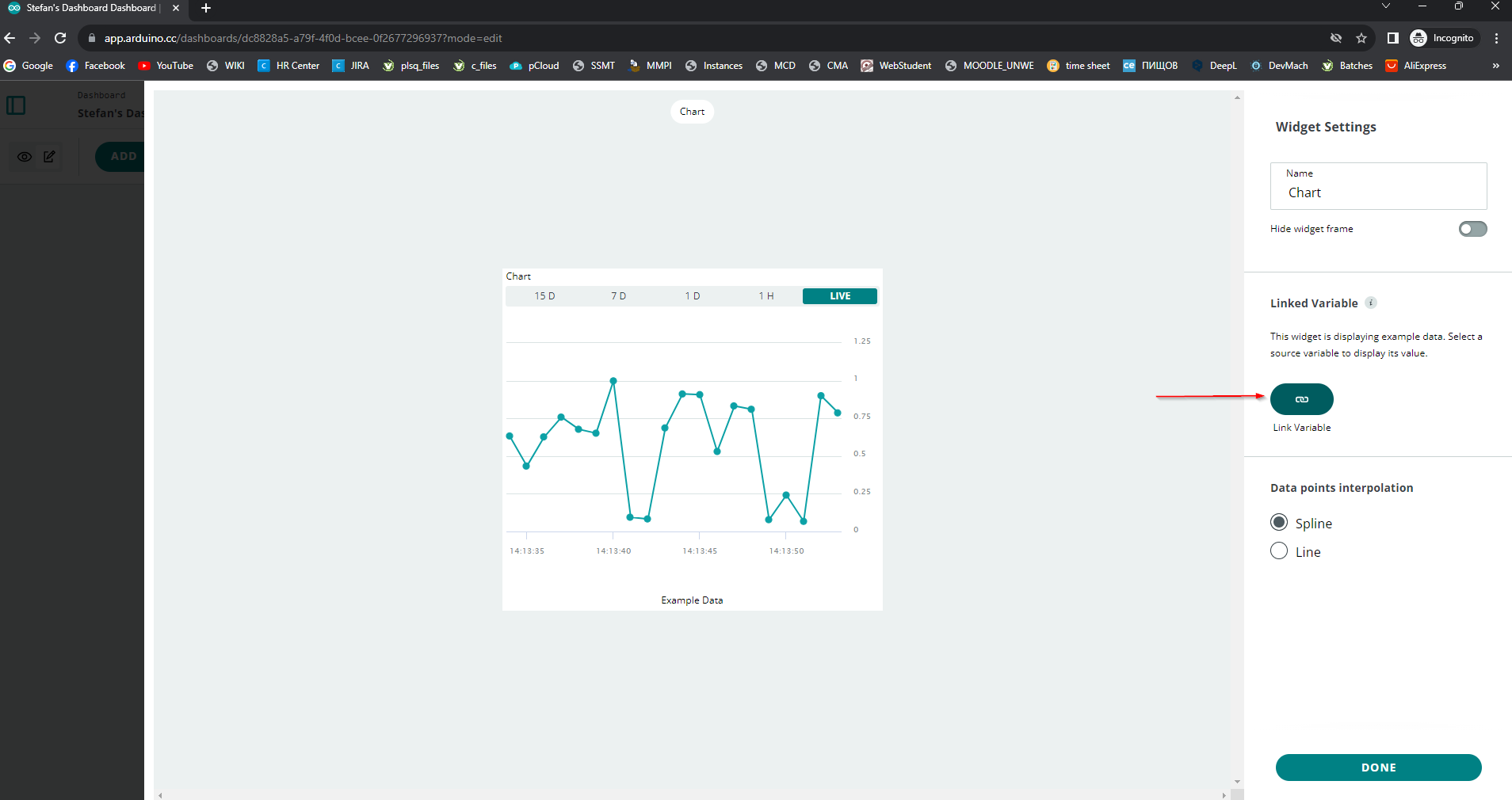
-> Push бутон за генериране на съобщение за данните за температура и влажност, който ще вържем към променливата showTemperature.



