A Prototype of Classroom Energetically Efficient

Diego A. Godoy(✉), Santiago H. Bareiro, Fabián E. Favret, Juan P. Blariza, and Guillermo Colotti

Centro de Investigación en Tecnologías de la Información y Comunicaciones (CITIC), Universidad Gastón Dachary, Av. López y Planes 6519, 3300 Posadas, Argentina
{diegodoy,hbareiro,efabianfavret}@citic.ugd.edu.ar, juanblariza@gmail.com, gui.colotti@gmail.com

Abstract. Various organizations have reported that buildings (among them, educational institutions) are responsible for the consumption of 40% or more of all the primary energy produced worldwide. The control of temperature and lighting in said institutions is carried out manually. That means that every time a classroom is used, people must turn on lights and air conditioners, and then take care of turning them off whenever they are not required. Faced with this scenario, the alternative proposed in this work allows efficient automatic control of lighting and temperature preferences for each professor and each class.

That is why a Prototype of Classroom Energetically Efficient was built in this paper. It has three modules: The one is the web application that was developed using the Laravel framework for the backend and Vuejs for the frontend. Its main function is to send commands to devices. The second is the IoT framework, which fulfills the function of communicating the web application with the hardware, providing the necessary endpoints, and making the registered data available. And finally, the hardware that was built using NodeMCU ESP8266 boards. Its function is to be an actuator i.e. receive the data from the IoT framework and executes commands. We also build a classrom mockup to show the prototype in action.

Also, the performance tests of different scenarios were carried out, being satisfactory, and allowing the development of the planned functionalities.

Keywords: Internet of Things · Smart classrooms · Energy efficiency

1 Introduction

A modern educational institution has a large number of classrooms, each with many lighting and cooling devices (air conditioners). "Various organizations, committed to the efficient use of energy and the conservation of our environment, have reported that buildings are responsible for the consumption of 40% or more of all the primary energy produced worldwide…" [1]. For this reason, it is reasonable to say that the energy consumed in a school day during peak hours is very high.

S. Nesmachnow and L. Hernández Callejo (Eds.): ICSC-CITIES 2020, CCIS 1359, pp. 17–29, 2021.
https://doi.org/10.1007/978-3-030-69136-3_2

That is normal (as well as in any, or most, educational institutions) considering the daily use and movement of students and professors. As well, it is also normal that the temperature control (by turning on/off and regulation in air conditioners) and lighting (by turning on/off lights) in the classrooms, is carried out manually. This means that every time a classroom must be used, one person must take care of turning on the lights and air conditioners (if necessary), and turning them off when they are not required. And if we take into account that the people who perform these tasks are the same janitors and secretaries, responsible for many other tasks. Several devices may be turned on unnecessarily for several hours.

Likewise, it is observed that each professor usually has a lighting requirement in the classroom. For example, while there are professors who require the maximum possible lighting (all classroom lights on), others do not use the headlights (near the blackboard), since they use projectors with presentations and/or slides. In this way, the visualization of slides showed by projectos are more clear. So also other professors prefer all lights completely off for the same reason. The temperature will depend on the weather conditions. For example, days with extreme temperatures (hot or cold), will require more use of air conditioners.

Therefore, it is extremely important to find the method to manage efficiently the energy consumption, not only to reduce the institution's expenses but also to help preserve finite resources and thus mitigate the environmental impact due to its unnecessary use.

Consistently and because the technology advances it is necessary to design a solution to that problem through the use of an IoT Framework evaluated in [2], a Web application and specific hardware [3], which allows each professor to independently configure their desired lighting and temperature profile for the classroom to be used, and that this is applied automatically in the right time.

2 Related Works

Educational institutions are one of the main responsible for the amount of energy consumed, for the number of activities carried out in classrooms, offices, libraries, and also for the waste of energy due to the inefficient use of electricity [4], but also, by the mobilization of people using vehicles [5].

In [1] a line of action is established regarding customs and policies for the good use of energy that not only promotes the development, implementation, and adaptation of software and hardware. Instead, they serve as tools to save money at the National University of Misiones. In [6] the research process for the development of an IoT system is presented, which has been designed to promote an intelligent lighting service in an academic environment. The IoT system orchestrates a series of sensors, monitoring systems, and controlled actions, based on the principle of making available the functions of the system and the record of consumption in real-time through web services. Likewise, in [7] the design and implementation of an intelligent automated system based on Ethernet for the conservation of electrical energy using a second-generation INTEL GALILEO development board are proposed. The proposed system works on automation so that electrical devices and switches can be remotely controlled and monitored without any human intervention. The project developed in [8] uses IoT-based technology to achieve automation in classrooms and proposes an approach to control and manage electrical equipment such as fans and lights based on the presence of people.

3 Methodology

This section presents the hardware and technology used in this research project

3.1 Used Hardware

The hardware used is the NodeMCU ESP8266 (Fig. 1). NodeMCU is an open source IoT platform. Includes firmware that runs on Espressif Systems ESP8266 WiFi SoC (System on Chip) and hardware that is based on the ESP-12 module [9].

