**Assembly of circuit with potentiometer for car and sending IoT data. Difficulty level:** Medium

**Goals**

Automotive IoT is the integration of gadgets, sensors, cloud computing, applications, and other such components into vehicles to function as a complex system for the connection of cars, predictive maintenance, fleet management, OEMs, insurance, and more. The integration of the Internet of Things in the automotive industry allows manufacturers to implement sought-after innovations that can ultimately transform cars into near-artificial intelligence. At a didactic level, we are now going to develop some exercises using sensors for data acquisition, processed by the Arduino microcontroller.

This exercise intends to show how a potentiometer with the function of controlling the voltage works, which allows it to be applied in the most diverse functionalities of the automotive sector, such as controlling the acceleration of a car engine. In this example students can see how the car engine accelerates for a short period of time.

For the possible sending of data, it will be necessary to apply, for example, the ESP8266 ESP-01 module that allows the connection of several devices to the internet (or local network), and consequent sending of data from the sensors applied to the autonomous system.

**Image-1:** Understanding the application of Potentiometer in a car and communicating with IoT.

Uma imagem com texto, captura de ecrã

Descrição gerada automaticamente

**Image 1:** application of Potentiometer in a car and communicating with IoT

**Skills**

* The skills our students will gain are:
* Students' ability to build circuits will be developed.
* The ability to program the Arduino board and use the ESP8266 Module for Internet access will develop.
* The ability to receive data from the potentiometer sensor and send the received data to Thing Speak will be gained.
* Data analytics will improve their ability to connect with the Internet of Things.

**Required materials and circuit diagram.**

In this exercise we intend to learn how to draw diagrams (circuits), connect all the components correctly, develop software based on C language (Arduino), connect to the wifi network, communicate with an IoT server, ThingSpeak and read server-generated graphics.

|  |  |
| --- | --- |
| **Quantity** | **Component** |
| 1 | Arduino Uno R3 |
| 1 | ESP01-8266 |
| 1 | Power Supply (breadBoard) |
| 1 | BreadBoard |
| 1 | Potentiometer. |
| 7 | 330 Ω Resistor |
| 3 | Blue LED |
| 1 | Red LED |
| 2 | Green LED |
| 1 | Yellow LED |

**Table 1 - Components List**

**Materials table**

|  |  |
| --- | --- |
| Arduino | ESP01 - 8266 |
| Bread Board + Power Supply | Potentiometer |
| 330Ω | Red LED, blue LED, Green LED, Yellow LED |
| Jumper wire | |

Uma imagem com diagrama

Descrição gerada automaticamente

**Image 2 – Diagram circuit**

**Implementation**

Development of communication of microcontroller systems, and sensors, with the ThingSpeak IoT cloud.

The ESP8266 WiFi module (image 3) is a small shield with integrated TCP/IP protocol that can give any microcontroller access to the WiFi network. The ESP8266 is capable of both hosting an application and offloading all WiFi network functions from another application processor. Each ESP8266 module is pre-programmed with an AT command making its firmware settings, meaning that we can simply connect this module to the Arduino working as any other WiFi shield would. This module has a great cost/benefit ratio and has a very large and constantly growing user community.

Uma imagem com texto, eletrónica, circuito

Descrição gerada automaticamente

**Image 3 - ESP01 – 8266**

The module Potentiometer (image 4) is a analogy Rotation Sensor is Arduino compatible. It is based on a potentiometer. Its voltage can be subdivided into 1024, easy to be connected to Arduino with our sensor shield. Combined with other sensors, we can make interesting projects by reading the analogy value from the IO port.

