INTERNATIONAL JOURNAL FOR INNOVATION EDUCATION AND RESEARCH
ISSN: 2411-2933

Journal URL: https://scholarsjournal.net/index.php/ijier

DOI: https://doi.org/10.31686/ijier.vol10.iss9.3891

A review of Arduino and salesforce for data collection and storage designed for microclimate monitoring

Mariana Espindola Vieira[1], Erondina Azevedo de Lima[2], Lenildo Santos da Silva[3] and Valmor Pazos Filho[4]

[1] Bachelor in Computer Science, Federal University of Pelotas;

[2] Department of Physics, University of Brasilia

[3] Department of Civil Engineering, University of Brasilia

[4] Department of Civil Engineering, UNIPLAN

Abstract

This paper discusses the creation of a prototype station for monitoring microclimate and other variables such as carbon dioxide air concentration level and globe temperature through components that gather, process and store data from low-cost sensors. Its general purpose is to provide data on climate and other more granular features, such as microclimate. The specific objective was to seek solutions for the issue of securely storing microclimate data collected by the prototype using an Arduino open-source hardware board. The proposal to use Salesforce's PaaS computational cloud for data storage allows for the existence, with the current state of technology, of several tools capable of assisting both in collecting data as well as in securely storing them for later analysis. The study and the creation of such a prototype station for monitoring microclimate and other variables were carried out in the city of Campinas/SP. In the end, it was concluded that the station was efficient due to the accuracy of the data obtained by most sensors interfaced with Arduino and recorded on Salesforce's platform with the use of the NodeMCU Wi-Fi module. Some objectives were not met as one of the sensors was faulty and the integration between Arduino and NodeMCU could not be completed due to time constraints.

Keywords: sensors; globe temperature; carbon dioxide; microcontroller; cloud computing.

1. INTRODUCTION

1.1 The importance of microclimate monitoring

Microclimate monitoring is of great relevance for societies and urbanistic development, as well as for verifying environmental balance and sustainability. Monitoring data from the air and other natural resources also contributes to academic studies in these fields (SANTOS *et al.*, 2014). In addition to these contributions, microclimate monitoring can also assist bioclimatic architecture with thermal comfort verification.

The main form through which human beings organize themselves to live in society are the cities (LOPES, 1998). As a consequence of the evolution of urban centers, phenomena such as deforestation or the removal of native vegetation, soil sealing, river canalization and the construction of buildings and traffic lanes

for fuel-burning vehicles are taking place in cities. All of these practices contribute to the formation of Urban Heat Islands (UHI). UHIs become more hazardous in places that have higher urban density, entailing the use of artificial resources such as HVAC systems to control thermal comfort for people (ROMERO, 2013; ROMERO, M. A. B. *et al.*, 2019).

According to Lamberts, Dutra and Pereira (2014), "the microclimate can be designed and changed by the architect". With this statement, it is understood that the architect can play a very important role in the microclimate of urban areas by designing buildings that contribute to thermal comfort. Therefore, for the development of architecture, it is important for the architect to conduct studies on the microclimate in order to mitigate or avoid the bad effects of heat islands.

Technologies and their tools can assist in microclimate studies and research. Together with architecture professionals, the technology sectors can contribute to the development and progress of bioclimatic architecture in sustainable environmental rehabilitation of urban environments.

1.2 Climate and microclimate monitoring

In Brazil, the Instituto Nacional de Meteorologia [National Institute of Meteorology] (INMET) – currently under the management of the Ministério da Agricultura, Pecuária e Abastecimento [Ministry of Agriculture, Livestock and Supply] (MAPA) – is responsible for providing free meteorological information to the population. INMET collects weather-related data on temperature, relative humidity, wind direction and speed, atmospheric pressure and precipitation, among others, through Upper Air Sounding Systems (UASS), Surface Weather Observation Stations (ASOS/AWOS) and an Automatic Weather Station (AWS) Network (INMET, 2020). While it is possible to use those data to carry out studies on climate, INMET does not provide the more specific information that characterizes the microclimate. INMET's usage stations are fixed and do not capture microclimate in its granularity. Thus, as an alternative for studying and monitoring climate, some researchers use the general data provided by INMET, resorting to maneuvers such as increasing or decreasing degrees according to other parameters.

In a particular experiment, according to Romero and Vavallo (2015), the INMET database was used, but with an adjustment to correct the difference in altitude between the exact location of analysis and the location of the Institute's meteorological station. With the use of a sensor at a different location from the analysis site, it is understood that microclimate values may change. Therefore, using a portable, wireless microclimate measurement tool can be more accurate for on-site data collection.

Another downside of relying solely on the INMET database is that, in addition to the issues related to the lack of precision in results, there is also the human factor and the fact that reading occurs on a sparse schedule. INMET's conventional method of climate monitoring also has flaws in data integrity. In this sense, Torres *et al.* (2015) point out:

For many years, the assessment of these data was done through conventional weather stations run by INMET, which installed them, while the responsible technicians collected the data generated by these stations only three times a day. This is a very low frequency of collections, and there is also the human factor in measurements - that generates errors of systematic or random nature - which can lead to large disparities in relation to

the real value (TORRES *et al.*, 2015).

Thus, the pursuit for better ways of monitoring microclimate points towards the need for greater accuracy and reliability of data. However, there is also a need to optimize the costs of monitoring by creating cheaper stations.

In order to perform the collection and analysis of meteorological data, it is necessary to use instruments and equipment to measure these characteristics. The equipment available on the market is usually expensive, hence the importance of the implementation and prototyping of low-cost weather stations (TORRES *et al.*, 2015; VIEIRA, 2018). The purpose of these climate monitoring stations is to measure physical attributes, such as temperature, relative humidity, atmospheric pressure, wind speed, among others (ELIAS *et al.*, 2014; VIEIRA, 2018). Contemporarily, with the use of technologies, several tools, devices and equipment have been developed to assist in the collection, measurement, monitoring, analysis and storage of these data.

This study aims to review and contribute to the work that has been developed on microclimate monitoring with the use of technological innovations, in order to obtain greater data accuracy and cost reduction in the creation of monitoring stations. In this sense, it will cover similar research addressing the usage of sensors and microcontrollers to gather climate data, as well as of Salesforce platform's cloud technology to store them. These are factors that can and should be considered for an efficient climate data monitoring experiment, by means of a low-cost, portable micro station that has the advantage of performing the collection at a desired location, with the desired reading frequency and period.

In this work, these reviews were conducted alongside the creation of a prototype station for monitoring microclimate and other parameters such as sound intensity level, carbon dioxide air concentration level and globe temperature through components that capture, process and store sensor data.

To that end, effort was made to provide data on microclimate and some other variables more influential and suitable for thinking comfort at a human scale, since these more specific microclimate data are not made available to the population by public agencies such as INMET, which only provides general data on climate. Furthermore, the study sought to work out the difficulties in the storage of data collected by sensors on the Arduino board using the Salesforce platform.

1.3 Use of Arduino for monitoring

The creation of a prototype station for monitoring microclimate and other variables requires the use of sensors to capture data as well as a device capable of supplying them with electrical power while simultaneously controlling and processing the signals picked up by the sensors. This device can be a programmable microcontroller, which, by the use of code and mathematical formulas (cf. Appendix B), fulfills that intent.