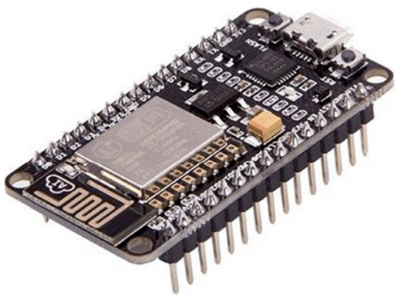


Fig. 1. Node MCU ESP8266.

The ESP8266 is a low-cost WiFi chip with a full TCP/IP stack and a microcontroller. The firmware can be programmed using the Lua scripting language, although currently the Arduino IDE also supports programming in C language [10].

3.2 Used Web Applications Technologies

In the web application development, different frameworks were used such as Laravel, VueJs, and Postman.

Laravel is an open source PHP framework for developing web applications and services through layered architecture, providing multiple functionalities required for any web application.

VueJs is a progressive JavaScript framework for creating user interfaces. It is an alternative to frameworks like Angular or React [11].

Postman is a tool that allows you to make HTTP requests to any REST API, whether third-party or your own, to test the operation of the API through a graphical interface.

3.3 Framework IoT Ubidots

Ubidots [12] is a platform for building, developing, testing, learning, and exploring the future of applications and solutions connected to the Internet [12].

Regardless of whether one or one thousand devices are connected, the same effort is required with all types of Ubidots devices. The creation of the new device in Ubidots can be replicated by automatically setting the variables, the device properties, and the appearance each time a new piece of hardware is detected. Some of its characteristics can be seen in [13]. The Ubidots service stack can be seen in Fig. 2.

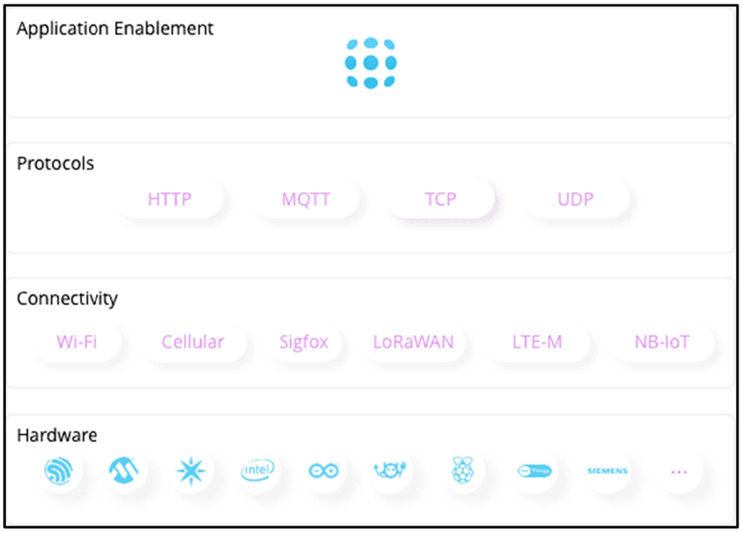


Fig. 2. Ubidots service stack

4 Proposed Solution Architecture

The technological solution to the problem consists of using an IoT framework that fulfills the function of carrying out communication between the parties(Web Application and Hardware), providing a method to store the information and make it available to read, to be consumed at the required time. The Web Application defines all the behavior to be followed and the hardware is only an executor of actions, ordered by the Web Application. Examples of this would be the web application reads the information from the temperature sensor sent by the hardware, and tells when to switch on or off an air conditioner. The proposed solution is shown in Fig. 3.

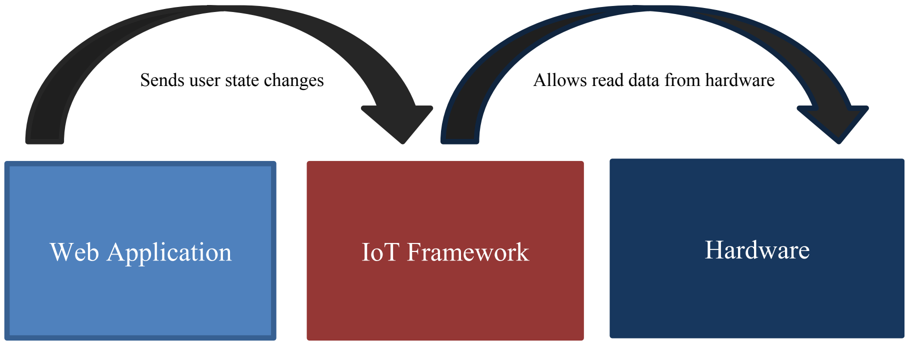


Fig. 3. Proposed solution architecture

Each arrow in the previous graphic represents an HTTP request performed by the web application and the hardware respectively. Based on this, we have four situations:

