**Assembly of circuit with** **RFID technology for car and sending IoT data**

**Difficulty level:** Difficult

**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 simulates RFID access control. In motorsport, it becomes safer to code contactless accesses. In this exercise it is possible for students to check and control the number of accesses, as well as to know if a car engine is in use at a given moment.

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 RFID technology in a car and communicating with IoT.

Uma imagem com texto, diagrama

Descrição gerada automaticamente

**Image 1:** application of RFID technology 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 brightness 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 (braedBoard) |
| 1 | BreadBoard |
| 2 | Push button |
| 2 | Led Green and Red |
| 1 | LCD display 2 x 16 (I2C) |
| 2 | Resistor 1KOhm |
| 2 | Resistor 330Ohm |
| 1 | DC motor |
| 1 | Module bridge L298N |

**Table 1 - Components List**

**Materials table**

|  |  |
| --- | --- |
| Arduino | ESP01 - 8266 |
| Bread Board + Power Supply | L298N Bridge Motor Module |
| DC Motor | LCD display 2 x 16 (I2C) |
| Jumper wire | 330Ω 1KΩ Leds |

Uma imagem com eletrónica, Engenharia eletrónica, circuito

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

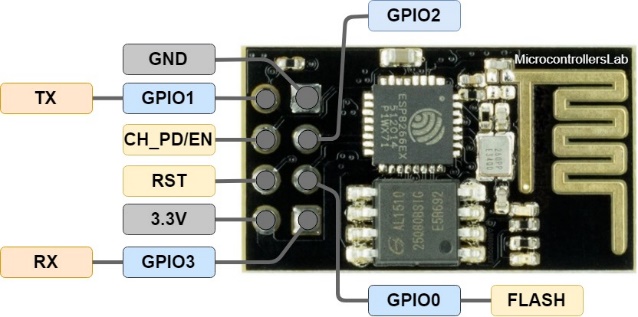
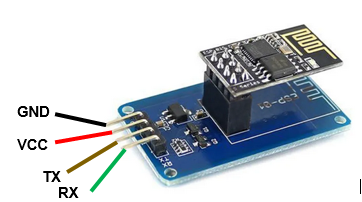
**Image 3 - ESP01 – 8266**

LM35 Linear Temperature Sensor is based on semiconductor LM35 temperature sensor. It can be used to detect ambient air temperature.