In Vieira (2018) and in Müller (2019), studies in the areas of technology and computing were reviewed to develop comfortmeter prototypes with the aim of measuring the internal microclimate for bioclimatic architecture. They used Arduino[1] microcontrollers, as they are open-source hardware compatible with a wide

[1] Available at: https://www.arduino.cc/. Accessed on: 6/23/2021. Arduino is an Italian company that manufactures microcontroller boards. Today there already are several board models, the most popular being the Arduino NANO, Arduino UNO and Arduino MEGA. These Arduino boards can be programmed in Arduino's own development environment, on any operating system. Its language is simple, structured and similar to C++

range of low-cost sensors. Air and globe temperature, relative humidity, wind speed and atmospheric pressure sensors were used to assess climate data in these authors' research on architecture and urbanism.

Arduino is an open-source hardware tool, which is programmable and can be connected to sensors, actuators and various modules, including those intended for Wi-Fi internet connection. Microcontrollers such as Arduino have been used in several works, such as: Schmidt *et al.*, (2020), Torres *et al.* (2015), Brito *et al.* (2017), Kusriyanto and Putra (2018), Üçgün and Kaplan (2017) and Simões (2017). These authors have conducted research using Arduino to collect climate data. Schmidt *et al.* (2020) and Torres *et al.* (2015) used memory card modules adapted for Arduino to store the collected data. Brito *et al.* (2017) used a Raspberry PI[2] microcomputer connected to Arduino to establish a connection with a database server in order to store the records and host a website for accessing them. Simões (2017) also used a database server on his research. Kusriyanto and Putra (2018) used Arduino in combination with both a memory card and a Wi-Fi module (ESP8266), the latter being responsible for sending the collected data to be stored in the cloud at dweet.io[3] along with freeboard.io[4], where graphs and dashboards can be generated on the collected data.

1.4 Cloud computing

Cloud computing provides space for storing and accessing data in files via software systems on a server connected to the internet with access security. As a result, it can be accessed by devices that connect to the internet, such as computers, smartphones, tablets and Wi-Fi modules for microcontrollers (SALESFORCE PAAS, 2020).

These clouds are provided in 3 different standard service models:

a) **SaaS:** Software as a Service – the system is already developed and ready for the end user, without customization flexibility;
b) **PaaS:** Platform as a Service – the system is partially developed and can be customized to meet the user's needs; and
c) **IaaS:** Infrastructure as a Service – the system is developed by the consumer and the cloud only provides the server and access security.

(VIEIRA, 2017).

[2] Available at: https://www.raspberrypi.org/. Accessed on: 6/23/2021.

[3] Available at: http://dweet.io/. Accessed on: 6/23/2021.

[4] Available at: https://freeboard.io/. Accessed on: 6/23/2021.

Table 1. Main differences between cloud computing service models

Cloud Model	**Software Development and Update**	**Hardware Installation and Maintenance**
Infrastructure as a Service	Not provided	Provided by Provider
Platform as a Service	Allows application customization	Provided by Provider
Software as a Service	Provided by Provider	Provided by Provider

Source: Adapted from Salesforce PAAS, 2020.

Thus, it is possible to use the internet and cloud computing to send collected climate data to an online server, as shown in "Confortímetro Lotus: Um Sistema Móvel de Baixo Consumo" [Lotus Comfortmeter: A Low Consumption Mobile System], a bachelor dissertation by Christian Müller (2019), wherein a prototype comfortmeter was produced using a microcontroller with sensors connected to a web server in order to store climate data.

In this work, other comfortmeter prototypes were also compared, whereof 4 out of the 6 models studied have wireless connection and 5 of them have smartphone applications designed either towards gathering data from users on site, such as in Zhao, Uduku and Murray-rust (2017), or towards controlling comfort in environments cooled by air-conditioning systems, as in Jazizadeh *et al.* (2014).

Table 2 – Prototypes listed in Müller's work

Works	Wi-Fi	Energy Efficiency	Portable	App
Zhao	Yes	No	No	Yes
Gao	No	No	No	Yes
Zhao; Uduku	Yes	No	No	Yes
Jazizadeh	Yes	No	Yes	Yes
Vieira	No	No	Yes	No
Present work	Yes	Yes	Yes	Yes

Source: Müller (2019).

For the purpose of the present study, a Salesforce cloud solution was chosen. Despite being a commercial cloud platform for sales services, it provides a development environment at no cost, including the full-featured software, the sole restriction being individual access limitation. This development environment is

created on Trailhead[5], the tool's study and testing portal.

In this environment, it was possible to create a cloud and begin the present work's first tests. The Salesforce Cloud is a Platform as a Service (PaaS). What is interesting about this choice is the ability to create an individual, secure platform with a pre-existing Salesforce system, which allows for object programming in a codeless, functional way; it also allows for customization through automation codes for receiving data via other systems' interface.

2. METHODS

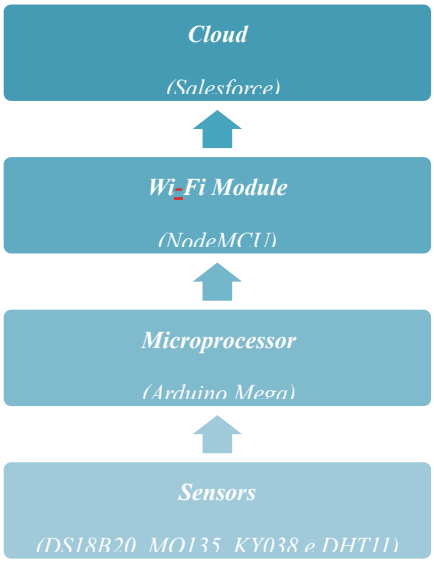
As stated, the creation of a prototype station for monitoring microclimate and other variables requires the use of sensors that gather these data from the study site. A device that controls and processes the signals captured by these sensors must be added to the station thereafter. Once that is done, it is also necessary to choose a safe storage practice for these data.

Technologies and their various tools have greatly contributed to studies and research on climate attributes. In this work, an Arduino microcontroller was interfaced with sensors suitable for microclimate studies with the aim of measuring the carbon dioxide air concentration level, globe and air temperature, relative humidity and sound intensity level in order to communicate with Salesforce's development cloud, via an Arduino-compatible Wi-Fi module.

Subsequently, to illustrate the broad process of building this Internet of Things (IoT) system, a diagram was made delineating the communication relationship between the four tools used, as shown in Figure 2. In the first stage, the sensors and the microcontroller were wired up together. A network connection between the Wi-Fi module and the cloud was thereupon established in the second stage. Ultimately, the only remaining step to be accomplished so that the general purpose of this work could be attained and an IoT system for microclimate monitoring could be built was a hookup between the micropocessor and the Wi-fi module.

[5] Available at: https://trailhead.salesforce.com/en/home. Accessed on: 6/23/2020.

Figure 1 – Schematic model for transmitting sensor data to the cloud



Source: Based on Bhadoriya *et al.*, 2016.

