Revista Multidisciplinar do Nordeste Mineiro, v.10, 2025
ISSN 2178-6925
DOI:10.61164/rmnm.v10i1.4031

IOT-DRIVEN OPTIMIZATION OF SELF-SUFFICIENT SOLAR ENERGY SYSTEMS THROUGH METEOROLOGICAL DATA INTEGRATION

OTIMIZAÇÃO DE SISTEMAS DE ENERGIA SOLAR AUTOSSUFICIENTES BASEADA EM IOT ATRAVÉS DA INTEGRAÇÃO DE DADOS METEOROLÓGICOS

Érico Schardong Rhor Piantkoski
Bachelor in Computer Science, Mato Grosso State University, Brazil
E-mail: erico@gmail.com
ORCID: https://orcid.org/0009-0008-6928-3818

Diógenes Antonio Marques José
Master of Computer Science, Mato Grosso State University, Brazil
E-mail: dioxfile@unemat.br
ORCID: https://orcid.org/0000-0002-9707-6022

Armando da Silva Filho
PhD in Environmental Physics, Mato Grosso State University, Brazil
E-mail: armandosf2000@unemat.br
ORCID: https://orcid.org/0009-0005-9602-4131

Recebido: 01/05/2025 – Aceito: 22/05/2025

Abstract

Over two million Brazilians lack access to electricity, as reported by the Brazilian Institute of Geography and Statistics (IBGE). Meanwhile, a significant portion of the population reliant on hydroelectric power faces recurring energy price hikes driven by water scarcity and systemic mismanagement. Decentralized solar energy systems offer a promising alternative; however, their intermittent nature due to seasonal variability can compromise reliability for off-grid users. To address this challenge, this study proposes an IoT-enabled embedded system that integrates meteorological data with a solar radiation prediction algorithm. The system dynamically correlates energy generation forecasts with consumption patterns to provide real-time recommendations for optimized usage, empowering users to adjust consumption proactively. During testing, the algorithm demonstrated strong alignment between predicted and actual solar energy generation, highlighting its potential to enhance energy resilience in off-grid scenarios.

Keywords: Solar Energy; Off-Grid Model; MicroPython; Python; Embedded Systems; IoT; Solar Radiation.

Resumo

Mais de dois milhões de pessoas não possuem acesso à energia elétrica no Brasil de acordo com o IBGE, enquanto a grande população que depende de hidrelétricas sofre constantemente com os aumentos nos preços gerados por escassez de água e má gestão. Uma alternativa para contornar

ISSN 2178-6925
DOI:10.61164/rmnm.v10i1.4031

esse problema se encontra na utilização do Modelo Off-Grid no âmbito de Energia Solar. No entanto, a ausência de energia, por conta da sazonalidade, pode ser um problema para os usuários desse tipo de modelo. Portanto, este trabalho apresenta a utilização de um algoritmo de predição de Radiação Solar implementado em um Sistema Embarcado, com base em dados meteorológicos e no contexto de Internet das Coisas – IoT. O sistema proposto infere a aquisição e gasto de energia elétrica, a fim de informar ao usuário se há necessidade de economia de energia. Os resultados da previsão de radiação, quando em comparação com a aquisição de energia, mostraram-se próximos e, consequentemente, promissores durante o período de testes.
Palavras Chaves: Energia Solar; Modelo Off-Grid; MicroPython; Python; Sistemas Embarcados; IoT; Radiação Solar.

1. Introduction

In Brazil, electricity generation is predominantly reliant on hydroelectric power plants. However, recent water crises and rising costs associated with this energy matrix have highlighted the need for alternative renewable sources. Among these, photovoltaic (PV) energy has gained significant attention due to its ability to convert solar radiation into electricity **(Boso; Gabriel; Gabriel Filho, 2015)**.

Solar radiation incident on the Earth's surface plays a fundamental role in global physical and biological processes **(Fu; Rich, 2000)**, while also representing a sustainable energy source capable of meeting human energy demands **(Ambarita, 2017)**. As a result, technological advancements — particularly the use of semiconductors for photoelectric conversion — have contributed to the growing viability of PV systems **(Suddard-Bangsund et al., 2016)**. The increasing interest in clean energy has also accelerated the adoption of solar systems, especially the *On-Grid* and *Off-Grid* configurations.