1. The web application sends state change data to the IoT framework. These changes of states refer to changes in the profiles according to the preferences of each professor. For example: If the air conditioning in classroom 1, at a certain time, should be turned on. The IoT framework stores this data and makes it available to be consumed. It should be noted that the IoT framework, in this case, does not handle the logic of when an air conditioning should be turned on or not, it simply receives the data. For example, "Air1: 1" data, already generated by the web application, and makes it available to be consumed by who requires it.
2. The available data is consumed by the hardware. The latter reads the data, through an HTTP Request, and just executes the action. For example, if the hardware read that the lights in classroom 1 should be on and they are off, it will turn them on.
3. There are certain times when the hardware needs to feed information to the system, for example, with the temperature sensor. In this case, it will take the ambient temperature and send it to the IoT framework, so that it is available to those who require it. As in previous cases, the IoT framework will only be in charge of storing and making the information available, without processing it.
4. Finally, there will be cases where the web application requires feedback from the system, such as, for example when the hardware reports the current temperature of a certain classroom. In this way, the Web Application will know what information to produce and send it to the IoT framework again.

5 Test Scenarios

The following scenarios will be used for testing the prototype.

1. A professor is far from the institution and has classes at that time.
2. A professor is at the institution and has classes at that time.
3. Functioning with different personalized professor profiles.

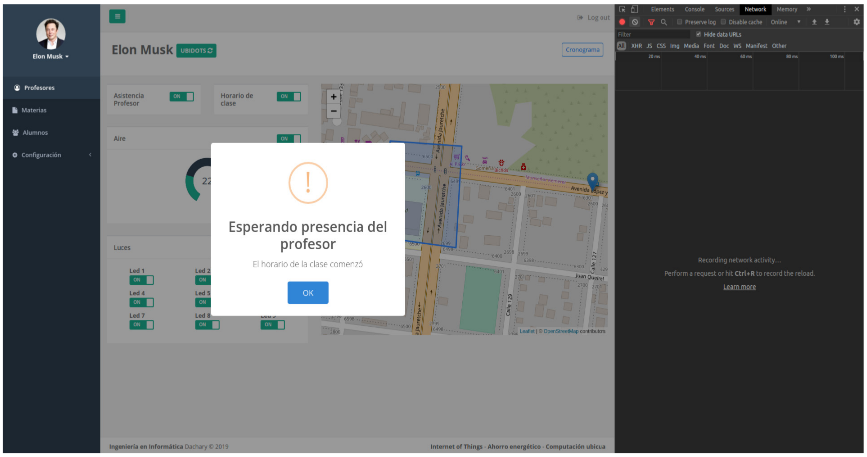
The activation condition of each profile includes the following three variables:

1. The professor is in the institution (position simulated by the marker on the map)
2. The professor has classes at that time.
3. The professor attends classes.

The actors involved are specified in each test. Also, if necessary, a different lighting and/or cooling profile is specified.

5.1 Scenario 1: A Professor is Far from the Institution and Has Classes at that Time

Actors Involved: Web Application.
In this test, the framework did not participate, because the one who sends the orders to later be read by the hardware, is the web application. In this case, with any lighting and cooling profile activated, and being class time for the professor, no request was sent to the IoT framework since the professor was far from the institution. Figure 4 shows in the "Network" tab how simulating any location of the professor on the map outside of the profile activation field, no request is sent.



Esperando presencia del profesor

El horario de la clase comenzó

Recording network activity...

Perform a request or hit Ctrl+R to record the reload.

Learn more

Fig. 4. Scenario 1

5.2 Escenario 2: A Professor is at the Institution and Has Classes at that Time

Actors Involved: Web Application, IoT Framework and Hardware.

Test Profile: All lights on and air conditioning on. Being the teacher's class schedule, the teacher being in the activation zone and marking that he attends classes in Fig. 5 we can see how the Web Application sent 10 requests correctly, corresponding to the 10 variables used.