Like so, this experiment took place in stages:

i) The first consisted of attaching the components to the breadboard, programming the Arduino Mega microcontroller and reading sensor data using Arduino Integrated Development Interface (IDE) on a computer;

ii) The second encompassed the preparation of the environment in a Salesforce development cloud and the programming of the NodeMCU Wi-Fi module with Arduino IDE, as well as the connection of this module to Salesforce over Wi-fi network; and

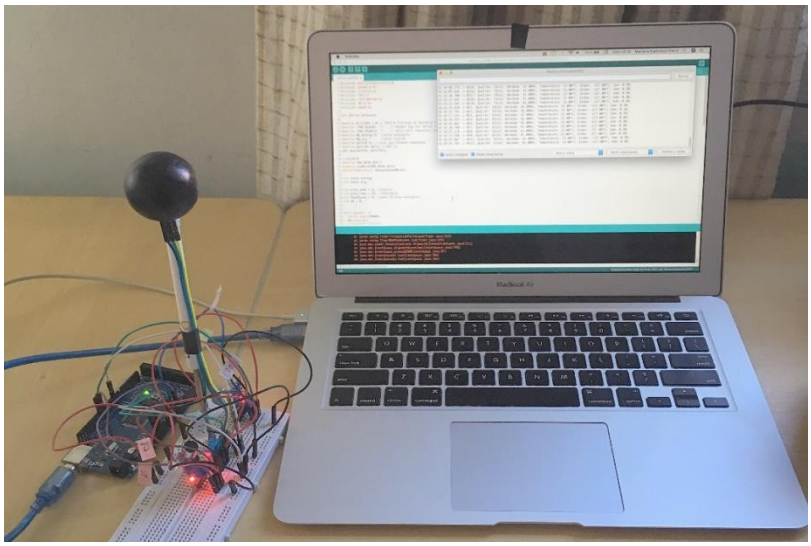
iii) A third stage, wherein Arduino Mega was to be connected to the NodeMCU module, was planned at first but was not fulfilled due to this article's deadline.

As a result, it was not possible to completely transfer the data collected by the prototype to the cloud, in order to securely store those records.

2.1 First stage

The first stage carried out in this experiment covered data collection, whereat 4 breadboarded sensors were connected to Arduino Mega, which in turn was connected to a computer via USB cable, whereupon data received from the sensors was stored in the Serial Monitor with Arduino IDE. This stage is represented in Figure 2. Details on how the sensors were placed on the breadboard in connection with Arduino can be found in Appendix A of this paper.

Figure 2 – Prototype connected to the Arduino IDE on the computer with the serial monitor receiving data collected from the sensors



Source: Authors' collection, 2020.

In order to analyze the variables carbon dioxide air concentration level, globe temperature, sound intensity level, relative humidity, and air temperature, the following sensors were selected:

MQ135 – gas sensor for air quality

The MQ135 sensor detects toxic substances in the air in a range from 10ppm to 200ppm, including carbon dioxide. The sensor is interfaceable with Arduino and can be calibrated (MQ135, 2020; SAI *et al.*, 2019);

DS18B20 – globe thermometer

Globe thermometers are typically used to obtain the Mean Radiant Temperature (MRT), an important unit of measurement for the calculation of thermal comfort indexes. To this end, the DS18B20, a low-cost digital temperature sensor compatible with Arduino, was inserted into a ping-pong ball painted in black paint (THORSSON *et al.*, 2007; VIEIRA *et al.*, 2017; VIEIRA, 2018);

KY-038 – sound sensor module

KY-038 is a sound intensity sensor capable of picking up sound variations in its proximity. It is possible to adjust its sensitivity by rotating the built-in potentiometer (SALLES, 2017); and

DHT11 – relative air humidity and temperature sensor

The DHT11 is a temperature and relative humidity sensor; it is compatible and performs well with Arduino, whilst capable of being connected to both analog or digital ports. However, this sensor may produce readings inaccurate by ±2.0ºC (plus or minus two degrees Celsius) as well as ±5% (plus or minus five percent) in the perception of relative humidity (VIEIRA, 2018).

The total amount spent on prototype components, not factoring in the computer, the breadboard and wires used for the assembly of the monitoring station, tallied 220.00 BRL (two hundred and twenty reais).

Table 3 – Components used

Component description	Model	Cost
Humidity and Temperature	DTH11	20.00 BRL
Globe Temperature	DS18B20	15.00 BRL
CO2 Percentage	MQ135	35.00 BRL
Sound intensity	KY-038	11.00 BRL
Wi-Fi Module	NodeMCU	39.00 BRL
NodeMCU	Arduino Mega	100.00 BRL
Total		220.00 BRL

Source: FilipeFlop's E-commerce website, 2020.

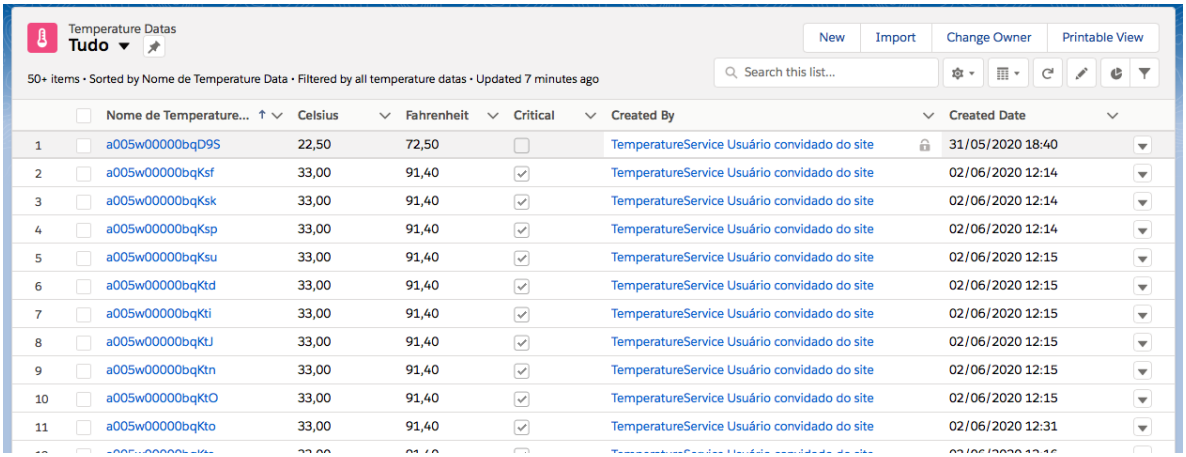
As can be seen in table 3, all the components, including the microcontroller, are low-cost and therefore make monitoring studies quite feasible.

2.2 Second stage

The second step was based on Wickramasinghe's tutorial available on MANELSOFT's[6] website, wherein a NodeMCU Wi-Fi module was used to send data collected by a temperature sensor to Salesforce. The choice of a NodeMCU Wi-Fi module for this study, as well as of the procedures and methods for connecting it to the Salesforce cloud, was made in accordance with this tutorial.