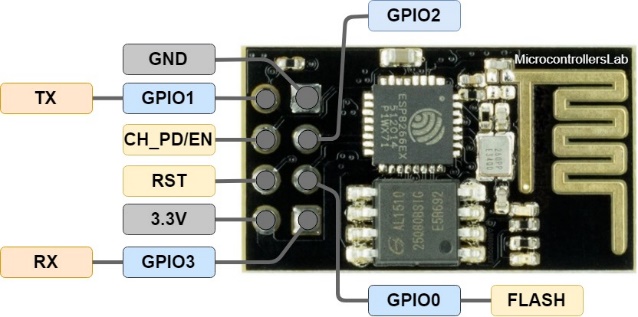
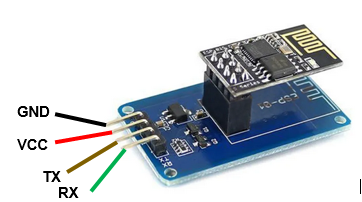


**Image 4 Potentiometer**

**Implementation in practice**

1. Assemble the circuit in the image 2;
2. Connect correctly ESP01-8266 image 5

**Image 5** ESP-01 Connections



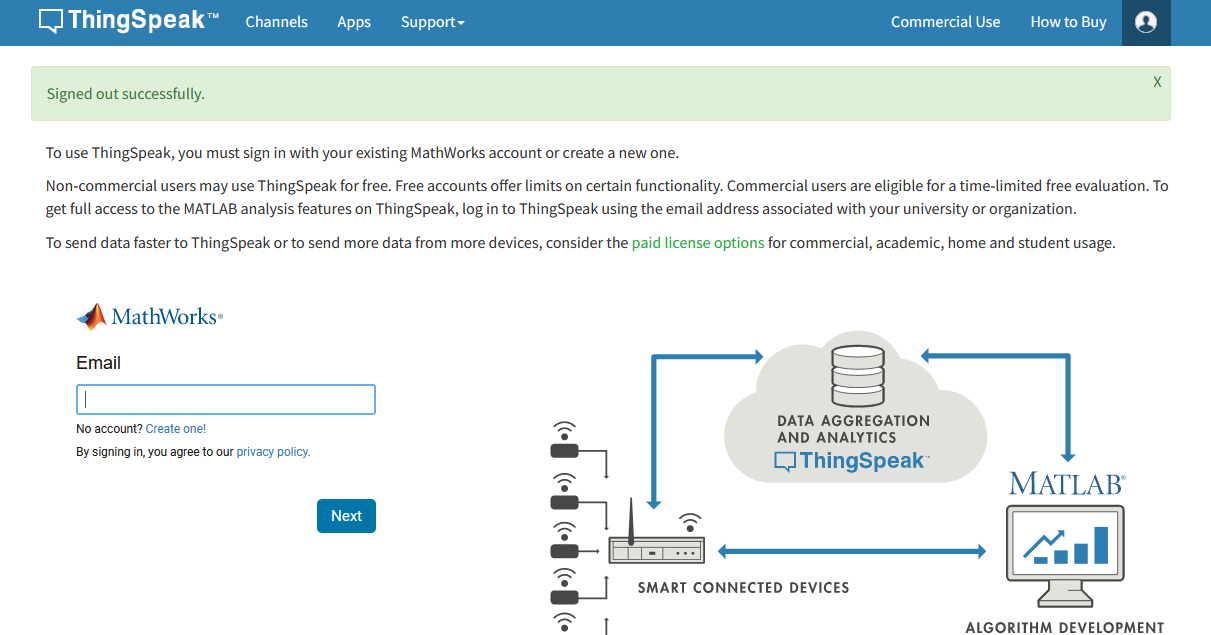
1. Real assembled circuit image 6

Uma imagem com texto, eletrónica

Descrição gerada automaticamente

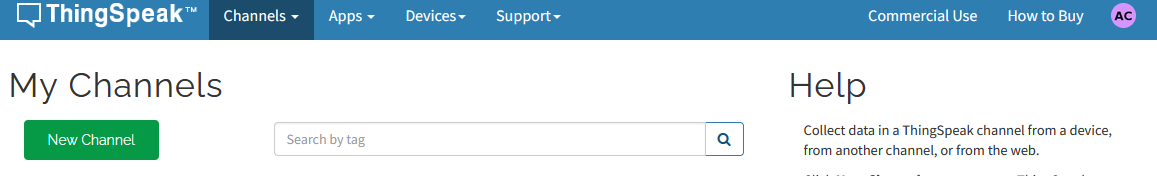