The *On-Grid* model operates in a hybrid fashion, integrating solar energy into the existing hydroelectric grid, thereby ensuring stability and avoiding the additional costs associated with battery storage. In contrast, the *Off-Grid* model functions autonomously, relying exclusively on solar radiation. This independence makes it particularly suitable for remote areas lacking access to the conventional power grid **(Boso; Gabriel; Filho, 2015)**. However, its dependence on weather conditions and vulnerability to seasonal and diurnal variations limit its reliability.

Revista Multidisciplinar do Nordeste Mineiro, v.10, 2025
ISSN 2178-6925
DOI:10.61164/rmnm.v10i1.4031

Therefore, to enhance the viability of *Off-Grid* systems, it is crucial to incorporate accurate meteorological data and solar radiation forecasting algorithms. Traditional forecasting methods include mathematical models, satellite-based data, and stochastic analyses, which support energy generation estimates. In this context, we propose an embedded system based on the **ESP8266** module, capable of:

1. Predicting direct and diffuse solar radiation using data from **INPE** and **BDMEP**;
2. Monitoring real-time energy generation and consumption through dedicated sensors;
3. Alerting users when adjustments in consumption are necessary to prevent energy shortfalls.

The methodology involved the development of a circuit equipped with two sensors (for energy acquisition and consumption), an analog-to-digital converter, and the **ESP8266** module for **IoT** communication. A **TCP** server and a web-based interface were also implemented for real-time data visualization. During testing, the algorithm's predictions were compared to actual energy generation values, demonstrating high concordance and validating the system's effectiveness.

The remainder of this article is organized as follows: **Section 2** reviews related work; **Section 3** describes the proposed system along with the implementation and experimental methodology; **Section 4** presents the results and discussion; and finally, **Section 5** concludes the study and outlines directions for future research.

2. Related Work

Raman, Li, and Fan (2019), in their study *"Generating Light from Darkness"*, propose an innovative approach to mitigating the seasonal limitations of photovoltaic energy by generating electricity through the abstraction of cold from space. Their method is based on the principle of **radiative cooling**, wherein a sky- facing surface dissipates a significant portion of thermal radiation directly into outer space. This process allows the surface to cool below ambient air temperatures, establishing a thermal gradient that can be harnessed to power a thermoelectric

ISSN 2178-6925
DOI:10.61164/rmnm.v10i1.4031

generator.

In the context of the **Internet of Things (IoT)**, **Pop-Vadean et al. (2016)** address the technological gap between the relatively stagnant evolution of battery technologies and the rapid growth of connected devices. They underscore the strategic importance of **energy harvesting (EH)** — the process of capturing energy from ambient sources such as solar radiation, wind, or mechanical motion and converting it into usable electrical energy. EH is presented as a promising alternative to conventional battery reliance, particularly for **IoT** applications where autonomy and sustainability are critical.

Spanias (2018) proposes a model for energy acquisition in remote photovoltaic systems through the integration of **Smart Monitoring Devices (SMDs)** into individual solar panels. The model incorporates machine learning techniques for forecasting cloud movement, localized irradiation sensors, optimization under varying shading conditions, and compensation mechanisms for transient inverter losses. According to Spanias, this comprehensive approach can yield up to a **10% increase in total energy capture**.

Gunjal et al. (2019) introduce a **solar tracking system** aimed at improving photovoltaic efficiency. This system leverages the **ESP8266** module to provide Wi- Fi connectivity, enabling both data transmission and application hosting. Utilizing a single-axis tracking mechanism, the system demonstrated a measurable increase in average power output. Furthermore, it includes capabilities for detecting operational inefficiencies during rainy periods and employs machine learning algorithms to predict energy generation, enabling automated and accurate optimizations.

Biophysical models for estimating solar radiation are also well-represented in the literature. **Hartley, Carlini, and Bellocchi (2005)** present **GSRad**, a modular and extensible software component designed for scientific applications. **GSRad** estimates various solar radiation-related variables, including extraterrestrial radiation, clear-sky transmissivity, and global solar radiation at ground level. Its architecture emphasizes modularity and reusability, enabling functionality expansion without recompilation — features that enhance its adaptability across different

Revista Multidisciplinar do Nordeste Mineiro, v.10, 2025
ISSN 2178-6925
DOI:10.61164/rmnm.v10i1.4031

scientific and technological environments.

Ambarita (2017) underscores the immense potential of solar energy by noting that the solar radiation reaching Earth annually is approximately **3,400,000 Exajoules**, compared to a global annual human energy consumption of about **450 Exajoules**. This means that, in just over an hour, solar radiation could theoretically satisfy humanity's energy needs for an entire year. Building on this insight, the author developed a measurement device comprising a wind sensor, a silicon pyranometer, and a temperature sensor. The collected data were compared to clear-sky radiation estimates from simulation software. Although the observed discrepancy was less than **10%** under clear-sky conditions (Figure 1), the system showed limitations in accurately predicting radiation during cloudy periods.