[6] Available at: http://www.manelsoft.com/projects/nodemcu_sfdc_temperature_service.aspx. Accessed on: 6/23/2020

Figure 3 – Screenshot of Salesforce's testing environment, showing the list of records sent by NodeMCU connected to the Wi-Fi network



	Nome de Temperature...	Celsius	Fahrenheit	Critical	Created By	Created Date
1	a005w00000bqD9S	22,50	72,50	☐	TemperatureService Usuário convidado do site	31/05/2020 18:40
2	a005w00000bqKsf	33,00	91,40	☑	TemperatureService Usuário convidado do site	02/06/2020 12:14
3	a005w00000bqKsk	33,00	91,40	☑	TemperatureService Usuário convidado do site	02/06/2020 12:14
4	a005w00000bqKsp	33,00	91,40	☑	TemperatureService Usuário convidado do site	02/06/2020 12:14
5	a005w00000bqKsu	33,00	91,40	☑	TemperatureService Usuário convidado do site	02/06/2020 12:15
6	a005w00000bqKtd	33,00	91,40	☑	TemperatureService Usuário convidado do site	02/06/2020 12:15
7	a005w00000bqKti	33,00	91,40	☑	TemperatureService Usuário convidado do site	02/06/2020 12:15
8	a005w00000bqKtJ	33,00	91,40	☑	TemperatureService Usuário convidado do site	02/06/2020 12:15
9	a005w00000bqKtn	33,00	91,40	☑	TemperatureService Usuário convidado do site	02/06/2020 12:15
10	a005w00000bqKtO	33,00	91,40	☑	TemperatureService Usuário convidado do site	02/06/2020 12:15
11	a005w00000bqKto	33,00	91,40	☑	TemperatureService Usuário convidado do site	02/06/2020 12:31

Source: Authors' collection, 2020.

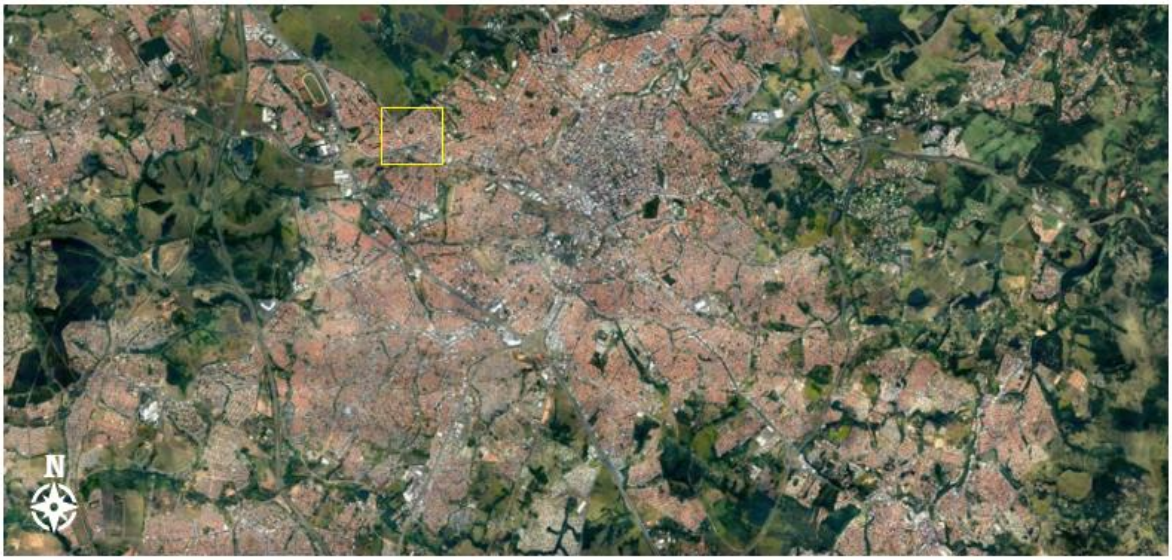
Figure 3 exhibits some communication tests between the Wi-Fi module and Salesforce, with the aim of creating records in the development cloud. Since the communication between Arduino Mega and NodeMCU could not be completed, the data collected by the sensors could not be sent to Salesforce's testing environment. That being said, the same Figure 3 exclusively shows the logs created from these tests, meaning that no realistic value captured by a temperature sensor was used in the code (Appendix C), wherein an arbitrary temperature value was used instead. The code used to connect NodeMCU to this work's test environments's website is included in Appendix C, whilst other configuration and programming procedures necessary for Salesforce's development cloud are contained in MANELSOFT's tutorial.

As noted in relation to the effort made to store the data gathered by the sensors in the cloud, all stages were concluded except for the connection between Arduino Mega and NodeMCU, which remained unattended in virtue of schedule.

3 RESULTS

The totality of developments and technical trials of this work (including monitoring and data collection) were carried out indoor, in a residential setting, in the city of Campinas, in the interior of the state of São Paulo, during a period of social isolation resulting from the covid-19 pandemic (BRASIL, 2020).

Figure 4 – Satellite image of the metropolitan region of Campinas/SP



Source: Google Earth, 2020.

Figure 4 shows, in yellow highlight, the region where data collection was carried out. According to Monteiro (2018) and Kaloustian (2016), the climate zone of this region is classified as LCZ31, as it is a residential neighborhood with low-rise, red-roofed houses and little vegetation. Based on Weather Spark's[7] "average weather conditions for Campinas and region", it is known that Campinas usually has a mild climate, with temperatures that typically vary from 13°C to 29°C, with higher average relative humidity in the period from October to April (during spring and summer) and lower from April to October (during autumn and winter).

Data collection took place in winter, on June 6, 2020, over a period of nearly 6 hours of reading from 5:17:59 pm to 11:11:23 pm. A total of 10,310 lines of sensor readings were registered for each of the 5 discrete variables in Table 3, amounting to 51,550 records, in this order: carbon dioxide PPM; relative humidity percentage; air temperature in degrees Celsius; globe temperature, also in degrees Celsius; and sound pressure level in decibels.

Table 4 – First readings according to Arduino IDE's Serial Monitor

Reading	Time (24h)	CO2 ppm	Humidity %	Temperature °C	Globe temperature °C	Sound intensity
000001	17:17:59	54.00	65.00	26.00	23.94	0.29
000002	17:18:01	54.00	65.00	26.00	23.94	0.30

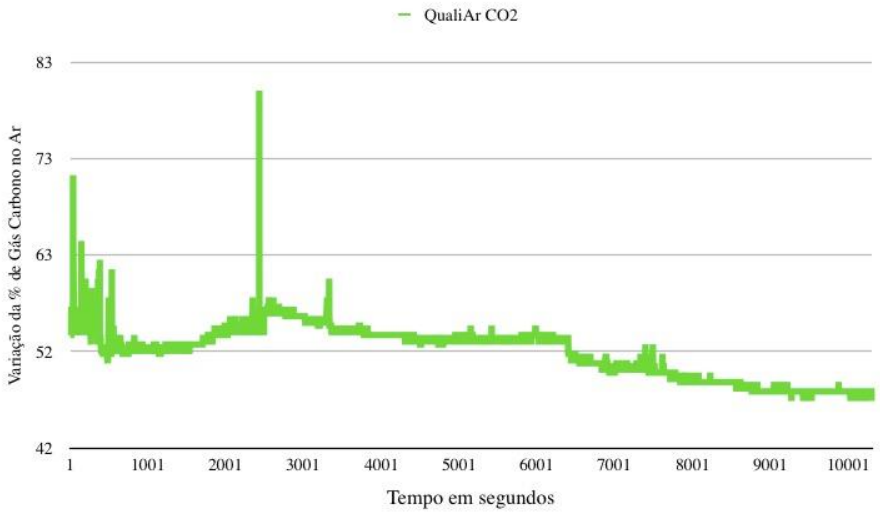
[7] Available at: https://pt.weatherspark.com/y/30197/Clima-caracter%C3%ADstico-em-Campinas-e-Regi%C3%A3o-Brasil-durante-o-ano. Accessed on: 7/13/2020.