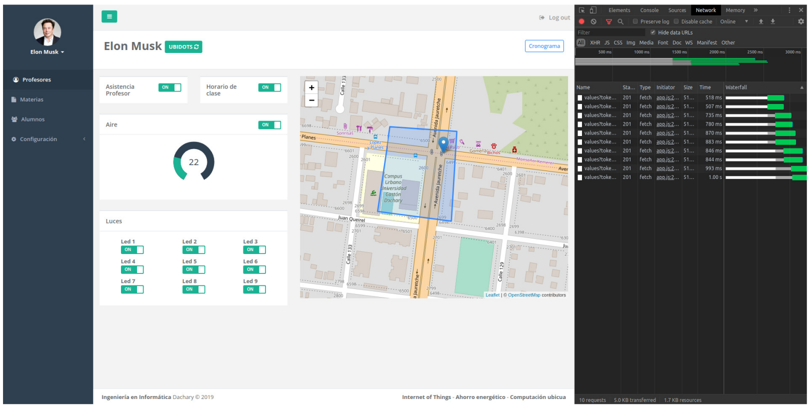

Elon Musk
Profesores
Materias
Alumnos
Configuración
Log out
Elon Musk UBIDOTS
Cronograma
Asistencia Profesor ON
Horario de clase ON
Aire ON
22
Luces
Led 1 ON
Led 2 ON
Led 3 ON
Led 4 ON
Led 5 ON
Led 6 ON
Led 7 ON
Led 8 ON
Led 9 ON
Campus Urbano Universidad Gastón Dachary
Avenida Jauretche
Leaflet | © OpenStreetMap contributors
Elements Console Sources Network Memory
Name Status Type Initiator Size Time Waterfall
10 requests 5.0 KB transferred 1.7 KB resources
Ingeniería en Informática Dachary © 2019
Internet of Things - Ahorro energético - Computación ubicua

Fig. 5. Web application sending power signal for lights and air conditioning using Ubidots IoT Famework.

When clicking on the first request sent by the Web Application, corresponding to the first variable "led1" of Ubidots, the request details are observed in Fig. 6:

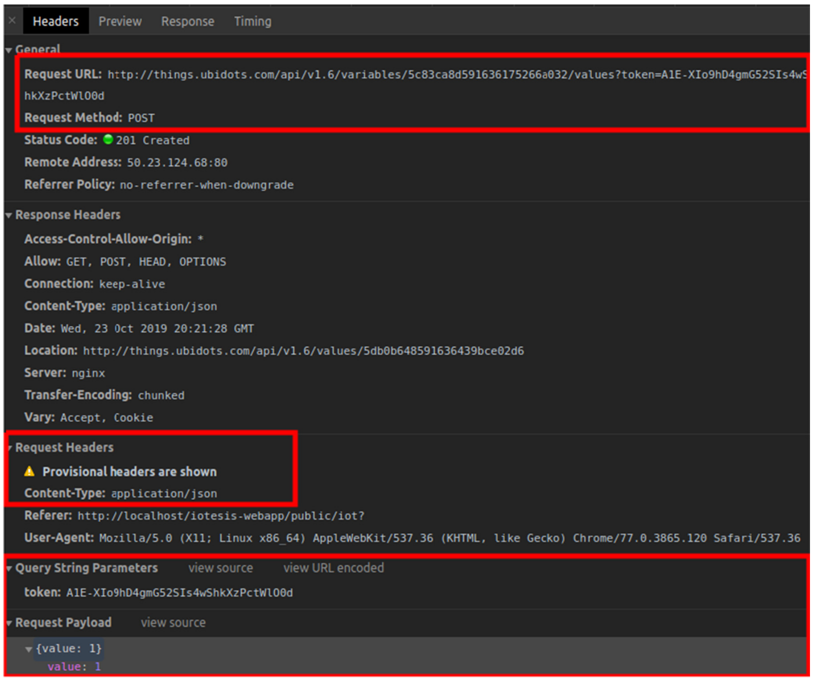

```
× Headers  Preview  Response  Timing
▾ General
  Request URL: http://things.ubidots.com/api/v1.6/variables/5c83ca8d591636175266a032/values?token=A1E-XIo9hD4gmG52SIs4wS
  hkXzPctWl0Od
  Request Method: POST
  Status Code: ● 201 Created
  Remote Address: 50.23.124.68:80
  Referrer Policy: no-referrer-when-downgrade
▾ Response Headers
  Access-Control-Allow-Origin: *
  Allow: GET, POST, HEAD, OPTIONS
  Connection: keep-alive
  Content-Type: application/json
  Date: Wed, 23 Oct 2019 20:21:28 GMT
  Location: http://things.ubidots.com/api/v1.6/values/5db0b648591636439bce02d6
  Server: nginx
  Transfer-Encoding: chunked
  Vary: Accept, Cookie
▾ Request Headers
  ⚠ Provisional headers are shown
  Content-Type: application/json
  Referer: http://localhost/iotesis-webapp/public/iot?
  User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/77.0.3865.120 Safari/537.36
▾ Query String Parameters    view source    view URL encoded
  token: A1E-XIo9hD4gmG52SIs4wShkXzPctWl0Od
▾ Request Payload    view source
  ▾ {value: 1}
      value: 1
```

Fig. 6. first request sent by the Web Application.

The Request URL shows the endpoint to which the request was sent, corresponding to the first variable (“led1”). Other important data are observed, such as:

– Status Code 201: The request was successfully sent.
– Content-Type application/json: One of the headers required by the framework.
– Token
– Variable ID in the URL.

In the following requests corresponding to the other lights and air, the same results were obtained. As seen in Fig. 7, it is verified that the data was written correctly in the Framework, using Ubidots Dashboard created for quick visualization.