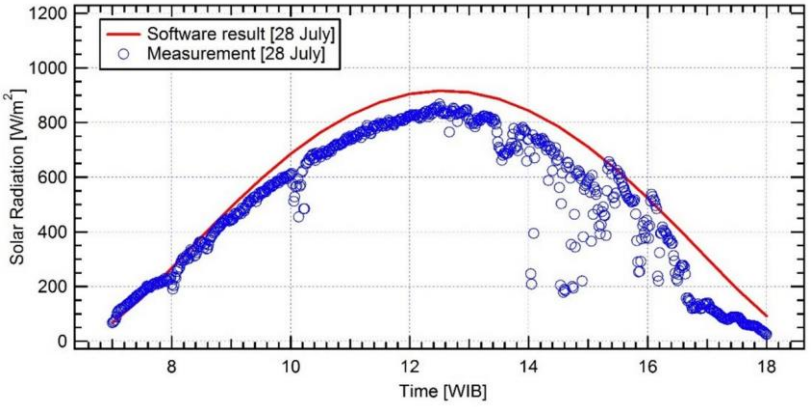

Figure 1 – Comparison between software estimates and measured values under clear-sky conditions. *Source: Ambarita (2017, p. 4).*

To address the challenge of solar radiation estimation under cloudy conditions, **Moojen, Cavalcante, and Mendes (2012)** proposed a model incorporating empirical cloud cover data, a **Digital Elevation Model (DEM)** of the Eldorado do Sul region (RS), and climatological records from a local meteorological station. Their methodology enables estimation of both direct and diffuse radiation at the surface, with terrain elevation factored as a variable. Validation with direct measurements of global radiation taken throughout 2009 confirmed the model's effectiveness. However, the use of **monthly average cloud cover** introduced inaccuracies in daily estimates. Despite this limitation, the model remains **cost-**

ISSN 2178-6925
DOI:10.61164/rmnm.v10i1.4031

effective and **reasonably accurate**, particularly when enhanced with frequent cloud cover observations, making it more reliable at monthly timescales.

3. Proposal Description

This chapter provides a detailed description of the proposed system and the steps undertaken to achieve the objectives of this study. The primary goal of this work is to enable — or at least assist — users of *Off-Grid* energy systems in conserving energy. Given that the strategies to achieve this goal rely on deterministic assumptions, the approach varies depending on the availability of stochastic models and satellite data. Accordingly, distinct methodologies are applied in scenarios with and without access to such data sources.

3.1 Empirical Associations

The initial stage of the analysis involved evaluating the user's energy generation and consumption without referencing external databases. However, this approach presents two significant limitations. First, it requires the user to accumulate data over at least one full year to build a comprehensive dataset of daily energy patterns. Second, even after this period, the resulting database would be inherently limited — valid only for that specific year — and would fail to account for potential future variations caused by climatic or environmental changes. In other words, relying solely on historical data may not provide reliable or adaptable insights. To address both issues, a more temporally constrained approach is proposed — one that compares current energy consumption and generation with the average values from the same day of the previous week. This method assumes that energy generation is influenced by direct and diffuse solar radiation measured by the meteorological station, while consumption trends can be contextualized by referencing past generation data. This comparison enables users to assess whether current usage patterns are sustainable and to anticipate the need for energy conservation.

Revista Multidisciplinar do Nordeste Mineiro, v.10, 2025
ISSN 2178-6925
DOI:10.61164/rmnm.v10i1.4031

Figure 2 (representing Day 1 of the first week) illustrates an *Off-Grid* energy acquisition model, incorporating the **IoT-based** system developed in this study. On that particular day, the user generated more energy than was consumed — otherwise, the system would have depleted its stored energy. By calculating the difference between energy generation ("Gain") and consumption ("Spent") as a percentage, the **Energy Savings Need Algorithm (ASNA)** can provide real-time guidance to users:

- **If Spent is ≥ 20% greater than Gain**: **Energy savings highly recommended;**
- **If Spent is ≥ 10% greater than Gain**: **Energy savings strongly recommended;**
- **If Spent is ≥ Gain**: **Energy savings recommended;**
- **If Spent is within ±10% of Gain**: **Acceptable energy level;**
- **If Spent is ≥ 20% less than Gain**: **Good energy level.**

This adaptive feedback mechanism empowers users to manage their energy usage proactively, based on recent trends and real-time data, rather than relying solely on long-term historical patterns that may not reflect current or future conditions.