000003	17:18:03	54.00	65.00	26.00	23.94	0.29
000004	17:18:05	54.00	65.00	26.00	23.94	0.29
000005	17:18:07	54.00	65.00	26.00	23.94	0.30

Source: authors, 2020.

The data collected in this experiment were recorded in the Arduino IDE's Serial Monitor. These data remain on the computer's cache memory, therefore being prone to permanent data loss if the program is closed. To prevent this from happening, these readings were copied to a text file and saved locally on the computer's hard drive as soon as the experiment was over. With this text file, it was possible to generate charts on each of the variables by time in seconds.

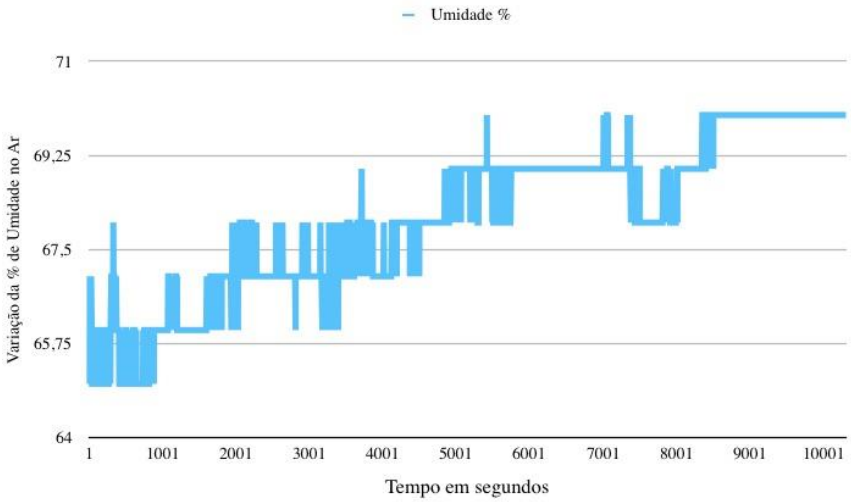
Figure 5 – MQ-135 sensor's carbon dioxide data chart



Source: authors, 2020.

As can be seen in Figure 5, a flammable element was burned just before the experiment so as to release a larger than normal amount of carbon dioxide into the environment, thereby making it possible to verify that the CO_2 ppm rates were higher at first and dropped over time.

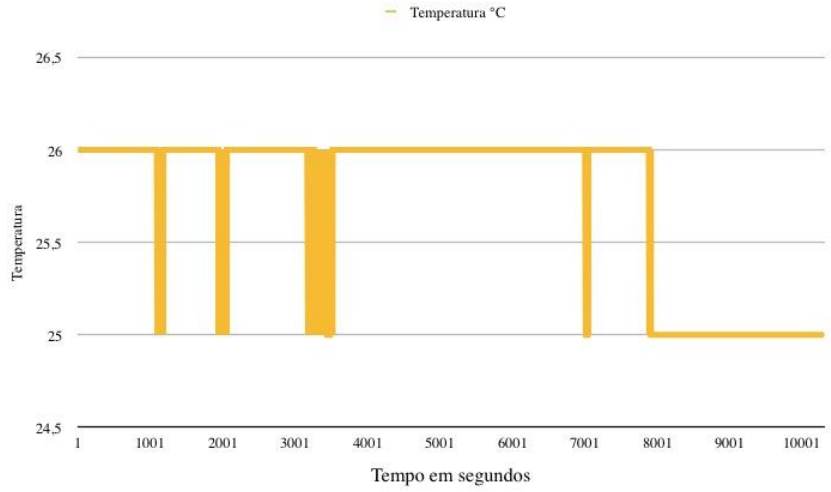
Figure 6 – DHT11 sensor's relative humidity data chart



Source: authors, 2020.

The DHT11 sensor is able to measure the variables of air temperature and relative humidity. From Figure 6, it can be seen that the relative humidity increased over time, after it started to rain during the experiment.

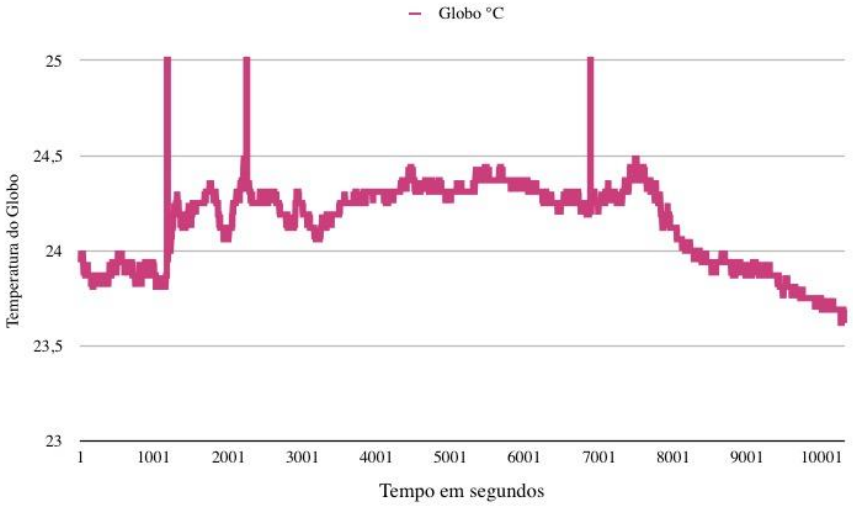
Figure 7 – DHT11 sensor's temperature data chart



Source: authors, 2020.

Along with relative humidity, the DHT11 sensor also measured air temperature, but, as shown in Figure 7, it is worth noting that this sensor's temperature measurement averaged aout to around 25.5°C, which is considerably less accurate in terms of temperature variations, when compared to the DS18B20 globe temperature sensor.

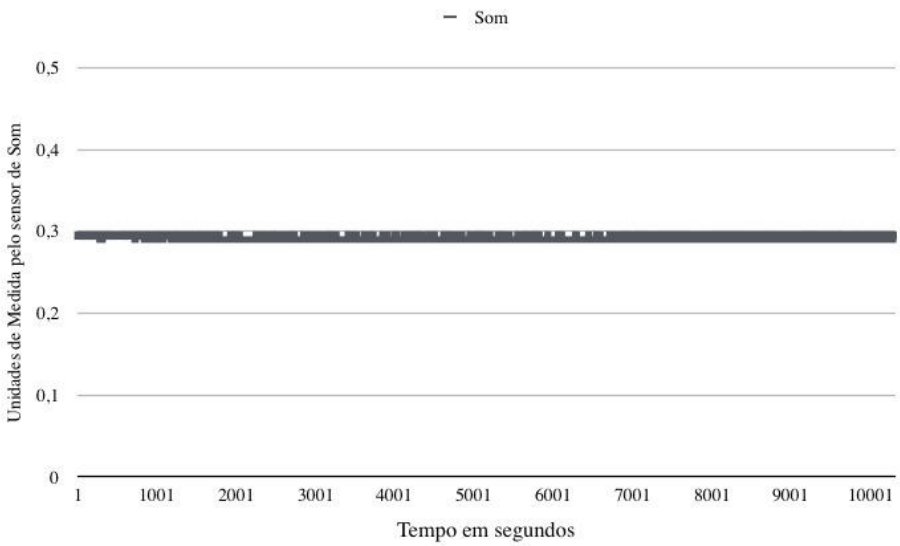
Figure 8 – Globe temperature sensor's data chart



Source: authors, 2020.