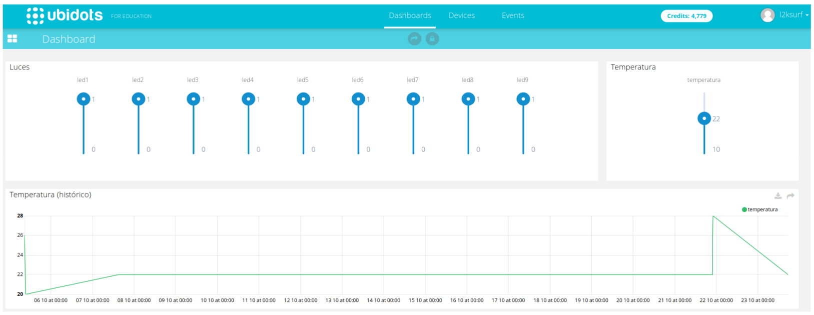


Fig. 7. Public Ubidots dashboard, showing widgets of all variables.

In the Classroom mockup where the hardware is installed, all the lights and the engine were turned on, as we can see in Fig. 8:

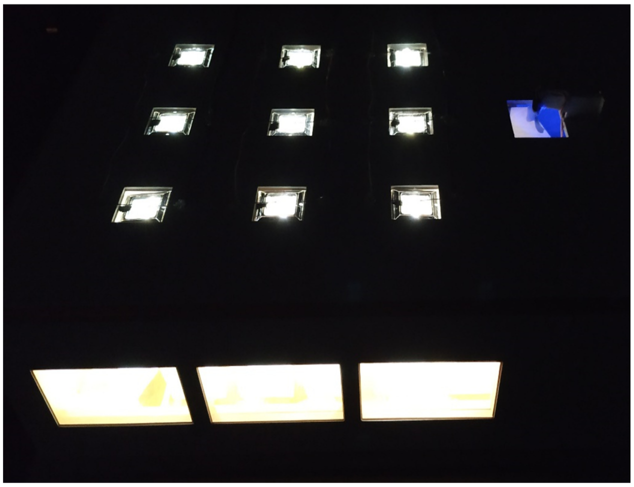


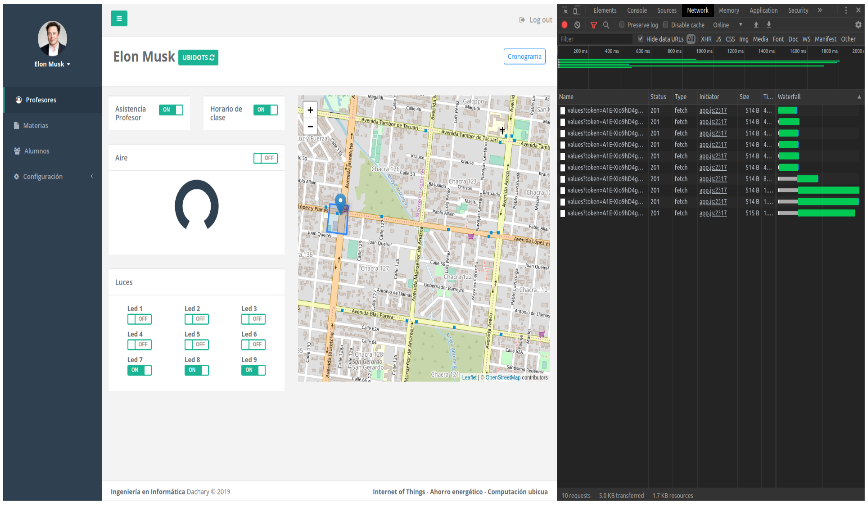
Fig. 8. A classroom mockup with all light and engines turned on.

5.3 Scenario 3: Functioning with Different Personalized Professor Profiles

Actors Involved: Web Application, IoT Framework and Hardware.

Test Profile: Only the last three lights turned on (in the classroom back) and air conditioning turned off.

The new profile with only the lights at the back of the classroom on (LEDs 7, 8, and 9) and the air conditioning off as we can see in Fig. 9.