**Implementation in practice**

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

**Image 4** ESP-01 Connections



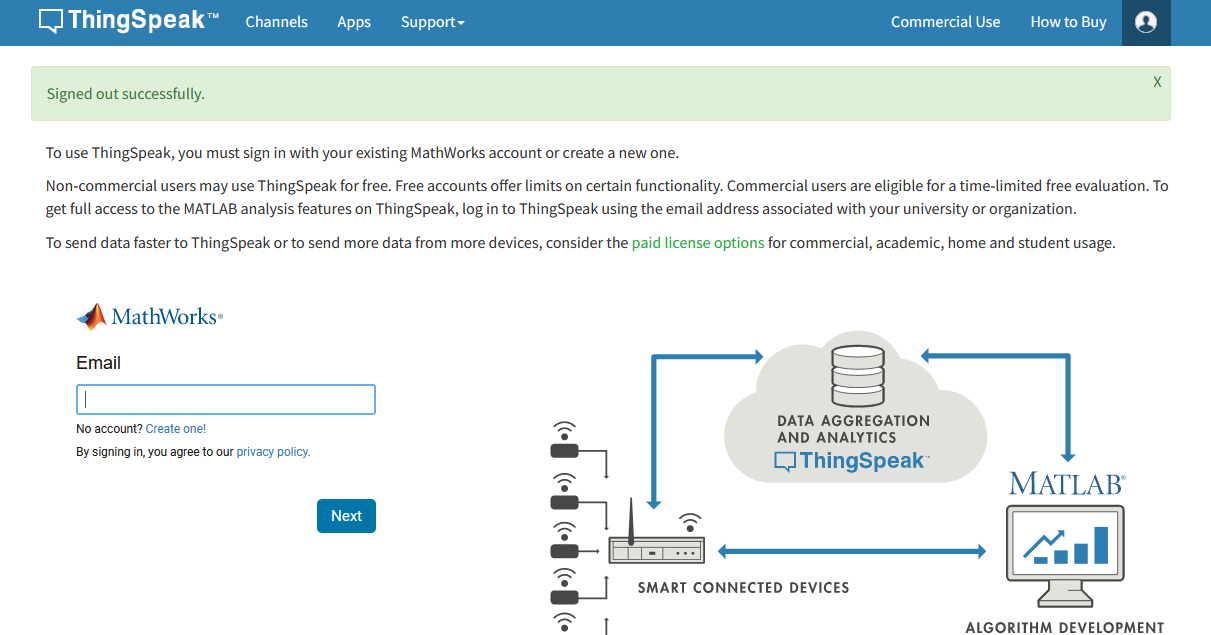
1. Real assembled circuit image 6

Uma imagem com eletrónica, Fios elétricos, Engenharia eletrónica, cabo

Descrição gerada automaticamente

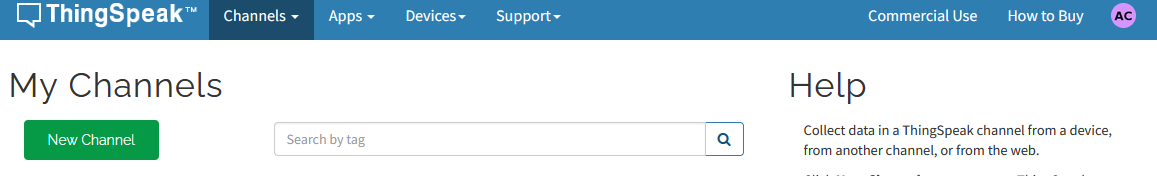
**Image 5** Real circuit in breadboard

1. Create a ThingSpeak account image 7



**Image 6 - Thing Speak**

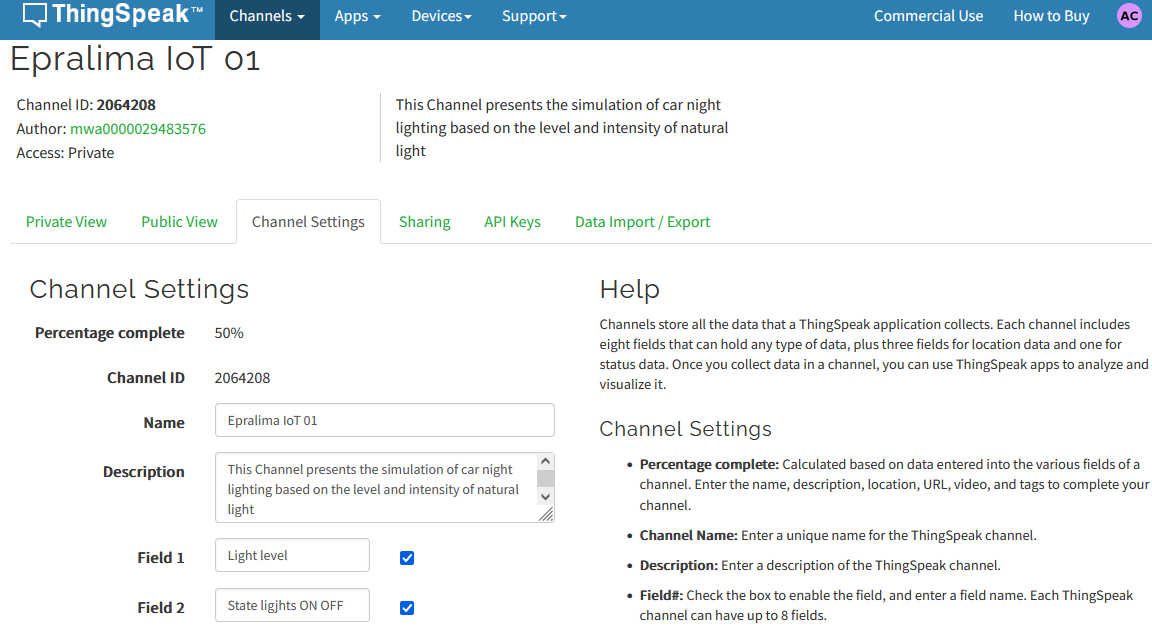
1. Create a new channel image 8



**Image 7 Interface ThingSpeak**

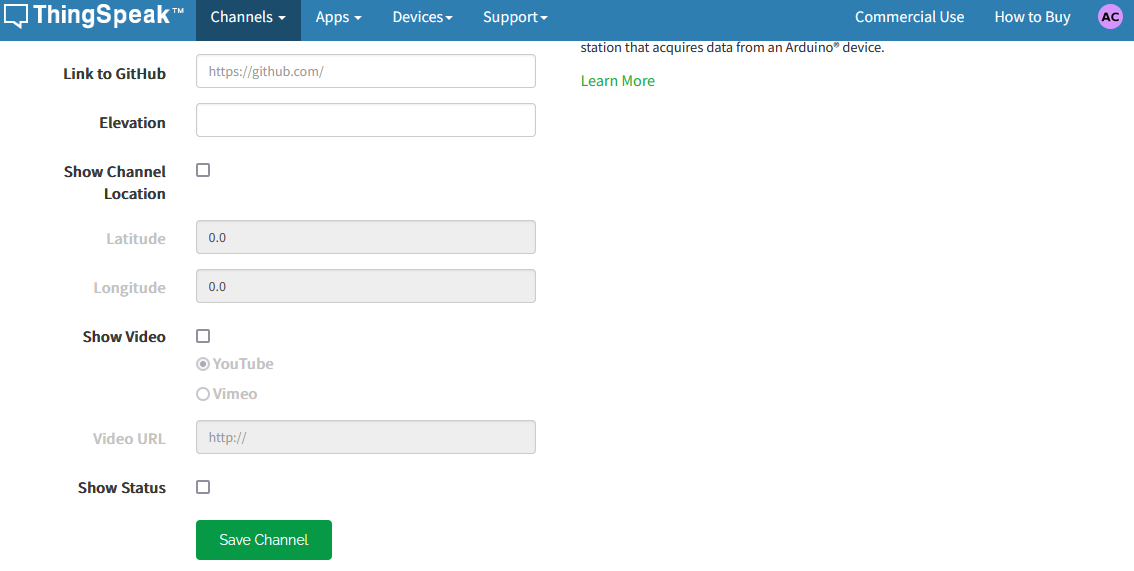
1. Configure channel, with name, description, and fields. Image 9.