In the analysis of the globe temperature sensor data in Figure 8, it was possible to observe that the temperature varied around 24°C, while presenting three outliers. These discrepant readings occurred due to changes in the microclimate of the test environment attributable to the purposeful flick of a lighter, in an effort to detect brief and transient changes in the experiment. This provides insight into the importance of high frequency in data reading to model both the analog and the continuous variants of reality.

Figure 9 – KY-038 sound detection sensor's data chart



Source: authors, 2020.

In order to read the analog and digital outputs from the sound sensor and convert them to decibels, a logarithmic formula had to be obtained and applied. Nevertheless, a small constant variation was detected,

which was not proportional to ambient sound. The KY-038 sensor, though successfully used by Salles (2017), did not achieve a satisfactory result in this work.

FINAL THOUGHTS

After the present study, it was verified that microclimate monitoring can contribute significantly to the observation of heat islands in urban centers and to the monitoring of climate change in the world. It was also found that there are already several researches using Arduino interfaced with sensors for the measurement of climate and microclimate features, whether these prototypes are connected to the internet or to the cloud, or to a data storage module such as an SD memory card, with no connection to the internet. But since the cost of implementing such a prototype connected to the cloud is low, its use is relevant, because nothing else makes remote and continuous monitoring possible.

It can be argued that the combination of Arduino and Salesforce expands storage capacity and improves data security as well, but both an increase in collection time and the installation of a microcontroller with a Wi-Fi module in the station are advised. It was also found that remote and continuous monitoring is made possible by cloud computing.

The prototype station built for this study proved to be efficient in terms of correctly gathering data from sensors other than KY-038, which presented a manufacturing defect. The procedures of the second stage of the experiment, *i. e.*, to connect the NodeMCU Wi-Fi module to a local wireless network and to send the command for the creation of records in Salesforce's cloud computing environment, based on Wickramasinghe's tutorial, were also succesfully accomplished. Finally, only the integration between Arduino and NodeMCU was left unattended, due to time constraints.

REFERENCES

BHADORIYA, R.; CHATTOPADHYAY, M. K.; DANDEKAR, P. W. Low cost IoT for laboratory environment. *In*: **SYMPOSIUM ON COLOSSAL DATA ANALYSIS AND NETWORKING** (CDAN), 2016, Indore. DOI: 10.1109/CDAN.2016.7570939. Available at: https://ieeexplore.ieee.org/document/7570939. Accessed on: 23 jun. 2020.

BRASIL, Conselho Nacional de Saúde. **Recomenda aos Poder Executivo, federal e estadual, ao Poder Legislativo e ao Poder Judiciário, ações de enfrentamento ao Coronavírus.** Recomendação n.º 027, de 22 de abril de 2020. Available at: http://conselho.saude.gov.br/recomendacoes-cns/1132-recomendacao-n-027-de-22-de-abril-de-2020. Accessed on: 30 jun. 2020.

BRITO, R. C.; FAVARIM, F.; CALIN G.; AND TODT, E.; Development of a low cost weather station using free hardware and software. *In*: **LATIN AMERICAN ROBOTICS SYMPOSIUM (LARS) e 2017 BRAZILIAN SYMPOSIUM ON ROBOTICS** (SBR), 2017, Curitiba. DOI: 10.1109/SBR-LARS-R.2017.8215292. Available at: https://ieeexplore.ieee.org/document/8215292?section=abstract. Accessed on: 23 jun. 2020.

BRUNEO, D.; DISTEFANO, S.; GIACOBBE, M.; MINNOLO, A. L.; LONGO, F.; MERLINO, G.; MUL-

FARI, D.; PANARELLO, A.; PATANÈ, G.; PULIAFITO, A.; PULIAFITO, C.; TAPAS, N.; An IoT service ecosystem for Smart Cities: The #SmartME project. **Internet of Things**, Amsterdã, v. 5, 2019, p. 12–33, mar. 2019. Available at: https://www.sciencedirect.com/science/article/pii/S2542660518301008. Accessed on: 23 jun. 2020.

ELIAS, Alexandre; SILVA, Jefferson C. P.; GONÇALVES, Rafael N.; SOUZA, Thiago; Ardweather: uma estação meteorológica baseada no arduino e em web services restful. *In*: **XIV SAFETY, HEALTH AND ENVIRONMENT WORLD CONGRESS**, 2014, Cubatão. DOI: 10.14684/shewc.14.2014.44-48. Available at: https://www.researchgate.net/publication/281525465_ARDWEATHER_UMA_ESTACAO_METEOROLOGICA_BASEADA_NO_ARDUINO_E_EM_WEB_SERVICES_RESTFUL. Accessed on: 23 jun. 020.

INMET. **Instituto Nacional de Meteorologia**. Available at: www.inmet.gov.br/portal/index.php?r=clima/graficosClimaticos. Accessed on: 23 jun. 2020.

JAZIZADEH, F. S.M.ASCE; GHAHRAMANI, Ali; BECERIK-GERBER, Burcin A.M.ASCE; Human-building interaction framework for personalized thermal comfort-driven systems in office buildings. **Journal of Computing in Civil Engineering**, v. 28, n. 1, p. 2-16, 2014.

KALOUSTIAN, N.; BECHTEL, B. Local climatic zoning and urban heat island in Beirut. **Procedia Engineering**, v. 169, p. 216–223, 2016.

KUSRIYANTO, M.; PUTRA, A. A. **Weather Station Design Using IoT Platform Based On Arduino Mega.** *In*: INTERNATIONAL SYMPOSIUM ON ELECTRONICS AND SMART DEVICES (ISESD), 2018, Bandung. Doi: 10.1109/ISESD.2018.8605456. Available at: https://ieeexplore.ieee.org/document/8605456. Accessed on: 23 jun. 2020.

LAMBERTS, R.; DUTRA, L.; PEREIRA, F. O. R. **Eficiência energética na arquitetura**. 2014, São Paulo: Pro-Livros, 382p.

LOPES, R. **A cidade intencional: o planejamento estratégico de cidades**. 1998, Rio de Janeiro: Mauad Editora Ltda., 184p.

MONTEIRO, V. S. **Zonas Climáticas Locais e a relação com a morfologia urbana.** Estudo de caso: Campinas/SP. 2018. Dissertação (Programa de Pós-Graduação em Sistemas de Infraestrutura Urbana) – Pontifícia Universidade Católica de Campinas, 164p.

MÜLLER, C. G. **Confortímetro Lotus: Um Sistema Móvel de Baixo Consumo.** Orientador: Anderson Priebe Ferrugem. 2019. Trabalho de Conclusão de Curso (Engenharia de Computação) – Centro de Desenvolvimento Tecnológico, Universidade Federal de Pelotas, 57p.

ROMERO, M. A. B.; BAPTISTA, G. M. M.; LIMA, E. A.; WERNECK, D. R.; VIANNA, E. O.; SALES, G. L.; **Mudanças climáticas e ilhas de calor urbanas**. 2019, 1. ed. Brasília: Universidade de Brasília, 151p. Available at: http://repositorio.unb.br/handle/10482/34661. Accessed on: 6 de apr. 2020.