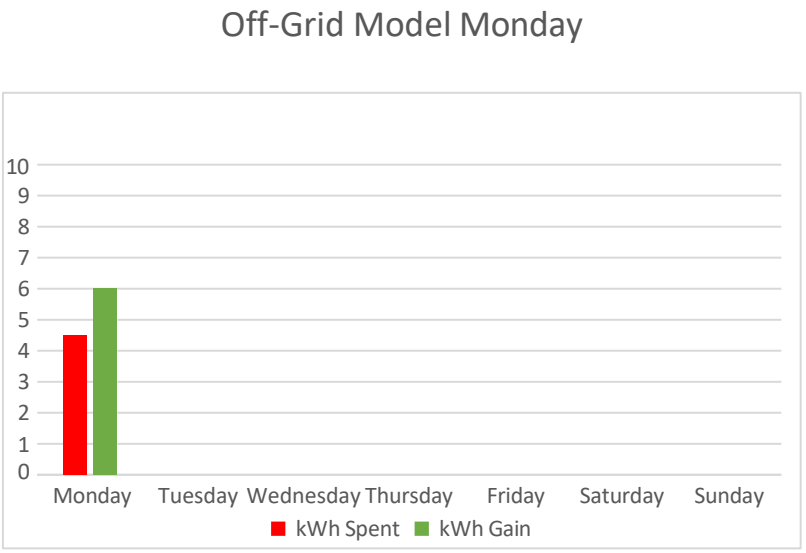

Figure 2 – Example 1 – Expense and income of day 1 of the first week of the Off-Grid acquisition model with the proposed system implemented (merely symbolic values). Source: From the author (2021).

Revista Multidisciplinar do Nordeste Mineiro, v.10, 2025
ISSN 2178-6925
DOI:10.61164/rmnm.v10i1.4031

As illustrated in Figure 2, on Monday, the system would classify the energy level as **"Good"**, given that the energy gain exceeded consumption by 25% (6 kWh generated versus 4.5 kWh consumed). However, the system's accuracy improves over time, as it aggregates data from previous days to calculate a more representative average — up to a maximum of seven days. In the example shown in Figure 2, only a single day of data is available; therefore, the system's assessment is based solely on the energy gain from that first day.

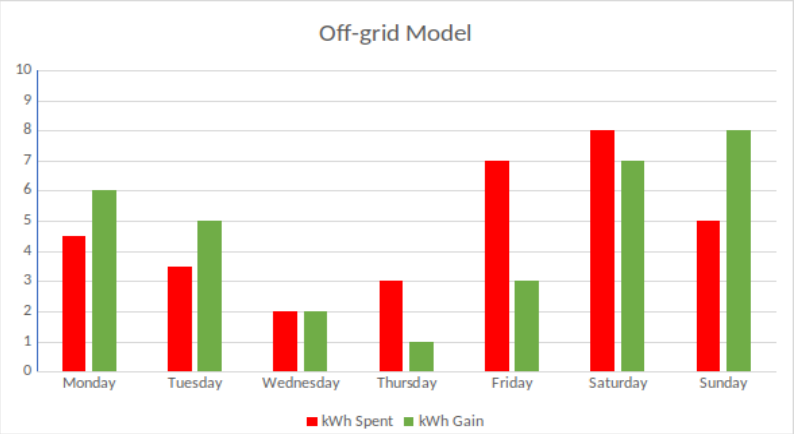

Figure 3 – Week 1 cost/gain of the Off-Grid acquisition model with the proposed system. Source: Author (2021).

To calculate the user's energy return for the day following that shown in Figure 3 — Monday of the second week — the system no longer compares only the energy gain and expenditure from the previous day (Sunday). Instead, it adopts a more robust approach by calculating the **seven-day average energy gain** (from Monday to Sunday), which in this case is approximately **4.57 kWh/m²**, and comparing it to the **three-day average energy expenditure** (from Friday to Sunday), approximately **6.67 kWh/m²**. Based on this comparison, the system would classify the energy level as **"Extremely Recommended Savings"**, as the expenditure exceeds the gain by roughly **31.46%**.