**Image 6** Real circuit in breadboard

1. Create a ThingSpeak account image 7



**Image 7 - Thing Speak**

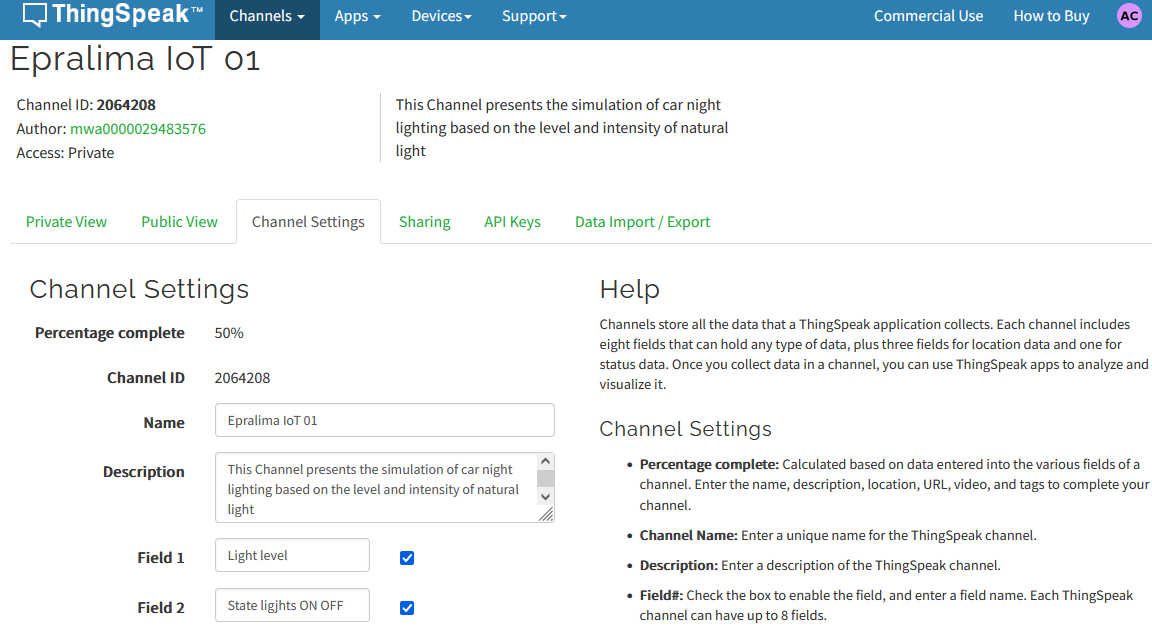
1. Create a new channel image 8



**Image 8 Interface ThingSpeak**

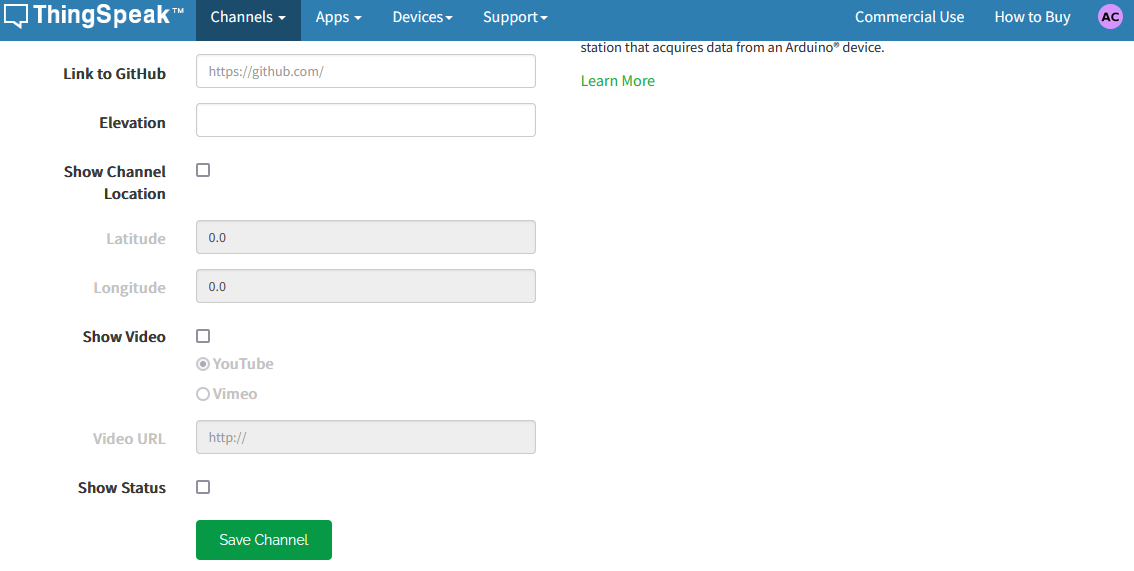
1. Configure channel, with name, description, and fields. Image 9.

**Note:** The fields refer to data processed by the microcontroller and data from the sensors under study. Each field will generate a graph.