ROMERO, M. A. B.; LIMA, E. A.; WERNECK, D. R.; PAZOS, V.; Instrumentação para medições na escala microclimática: uma proposta de Mochila Bioclimática. *In*: **Paranoá: Cadernos de Arquitetura e Urbanismo**, n. 26, p. 96–105, maio 2020. Available at: https://periodicos.unb.br/index.php/paranoa/article/view/29544. Accessed on: 6 apr. 2020.

ROMERO, M. A. B. **Arquitetura do Lugar: Uma visão Bioclimática da Sustentabilidade em Brasília**. 2011, São Paulo: Nova Técnica Editorial.

ROMERO, M. B.; VAVALLO, H. M. O microclima criado por espelhos d'água: estudo de caso do espelho d'água do Congresso Nacional. *In*: **Paranoá: Cadernos de Arquitetura e Urbanismo**, v. 14, n. 14, p. 9–17, ago. 2015.

SAI, K. B. K.; MUKHERJEE, S.; SULTANA, H. P. Low Cost IoT Based Air Quality Monitoring Setup Using Arduino and MQ Series Sensors With Dataset Analysis. *In*: **Procedia Computer Science**, v. 165, p. 322-327, 2019. ISSN 1877-0509. DOI: https://doi.org/10.1016/j.procs.2020.01.043. Available at: http://www.sciencedirect.com/science/article/pii/S187705092030051X. Accessed on: 23 jun. 2020.

SALESFORCE. **Salesforce PAAS**. Disponível em: https://www.salesforce.com/ca/paas/. Accessed on: 23 jun. 2020.

SALLES, L. M. M**. Desenvolvimento de um dispositivo de medição de ruído com base no sistema OpenWrt**. Dissertação (Mestrado em Engenharia Mecânica) Universidade Estadual Paulista "Júlio de Mesquita Filho", Ilha Solteira, 83p, 2017.

SANTOS, F; OLIVEIRA, A. S. de; NOGUEIRA, M. C. de J. A.; MUSIS, C. R. De; NOGUEIRA, José de S.; Análise do clima urbano de Cuiabá-MT-Brasil por meio de transectos móveis. *In*: **Paranoá: cadernos de arquitetura e urbanismo**, n. 11, p. 45–54, 2014. DOI: 10.18830/issn.1679-0944.n11.2014.12083.

SCHMIDT, L. da R.; FREITAS, F. A. L. M.; MALDANER, S. Meteorological Monitoring Systems employing Arduino Platform. *In*: **Ciência e Natura**, v. 42, p. 36, 2020.

SIMÕES, N. A. V. **Classificação do clima local de sítios urbanos de Feira de Santana**. Dissertação (Mestrado em Computação Aplicada) – Universidade Estadual de Feira de Santana, 114p, 2017.

THORSSON, S.; LINDBERG, F.; ELIASSON, I.; HOLMER, B. Different methods for estimating the mean radiant temperature in an outdoor urban setting. *In*: **International Journal of Climatology**, v.27, n.14, p.1983–1993, 2007.

TOLENTINO, G. C.; TSUKAMOTO, D. B.; NOMURA, S. Estudo de caso: Utilização do Arduino para um Sistema de Controle remoto de dispositivos via internet. *In*: **XI CONFERÊNCIA DE ESTUDOS EM ENGENHARIA ELÉTRICA**, 2013, Uberlândia.

TORRES, J. D.; MONTEIRO, I. O.; DOS SANTOS, J. R.; & ORTIZ, M. S. Aquisição de dados meteorológicos através da plataforma Arduino: construção de baixo custo e análise de dados. *In*: **Scientia Plena**, v. 11, n. 2, 2015.

TRAILHEAD Salesforce. **Saleforce.** Available at: https://trailhead.salesforce.com/en/home. Accessed on: 23 jun. 2020.

ÜÇGÜN. H.; KAPLAN, Z. K. 2017. Arduino based weather forecasting station. *In*: **INTERNATIONAL CONFERENCE ON COMPUTER SCIENCE AND ENGINEERING** (UBMK), 2017, Antalya. DOI: 10.1109/UBMK.2017.8093397. Available at: https://ieeexplore.ieee.org/document/8093397/citations#citations. Accessed on: 20 jun. 2020.

VIEIRA, M. E. Utilização do sensor PT100 no Arduino para captação da TMR. *In*: **XXIX CONGRESSO DE INICIAÇÃO CIENTÍFICA** – UFPel, 2017, Pelotas.

VIEIRA, M. E. **Estudo e desenvolvimento do confortímetro lotus na plataforma arduino**. Trabalho de Conclusão de Curso (Bacharelado em Ciência da Computação) – Centro de Desenvolvimento Tecnológico, Universidade Federal de Pelotas, 61p, 2018.

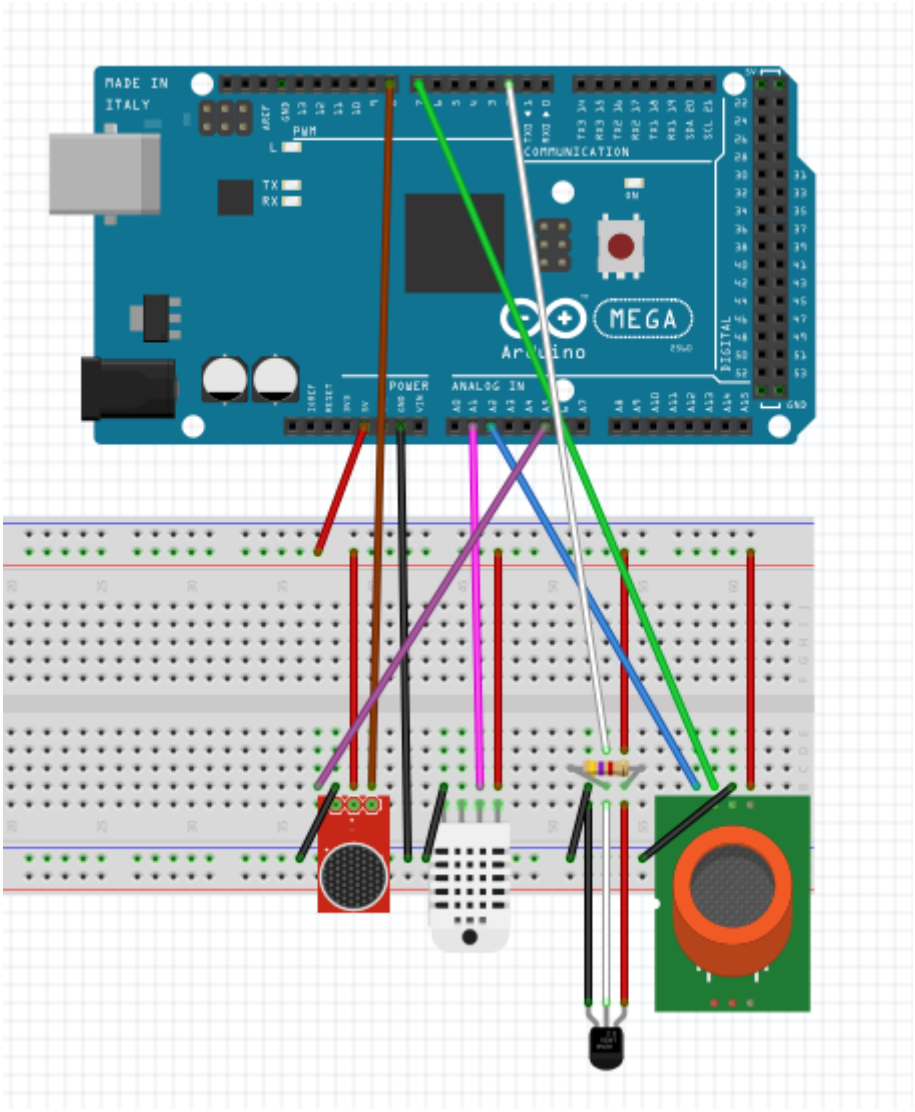
WEATHER Spark. **Condições meteorológicas médias de Campinas e Região**. Available at: https://pt.weatherspark.com/y/30197/Clima-caracter%C3%ADstico-em-Campinas-e-Regi%C3%A3o-Brasil-durante-o-ano. Accessed on: 13 jul. 2020.