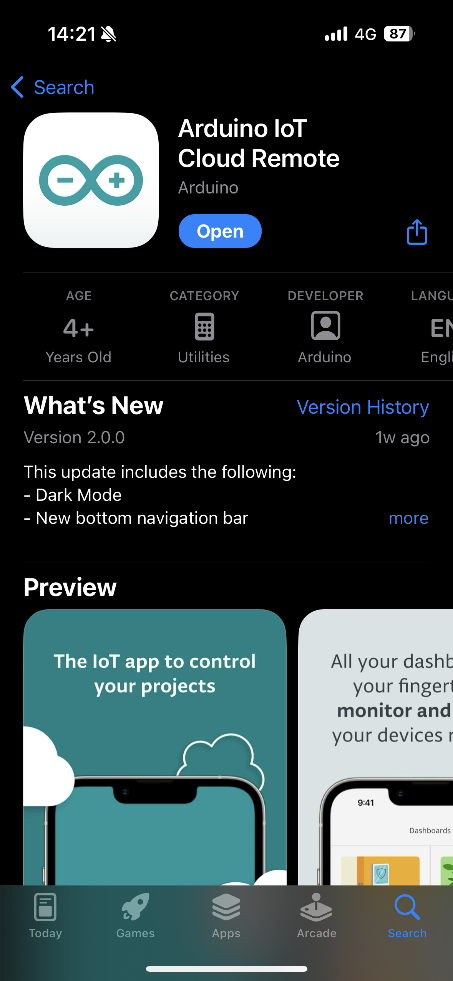


-> Диаграма(Chart), която ще показва температурата в Целзии по периоди. Връзваме я с променливата tempCelsius.





1. Можем да си изтеглим мобилното приложение Arduino Cloud на телефона от Google Play за смартфони с операционна система Android или от App Store за смартфони с операционна система IOS. През него също можем да контролираме устройствата след влизането в същия потребителски профил.



A screenshot of a device

Description automatically generated

1. Стартиране на програмата от бутон „Verify and upload“.

A screenshot of a computer

Description automatically generated

1. От меню Dashboards(през уебсайта) при натискане на бутон „Download historic data“ можем да изтеглим история на стойностите на всички променливи, които са свързани към отделните бутони и графики.

A screenshot of a computer

Description automatically generated

1. Данните се изпращат по имейл адрес, с който сме си направили регистрацията(студентската поща, дадена ни от УНСС).

A screenshot of a computer

Description automatically generated

1. При отваряне на линка от получения имейл се изтегля архив, който съдържа .cvs файлови с всички стойности по периоди на промяна за всяка променлива в отделен файл.

A screenshot of a computer

Description automatically generated

# Описание на кода на програмата:

/\*

Този Sketch е генериран от Arduino IoT Cloud Thing

Деклариране на променливите, които ще използваме. Генерират се автоматично:

float tempCelsius;

bool bulbControl;

bool fanControl;

bool showTemperature;

Променливите, които са маркирани като READ/WRITE в Cloud Thing, ще имат и функции, които се извикват, когато техните стойности се променят от информационното табло. Тези функции се генерират заедно с Thing и се добавят в края на тази скица.

\*/

#include "thingProperties.h" //библиотека, генерирана автоматично от Arduino Cloud

#include "DHT.h" // библиотека, необходима при използване на датчика за температура и влажност

#define DHTPIN 2 // DHT сензорите за влажност/температура са свързани към пин2

// Откоментирайте този тип датчик за температура и влажност, който използвате!

//#define DHTTYPE DHT11 // DHT 11

#define DHTTYPE DHT22 // DHT 22 (AM2302)

//#define DHTTYPE DHT21 // DHT 21 (AM2301)

DHT dht(DHTPIN, DHTTYPE); // Инициализиране на DHT датчика.

#define TEMPERATURE\_THRESHOLD\_1 28 // максимална температура за включване на вентилатора

#define TEMPERATURE\_THRESHOLD\_2 30 // максимална температура за включване на вентилатора + лампичката

#define fan\_pin 11 // Дефиниране на вентилатора на пин 11

#define led\_pin 13 // Дефинираме лампичката на пин 13

int calibrationTime = 45; // времето, което даваме на сензора за движение за калибриране.

long unsigned int lowIn; // времето, в което сензорът за движение излъчва нисък импулс

long unsigned int pause = 5000; // количеството милисекунди, през които сензорът за движение трябва да е изключен, преди да приемем, че движението е спряло.

boolean lockLow = true; // променлива, която гарантира, че ще изчакаме преминаването към LOW(изключено), преди да направим какъвто и да е изход.

boolean takeLowTime; // променлива, която ще показва дали сме калкулиране времето на работа на сензора за движение

#define motionSensorPin 3 // дефиниране на сензора за движение на пин 3

int counter1 = 1, counter2 = 0; // помощни променливи броячи

void setup() { // Началната фунцкия на програмата

// Инициализираме Серийният монитор

Serial.begin(9600);

// Това забавяне дава възможност за изчакване на сериен монитор, без да се блокира, ако такъв не бъде намерен.

delay(5000);

Serial.println("============================================================");