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



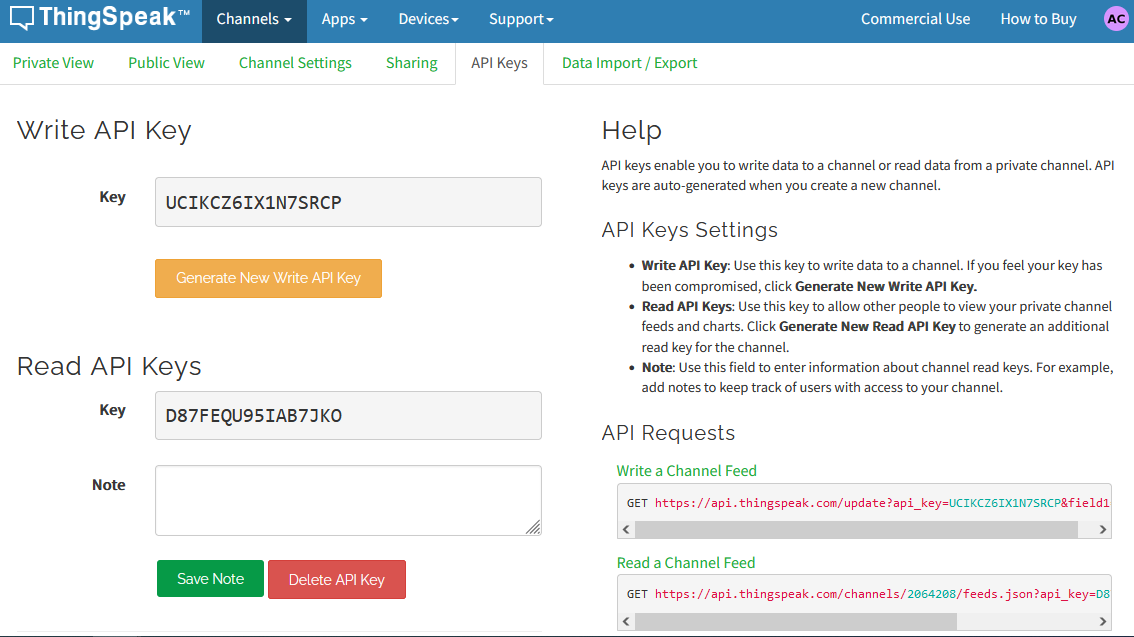
**Image 8 Configure Channel**

1. Save settings channel Image 10



**Image 9 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 10 - 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, captura de ecrã, Tipo de letra, número

Descrição gerada automaticamente

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

Uma imagem com texto, captura de ecrã, Tipo de letra, número

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.

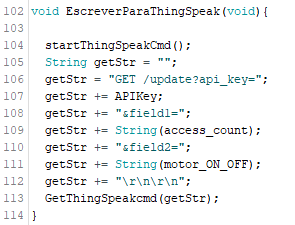
Uma imagem com texto

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 **AccessAllowed() and AccessDenied()** These procedures provide instructions to the various components (Display, Leds and DC Motor) depending on whether the access is valid or not.

Uma imagem com texto, captura de ecrã, Tipo de letra, número

Descrição gerada automaticamente

Various procedures for controlling the motor and writing to the LCD.

Uma imagem com texto, captura de ecrã, Tipo de letra, número

Descrição gerada automaticamente

**Results**

It is possible to control the number of accesses as well as to understand if an engine is in use at any given time.

Uma imagem com texto, file, diagrama, Gráfico

Descrição gerada automaticamente

**Image 11 – 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 texto, captura de ecrã, Tipo de letra, número

Descrição gerada automaticamente

**Tabela 2 - DataSheet**

**In short**

This exercise simulates RFID access control. In motorsport, it becomes safer to encode accesses through contact less.