Despite this improvement in predictive capacity, the model still shows **limitations in the short term**. Specifically, it does not allow the user to track energy consumption with the same gradual sensitivity that photoperiod variations provide over longer periods. Nonetheless, although this method is less precise, it remains a viable solution in the absence of real-time solar radiation data.

Revista Multidisciplinar do Nordeste Mineiro, v.10, 2025
ISSN 2178-6925
DOI:10.61164/rmnm.v10i1.4031

3.2 Satellite and Stochastic Data

The second approach involves leveraging a **reliable solar radiation database** to enhance the system's accuracy. Two sources are utilized:

1. **National Institute for Space Research (INPE)** – Provides solar radiation values (Wh/m²) as daily monthly averages.
2. **Meteorological Database for Teaching and Research (BDMEP)** – Offers detailed data on the local **sunshine duration (photoperiod in hours)**.

By combining these datasets and applying them to formulas **(1)** and **(2)**, the system calculates a specific value of solar energy (Wh/m²) for each geographic coordinate, treating each *Off-Grid* system as **a unique energy model**. The selection of these databases was based on **scientific reliability and public accessibility**, rather than commercial factors, as both sources are free and available for research purposes upon registration.

$$imD=\sum_D \quad \textbf{(1)}$$

$$\sum_D = \frac{D_{A-1}+D_{A-2}+D_{A-3}+\ldots+D_{A-30}}{30} \quad \textbf{(2)}$$

Therefore, the first step is to estimate the efficiency[1] per m. This factor is important and should always be taken into account, as it directly affects energy acquisition. To do this, the Estimated Radiation value for a D day (ReD), equation (3), is first compared with the value acquired on the previous day, arriving at a percentage difference. The following month, this efficiency is measured in the same way, taking into account the average monthly gain, however, instead of a single day.

$$estimated\ gain=\cdot eficience \quad \textbf{(3)}$$

[1] Solar modules from different manufacturers have different performances. Knowing the efficiency is vital for this work, not only because it influences the comparison algorithm, but also so that the user can know the health of their modules from time to time. However, not all panels have a 1x1 m^2 standard. It is therefore necessary to calculate the gain per m2 beforehand based on the average gain.

ISSN 2178-6925
DOI:10.61164/rmnm.v10i1.4031

The product of the system's **efficiency (%) per square meter** and the **ReD value (Wh/m²)** — as defined in Equation **(3)** — yields the **estimated energy gain (Wh/m²)**. This value is expected to closely approximate the actual energy generated by the user. In accordance with the logic presented in **Section 3.1**, this estimated gain is then compared to the **average energy expenditure over the past three days**, as outlined by the *Energy Savings Need Algorithm (ESNA)*.

3.3 Combined Approach Using Multiple Techniques

The third approach involves a **hybrid formulation** that integrates the two previous methods to provide a more comprehensive solution, particularly during periods when database information is **incomplete, inconsistent, or unavailable**. In this approach, the user's own data — both energy gain and consumption — is compared with the data obtained from the external databases. The system then interprets this comparison to generate a recommendation for the **expected level of energy savings**.

To implement this approach effectively, three components are essential:

1. A **data acquisition mechanism** to collect the user's real-time energy gain and consumption data;
2. A **processing environment** capable of analyzing the external database values and comparing them to the user's energy data; and
3. A **user interface** that clearly communicates the results and savings recommendations in an understandable format.

This combined strategy allows for adaptive decision-making even in the absence of ideal data conditions, increasing the reliability and usability of the *Off- Grid* energy management system.

3.4 Materials and Methods

3.4.1 Implementation of the Proposal in the Off-Grid Model

To establish a system for measuring energy acquisition and consumption, a **1 m² MPrime M150P – 12V solar panel** was used (Figure 4). For optimal solar exposure, the panel was installed in **Barra do Bugres-MT**, oriented **15º northward**

Revista Multidisciplinar do Nordeste Mineiro, v.10, 2025
ISSN 2178-6925
DOI:10.61164/rmnm.v10i1.4031

to maximize radiation capture. To simulate energy consumption, **three 12V, 55W lamps** were connected directly to a **battery-powered system**, representing typical *Off-Grid* usage conditions.

Figure 4 – MPrime M150P-12v. Source: From the author (2019).

In addition, we used an **ESP8266 V3 12F** board that was designed to solve the problem of capturing user acquisition and consumption data. Figure 5 exemplifies a conventional *Off-Grid* system with the proposed system, with a contact point between the photovoltaic panel and the charge controller, in order to infer the acquisition (gain), and another between the inverter and the consumption,

Revista Multidisciplinar do Nordeste Mineiro, v.10, 2025
ISSN 2178-6925
DOI:10.61164/rmnm.v10i1.4031

to infer the consumption (expense).