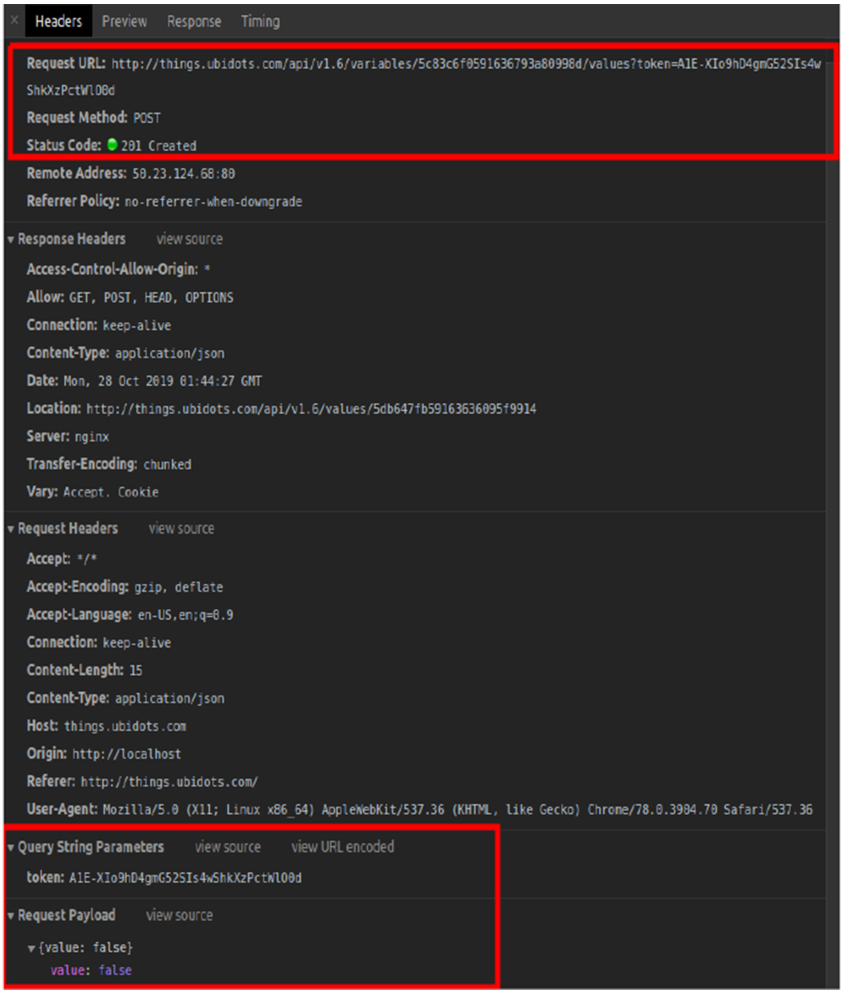
Name	Status	Type	Initiator	Size	Ti...
values?token=A1E-Xlo9hD4g...	201	fetch	app.js:2317	514 B	4...
values?token=A1E-Xlo9hD4g...	201	fetch	app.js:2317	514 B	4...
values?token=A1E-Xlo9hD4g...	201	fetch	app.js:2317	514 B	4...
values?token=A1E-Xlo9hD4g...	201	fetch	app.js:2317	514 B	4...
values?token=A1E-Xlo9hD4g...	201	fetch	app.js:2317	514 B	4...
values?token=A1E-Xlo9hD4g...	201	fetch	app.js:2317	514 B	4...
values?token=A1E-Xlo9hD4g...	201	fetch	app.js:2317	514 B	8...
values?token=A1E-Xlo9hD4g...	201	fetch	app.js:2317	514 B	1....
values?token=A1E-Xlo9hD4g...	201	fetch	app.js:2317	514 B	1....
values?token=A1E-Xlo9hD4g...	201	fetch	app.js:2317	515 B	1....

10 requests | 5.0 KB transferred | 1.7 KB resources

Fig. 9. Web application sending on signal only for the last three lights and air conditioning off, using Ubidots.

The profile was configured, to leave only the lights in the background of the classroom on, and you can see how the web application sent ten requests correctly. For the lights that should be on, he sent “value”: 1, and for those that should be turned off, he sent “value”: 0. The air conditioning, with a temperature lower than 24° C, turns off.