Serial.println("New test!"); // Съобщение за нов тест

pinMode(motionSensorPin, INPUT); // Конфигурира посочения пин да се държи като вход или изход.

digitalWrite(motionSensorPin, LOW); // Изключваме сенсозът за движение

// Даваме на сензора за движение време да се калибрира

Serial.print("Calibrating motion sensor ");

for(int i = 0; i < calibrationTime; i++){

Serial.print(".");

delay(1000);

}

Serial.println(" done");

Serial.println("MOTION SENSOR ACTIVE");

delay(50);

pinMode(fan\_pin, OUTPUT);

digitalWrite(fan\_pin, HIGH); // Включване на вентилатора

pinMode(led\_pin, OUTPUT);

dht.begin();

delay(2000);

float t = dht.readTemperature(); // Отчитане на температурата по Целзий (по подразбиране)

tempCelsius = t; // съхраняване на температурата в глобална променлива

float h = dht.readHumidity();

Serial.print("Humidity: ");

Serial.print(h);

Serial.print(" %\t");

Serial.print("Temperature: ");

Serial.print(t);

Serial.println(" \*C ");

// Defined in thingProperties.h

initProperties();

// Connect to Arduino IoT Cloud

ArduinoCloud.begin(ArduinoIoTPreferredConnection);

/\*

Следната функция ви позволява да получите повече информация

свързана със състоянието на връзката с мрежата и IoT Cloud и грешките.

колкото по-голямо е числото, толкова по-подробна информация ще получите.

По подразбиране е 0 (само грешки).

Максималната стойност е 4

\*/

setDebugMessageLevel(2);

ArduinoCloud.printDebugInfo();

}

void loop() { // Основната функция на програмата

ArduinoCloud.update();

// Изчакване 2 секунди между изчисленията

delay(2000);

// Отчитането на температурата или влажността отнема около 250 милисекунди!

// Показанията на сензора могат да бъдат с давност до 2 секунди (сензорът е много бавен).

float h = dht.readHumidity(); // Отчитане на влажноста в проценти

float t = dht.readTemperature(); // Отчитане на температурата по Целзий (по подразбиране)

tempCelsius = t;

float f = dht.readTemperature(true); // Прочитане на температурата по Фаренхайт (isFahrenheit = true)

float hif = dht.computeHeatIndex(f, h); // Изчисляване на топлинния индекс по Фаренхайт (по подразбиране)

float hic = dht.computeHeatIndex(t, h, false); // Compute heat index in Celsius (isFahreheit = false)

// Проверка дали има проблем с използването на датчика за температура и влажност

if (isnan(h) || isnan(t) || isnan(f)) {

Serial.println("Failed to read from DHT sensor!");

return;

}

if(t < TEMPERATURE\_THRESHOLD\_1 && fanControl == false)

digitalWrite(fan\_pin, HIGH);

if(t >= TEMPERATURE\_THRESHOLD\_1 && t < TEMPERATURE\_THRESHOLD\_2) {

Serial.print("Humidity: ");

Serial.print(h);

Serial.print(" %\t");

Serial.print("Temperature: ");

Serial.print(t);

Serial.print(" \*C ");

Serial.print(f);

Serial.print(" \*F\t");

Serial.print("Heat index: ");

Serial.print(hic);

Serial.print(" \*C ");

Serial.print(hif);

Serial.println(" \*F");

Serial.println("FAN turning ON!");

digitalWrite(fan\_pin, LOW);

Serial.println("LED turned OFF!");

digitalWrite(led\_pin, LOW);

delay(10000);

}

else if(t >= TEMPERATURE\_THRESHOLD\_2) {

Serial.print("Humidity: ");

Serial.print(h);

Serial.print(" %\t");

Serial.print("Temperature: ");

Serial.print(t);

Serial.print(" \*C ");

Serial.print(f);

Serial.print(" \*F\t");

Serial.print("Heat index: ");

Serial.print(hic);

Serial.print(" \*C ");

Serial.print(hif);

Serial.println(" \*F");

Serial.println("FAN turning ON!");

digitalWrite(fan\_pin, LOW);

Serial.println("LED turning ON!");

digitalWrite(led\_pin, HIGH);

delay(10000);

}

else{

if(digitalRead(motionSensorPin) == HIGH) {

counter2++;

if(counter2 == 1) {

Serial.println("LED turning ON!");

}

digitalWrite(led\_pin, HIGH); // Стартиране на лампичката

if(lockLow) {

// гарантира, че ще изчакаме прехода към LOW, преди да направим какъвто и да е изход:

lockLow = false;

Serial.print("motion detected at ");