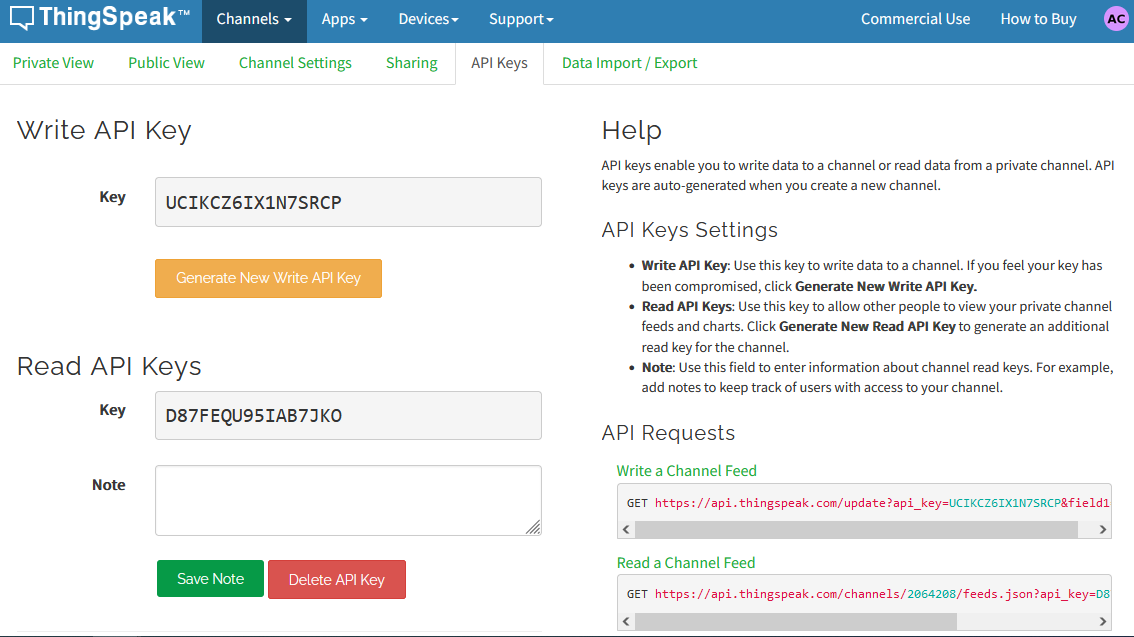
**Image 9 Configure Channel**

1. Save settings channel Image 10



**Image 10 Save settings channel.**

1. In this step, we will pay special attention to the api keys, as they are the ones that, through the string key, will allow access to the IoT repository in Arduino programming. Also very important are the API requests.