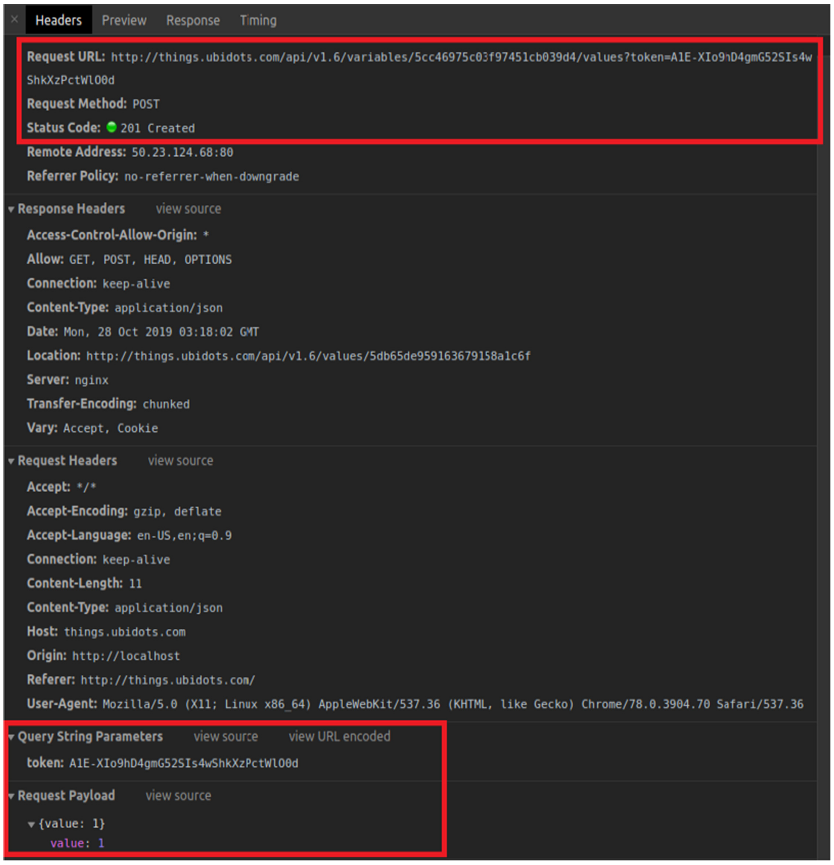
In the first request, it is observed that the Request Payload has the value of false (Fig. 10)



```
Headers   Preview   Response   Timing

Request URL: http://things.ubidots.com/api/v1.6/variables/5c83c6f0591636793a80998d/values?token=A1E-XIo9hD4gmG52SIs4w
ShkXzPctWl08d
Request Method: POST
Status Code: 201 Created
Remote Address: 50.23.124.68:80
Referrer Policy: no-referrer-when-downgrade

▾ Response Headers     view source
  Access-Control-Allow-Origin: *
  Allow: GET, POST, HEAD, OPTIONS
  Connection: keep-alive
  Content-Type: application/json
  Date: Mon, 28 Oct 2019 01:44:27 GMT
  Location: http://things.ubidots.com/api/v1.6/values/5db647fb59163636095f9914
  Server: nginx
  Transfer-Encoding: chunked
  Vary: Accept, Cookie

▾ Request Headers     view source
  Accept: */*
  Accept-Encoding: gzip, deflate
  Accept-Language: en-US,en;q=0.9
  Connection: keep-alive
  Content-Length: 15
  Content-Type: application/json
  Host: things.ubidots.com
  Origin: http://localhost
  Referer: http://things.ubidots.com/
  User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/78.0.3904.70 Safari/537.36

▾ Query String Parameters     view source     view URL encoded
  token: A1E-XIo9hD4gmG52SIs4wShkXzPctWl08d

▾ Request Payload     view source
  ▾ {value: false}
      value: false
```

Fig. 10. Request with signal to turn off “led1” variable.

The same happens with the requests for all the lights in the first two rows. On the other hand, for the last row of lights, we can see light turned on. To do this, we observe the request, the variable “led9” and Request Payload has true value, as we can see in Fig. 11:



```
Request URL: http://things.ubidots.com/api/v1.6/variables/5cc46975c03f97451cb039d4/values?token=A1E-XIo9hD4gmG52SIs4w
ShkXzPctWl00d
Request Method: POST
Status Code: 201 Created
Remote Address: 50.23.124.68:80
Referrer Policy: no-referrer-when-downgrade
▾ Response Headers     view source
   Access-Control-Allow-Origin: *
   Allow: GET, POST, HEAD, OPTIONS
   Connection: keep-alive
   Content-Type: application/json
   Date: Mon, 28 Oct 2019 03:18:02 GMT
   Location: http://things.ubidots.com/api/v1.6/values/5db65de959163679158a1c6f
   Server: nginx
   Transfer-Encoding: chunked
   Vary: Accept, Cookie
▾ Request Headers     view source
   Accept: */*
   Accept-Encoding: gzip, deflate
   Accept-Language: en-US,en;q=0.9
   Connection: keep-alive
   Content-Length: 11
   Content-Type: application/json
   Host: things.ubidots.com
   Origin: http://localhost
   Referer: http://things.ubidots.com/
   User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/78.0.3904.70 Safari/537.36
▾ Query String Parameters     view source     view URL encoded
   token: A1E-XIo9hD4gmG52SIs4wShkXzPctWl00d
▾ Request Payload     view source
   ▾ {value: 1}
       value: 1
```

Fig. 11. Request with signal to turn on "led9" variable.

As seen in Fig. 12, everything worked as we expected in the Ubidots Dashboard.

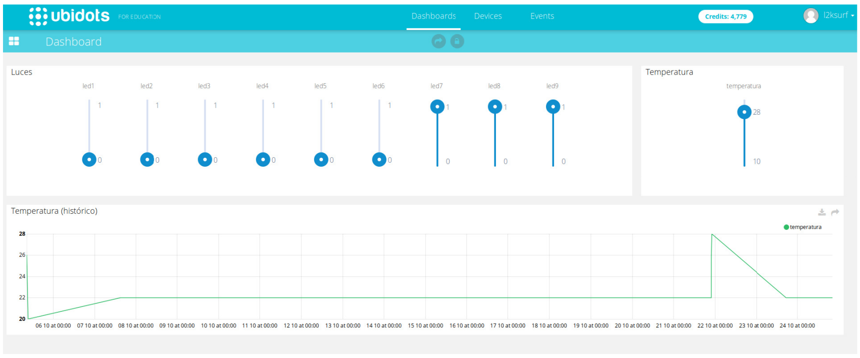


Fig. 12. Ubidots dashboard, showing the devices that must be turned on.

The hardware in the same way, it correctly read the framework data and executed the action (Fig. 13).