Serial.print(millis()/1000);

Serial.println(" sec");

delay(50);

}

takeLowTime = true;

counter1 = 0;

}

if(digitalRead(motionSensorPin) == LOW && bulbControl == false) {

digitalWrite(led\_pin, LOW);

counter1++;

if(counter1 == 1) {

Serial.println("LED turned OFF!");

}

if(takeLowTime) {

lowIn = millis(); // Запазване на времето от включено към изключено състояние

takeLowTime = false; // Уверение, че това се прави само в изключен режим

}

//Ако сензорът за движение е изключен за повече от дадената пауза, предполагаме, че няма да има повече движение

if(!lockLow && millis() - lowIn > pause) {

// гарантира, че този блок от код ще се изпълни отново само след засичане на движение

lockLow = true;

Serial.print("motion ended at "); //output

Serial.print((millis() - pause)/1000);

Serial.println(" sec");

delay(50);

}

counter2 = 0;

}

}

}

/\*

Тъй като BulbControl е променлива READ\_WRITE, функцията onBulbControlChange() се извиква всеки път, когато се получи нова стойност от IoT Cloud.

\*/

void onBulbControlChange() {

if(bulbControl){

digitalWrite(led\_pin, HIGH);

Serial.println("LED turning ON!");

}

else{

digitalWrite(led\_pin, LOW);

Serial.println("LED turned OFF!");

}

}

/\*

Тъй като FanControl е READ\_WRITE променлива, функцията onFanControlChange() се извиква всеки път, когато се получи нова стойност от IoT Cloud.

\*/

void onFanControlChange() {

if(fanControl){

Serial.println("FAN turning ON!");

digitalWrite(fan\_pin, LOW);

}

else{

Serial.println("FAN turned OFF!");

digitalWrite(fan\_pin, HIGH);

}

}

void onShowTemperatureChange() { //Генериране на съобщение за температурата след натискане на бутона „Show temperature“

if(showTemperature){

delay(2000);

float t = dht.readTemperature(); // Read temperature as Celsius (the default)

tempCelsius = t; // store the temperature in a global variable

float f = dht.readTemperature(true); // Read temperature as Fahrenheit (isFahrenheit = true)

Serial.print("Temperature: ");

Serial.print(t);

Serial.print(" \*C ");

Serial.print(f);

Serial.println(" \*F");

}

}

/\*

Since TempCelsius is READ\_WRITE variable, onTempCelsiusChange() is

executed every time a new value is received from IoT Cloud.

\*/

void onTempCelsiusChange() {

// Add your code here to act upon TempCelsius change

}

# Заключение.

Този проект беше едно голямо предизвикатество за мен. Първо започнах с гледане на клипове в Youtube за взимане на идеи за IOT. От там разбрах за микроконтролера Arduino UNO R3. Поръчах си него, заедно със сензора за движение, датчик за температура и влажност, лампички и кабели за свързване. Изтеглих си средата за разбработка Arduino IDE, преди да са ми пристигнали частите. На първата лекция проф. Боянов ми показа за първи път микроконтролера Arduino UNO R3 и там написах първата си IOT програма за мигане на лампичка на всеки 2 секунди. След като ми дойдоха частите започнах да мисля идеи как да ги свържа. Осъзнах, че ми трябва и вентилатор, за да има какво да контролирам с датчика за температура. Почнах да сглобявам уред по уред и функция по функция. След това си поръчах микроконтролер Arduino UNO R4 с WiFi модул. След като го получих, първото впечатление беше надписът Arduino Cloud на кутийката на микроконтролера. Зачудих се какво пък може да е това и като прочетох в интернет разбрах, че има такава онлайн безплатна платформа за IOT. Пак започна една голяма борба да интегрирам програмния код от средата Arduino IDE в онлайн платформата Arduino Cloud. Но променлива по променлива, функция по функция, всичко най-накрая проработи. Сериозен проблем се оказа и пускането на Hotspot от телефона, интернетът се пуска с 5 GH свързаност. И лаптопът ми се свързваше, но микроконтролерът не ставаше. Казвах си, какво става пак, няма как едното утройство да се свързва. След много прочетени теми по форуми разбрах, че в настройките на телефона ми има бутонче за смяна на интернет свързаността от 5 GH на 2.4 GH. Най-накрая всичко проработи. След всичкия този труд и няколко безсънни нощи сбъднах една моя мечта – да свържа софтуер с хардуер. Благодаря на проф. Любен Боянов и ас. Гено Стефанов, че ме амбицираха да направя наистина нещо интересно за мен, да се докосна до идеята за IOT.

A computer with wires on it

Description automatically generated