**Image 11 - API Keys**

1. Programming Arduino

Inclusion of the necessary libraries and declaration of variables and constants inherent to the program's operation.

Uma imagem com texto

Descrição gerada automaticamente

Void setup() function for initializing parameters for starting the program.

Uma imagem com texto

Descrição gerada automaticamente

**AT commands**

AT commands are the basic way to configure and trigger the ESP8266 when it is under control of an external device (like an Arduino, for example).

Current AT commands are direct descendants of the so-called "Hayes Standard" from 1981, used to allow personal computers to interact with telephone connections by directly controlling a mode.

The **InitWifiModule()** function initializes the ESP8266 through AT commands.

Uma imagem com texto

Descrição gerada automaticamente

The **envioDadosESP\_AT(str,int,boolean)** function is responsible for sending AT commands to the ESP8266

Uma imagem com texto

Descrição gerada automaticamente

The **startThingSpeakCmd(str,int,boolean)** function opens connection to ThingSpeak IoT analytics platform. The IP address of the ThingSpeak platform is: 184.106.153.149 with connection on port 80. The AT command to start ThingSpeak communication is AT+CIPSTART=PROTOCOL, IP\_ADRESS, PORT.

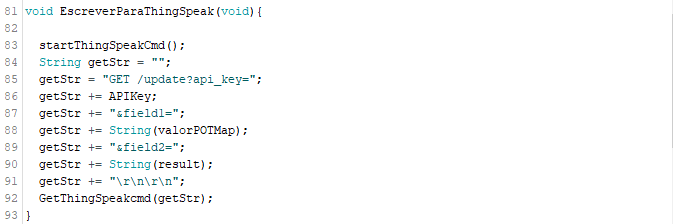
Uma imagem com texto

Descrição gerada automaticamente

The **EscreverParaThingSpeak** function generates a string to build an API Request.

**Example:**

**GET /update?api\_key=U………….P&field1= 0&field2= 0**



The **GetThingSpeak(str)** function, is responsible for determining and sending an API Request through the AT+CIPSEND command to write to the ThingSpeak channel, returning the message received by the response from the ThingSpeak data platform. The communication will be closed if the response is not favourable.

Uma imagem com texto

Descrição gerada automaticamente

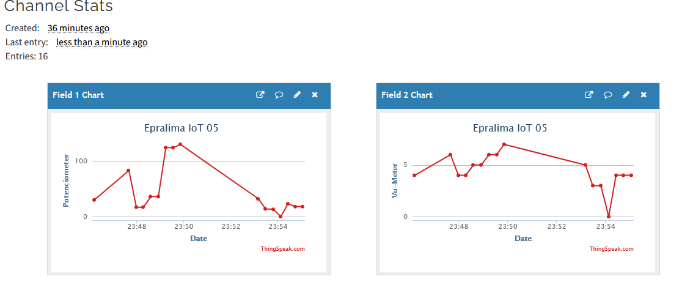
The **potenciometerRead()** function receives data from the Potentiometer sensor on an analogy port (A0) on the Arduino. In this function the values sent by the potentiometer are between 0 and 1000, however they were mapped to 0 to 128 to reduce the scale. Based on the sensor values and through a logarithmic calculation, the ports that will activate the vu-meter leds were mapped.

Uma imagem com texto

Descrição gerada automaticamente

**Results**

Given that a potentiometer has the function of controlling voltage, this allows it to be applied in the most diverse functionalities in the automotive sector, such as controlling the acceleration of a car engine.



**Image 12 – Results IoT ThingSpeak**

The data acquired by the ThingSpeak IoT platform can also be exported to CSV files and consequently imported into datasheets as shown in Table 2

Uma imagem com mesa

Descrição gerada automaticamente

**Tabela 2 - DataSheet**

**In short**

if this were applied, we could conclude that the car's engine had an acceleration for a short period of time for example.