W. Heldens, U. Heiden, T. Esch, A. Mueller and S. Dech, Suitability of remote sensing based surface information for a three-dimensional urban microclimate model. 2016. *In*: **IEEE INTERNATIONAL GEOSCIENCE AND REMOTE SENSING SYMPOSIUM** (IGARSS), 2016, Beijing, pp. 7322-7325. DOI: 10.1109/IGARSS.2016.7730910.

WICKRAMASINGHE, D. K. **Salesforce NodeMCU IoT Temperature Service**. Available at: http://www.manelsoft.com/projects/nodemcu_sfdc_temperature_service.aspx. Accessed on: 20 jun. 2020.

ZHAO, Y.; UDUKU, O.; MURRAY-RUST, D. EdenApp Thermal Comfort: A mobile app for measuring personal thermal comfort. In: **PASSIVE LOW ENERGY ARCHITECTURE** (PLEA), 2017 PROCEEDINGS – DESIGN TO THRIVE, 2017, Edinburgh.

APPENDIX A – PROTOTYPE ASSEMBLY, DESIGNED WITH FRITZING



Source: Authors' collection, 2020.

APPENDIX B – Arduino Code with Sensors used on the experiment

```
#include <DallasTemperature.h>
#include <OneWire.h>
#include <TimeLib.h>
#include "DHT.h"
#include <Wire.h>
#include <math.h>

#define TIME_HEADER    "T"      // Header tag for serial time sync message
#define TIME_REQUEST    7       // ASCII bell character requests a time sync message
#define MQ_analog A2    //analog pin
#define MQ_dig 7           //digital pin
#define DHTPIN A1 // connected pin
#define DHTTYPE DHT11 // DHT 11
DHT dht(DHTPIN, DHTTYPE);

//ds18b20
#define ONE_WIRE_BUS 2
OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature sensors(&oneWire);

int valor_analog;
int valor_dig;

int pino_somD = 8; //digital
int pino_somA = A5; //analog
int ReadSound = 0; //for analog reading
int dB = 0;

void setup()    {
   Serial.begin(9600);
   dht.begin();
   pinMode(13, OUTPUT);

//---MQ135---
   pinMode(MQ_analog, INPUT);
   pinMode(MQ_dig, INPUT);
```

```
//---DS18B20---
   sensors.begin();
}

void loop(){
   digitalClockDisplay();    //Time

//MQ---
     valor_analog = analogRead(MQ_analog);
     valor_dig = digitalRead(MQ_dig);

     Serial.print("QualiAr: ");
     Serial.print(valor_analog);
     Serial.print("Co2;");
     //Serial.print(" Digital: ");
   // Serial.print(valor_dig);
     //Serial.print("Co2;");

//--DHT11--
   float h = dht.readHumidity();
   float t = dht.readTemperature();
   Serial.print(" Umidade: ");
   Serial.print(h);
   Serial.print("%t;");
   Serial.print(" Temperatura: ");
   Serial.print(t);
   Serial.print("*C;");

//--DS18B20--
   sensors.requestTemperatures();
   Serial.print(" Globo: ");
   Serial.print(sensors.getTempCByIndex(0));
   Serial.print("*C;");

//--KY--
   float ReadSound = (analogRead(pino_somA));
   float voltage = ReadSound * (5.0 / 1023.0);
   //double dB = 20 * log10(ReadSound);
```

```
    Serial.print(" Som: ");
    Serial.print(voltage);
    Serial.print("; ");
    //Serial.print(" Som A: ");
    //Serial.print(ReadSound);
    //Serial.print("; ");

    Serial.println();
    delay(1000);
}

void digitalClockDisplay(){
    // digital clock display of the time
    Serial.print(hour());
    Serial.print(minute());
    Serial.print(second());
    Serial.print(";");
    Serial.print(" ");
}
#include <DallasTemperature.h>
#include <OneWire.h>
#include <TimeLib.h>
#include "DHT.h"
#include <Wire.h>
#include <math.h>

#define MQ_analog A2 //analog pin
#define MQ_dig 7 //digital pin
#define DHTPIN A1 //connected pin
int pino_somD = 8; //digital
int pino_somA = A5; //analog
int ReadSound = 0; //for analog reading
```

APPENDIX C – Arduino Code used with the Node MCU

```
#include <Arduino.h>
#include <ESP8266WiFi.h>
#include <ESP8266WiFiMulti.h>
#include <ESP8266HTTPClient.h>
#define USE_SERIAL Serial
```

```
ESP8266WiFiMulti WiFiMulti;
double temperatureString;
void setup() {
	USE_SERIAL.begin(115200);
	USE_SERIAL.setDebugOutput(true);
	USE_SERIAL.println();
	USE_SERIAL.println();
	USE_SERIAL.println();

	WiFiMulti.addAP("NomeDaRede", "senha");
}

float getTemperature() {
   float temp;

   delay(1000);
   temp = temp +1;
   return temp;
}

void loop() {
	// float temperature = getTemperature();
	//dtostrf(temperature, 2, 2, temperatureString);
	temperatureString = 33.0;

	USE_SERIAL.println("Reading");
	USE_SERIAL.println(temperatureString);

	if((WiFiMulti.run() == WL_CONNECTED)) {

		HTTPClient http;

		USE_SERIAL.print("[HTTP] begin...\n");
		// configure traged server and url
		http.begin("https://temperatureservicest-developer-edition.na172.force.com/TemperatureService/services/apexrest/iotservice?temperature=" + String(temperatureString), "07 19 1F BE 4B DB 60 F0 19 D2 AF F4 D5 06 E7 40 45 39 C3 E6");
		//http.begin("http://192.168.1.12/test.html"); //HTTP
```

```
        USE_SERIAL.print("[HTTP] GET...\n");
        // start connection and send HTTP header
        int httpCode = http.GET();

        // httpCode will be negative on error
        if(httpCode > 0) {
            // HTTP header has been send and Server response header has been handled
            USE_SERIAL.printf("[HTTP] GET... code: %d\n", httpCode);

            // file found at server
            if(httpCode == HTTP_CODE_OK) {
                String payload = http.getString();
                USE_SERIAL.println(payload);
            }
        } else {
            USE_SERIAL.printf("[HTTP] GET... failed, error: %s\n", http.errorToString(http-
Code).c_str());
        }
        http.end();
    }
    delay(5000);
}
```