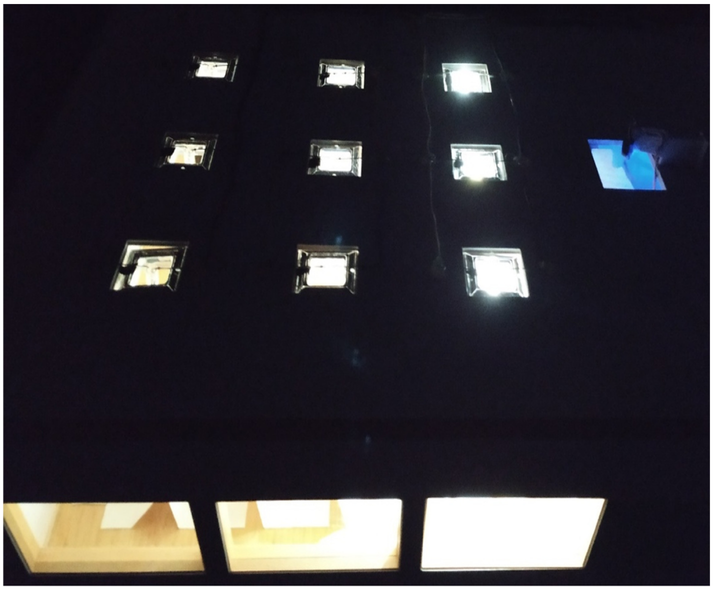


Fig. 13. The classroom mockup in the dark with only the last three lights are on.

6 Conclusions and Future Works

The use of the IoT Ubidots framework for the development of a Prototype of Classroom Energetically Efficient allowed to increase the implementation speed and the time saved in the construction of systems that need the interaction of devices connected to the internet. This is mainly due to the fact that it has the communication already resolved, having to focus only on the construction of the parts that must communicate.

Taking into account that all the packages that allow communication between the parties over the internet, using HTTP, the framework must guarantee the tools to give the necessary security to the packages, having SSL certificates and some method of authentication for the requests, either a token or an API_Key.

Finally, an efficient optimization in the time that lighting and cooling devices are on, avoiding idle time, translates into lower consumption, lower costs, and therefore less environmental pollution and greater energy efficiency.

Future work includes: Implement the prototypes with the MQTT protocol and later make the comparison against HTTP protocol used in this work.

Refactor the Ubidots firmware code for the NodeMCU ESP8266, improving the data reading and writing algorithms, to get better operations performance.

Design and implement fully functional versions of the web application which allow executing in a real way the change of teacher profiles as well as the dynamic configuration of the same, for each subject and schedule. Use the geo-positioning of a mobile device, for example, a cell phone, to locate the teacher, no just simulating the localization. Implement the prototype, in real classrooms of an educational institution to be used as a living laboratory.

References

1. Sosa, E.O., Godoy, D.A., Benítez, J., Sosa, M.E.: Eficiencia Energética y Ambientes Inteligentes. Investigación y Desarrollo Experimental en la UNaM, Posadas, Misiones, Argentina (2015)
2. Godoy, D., Bareiro, H., Fabret, F., et.al.: Propuesta de métricas para comparación de Framewoks IoT. In: Workshop de Investigadores en Ciencias de la Computación., El Calafate (2020)
3. Godoy, D., Bareiro, H., Fabret, F., et.al.: Análisis de componente de hardware para la utilización con Frameworks de IoT. In: Workshop de Investigadores en Ciencias de la Computación., El Calafate (2020)
4. Frimpong, E.A., Appiah, F.: Energy efficiency awareness and preparedness among students (2017)
5. Perez, J.A.: Proyectos de Eficiencia Energética para Instituciones de Educación Superior
6. González-Amarillo, C.-A., C.-G.-A., C.-L.: Smart lumini: a smart lighting system for academic environments using IOT-based open-source hardware. Revista Facultad de Ingeniería **29**(54) (2020)
7. Gupta, A., Gupta, P., Chhabra, J.: IoT based power efficient system design using automation for classrooms. In: Third International Conference on Image Information Processing (ICIIP), Waknaghat, India (2015)
8. Ani, R., Krishna, S., Akhil, H., Arun, U.: An Approach towards building an IoT based smart classroom. In: 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI), Bangalore, India (2018)
9. NodeMCU: Wikipedia. In: NodeMCU. https://www.nodemcu.com/index_en.html. Accessed 2020
10. Programar facil.. https://www.programarfacil.com/esp8266/como-programar-nodemcu-ide-arduino/
11. Vuejs. https://vuejs.org/v2/guide/. Accessed Aug 2020
12. Ubidots. https://ubidots.com/docs/. Accessed Aug 2020
13. Hernandez, M.: Ubidots Basics: Devices, Variables, Dashboards, and Alerts. https://help.ubidots.com/en/articles/854333-ubidots-basics-devices-variables-dashboards-and-alerts. Accessed Aug 2020