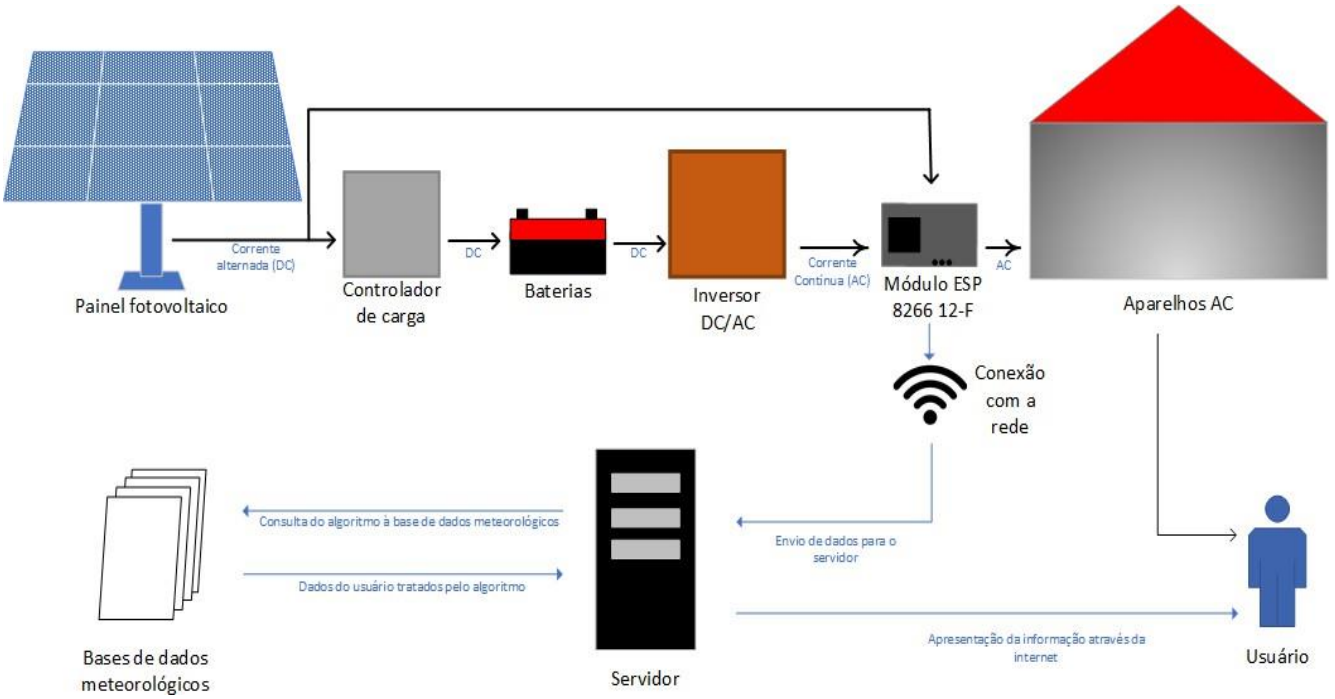

Figure 5 – Off-Grid energy acquisition model with ESP8266 implemented. Source: From the author (2020).

However, the **NodeMCU ESP8266** module is equipped with only one analog input, which limits it to a single analog reading. To overcome this limitation, the **MCP3008** microchip can be used. Operating within a voltage range of 2.7V to 5.5V, the **MCP3008** enables analog-to-digital conversion for multiple channels. To measure current in amperes (A) for internal software calculations, two **ACS712 current sensors** from Allegro were employed — one to measure current generated by the solar panel, and the other to monitor the internal power consumption of the residence.

The **ESP8266**'s ability to connect to the Internet enables the transmission of this data to a TCP server for processing, visualization, and analysis, thereby characterizing the system as an **Internet of Things (IoT)** application. In this context, the system simulates a conventional *off-grid* photovoltaic setup: electrical energy is harvested by the solar panel, regulated by a charge controller, and stored in batteries. The stored **Direct Current (DC)** is typically converted into **Alternating Current (AC)** via an inverter for household use. However, for experimental

Revista Multidisciplinar do Nordeste Mineiro, v.10, 2025
ISSN 2178-6925
DOI:10.61164/rmnm.v10i1.4031

purposes, a simplified *off-grid* system operating solely with DC power — without the use of an inverter — was implemented (Figure 6).

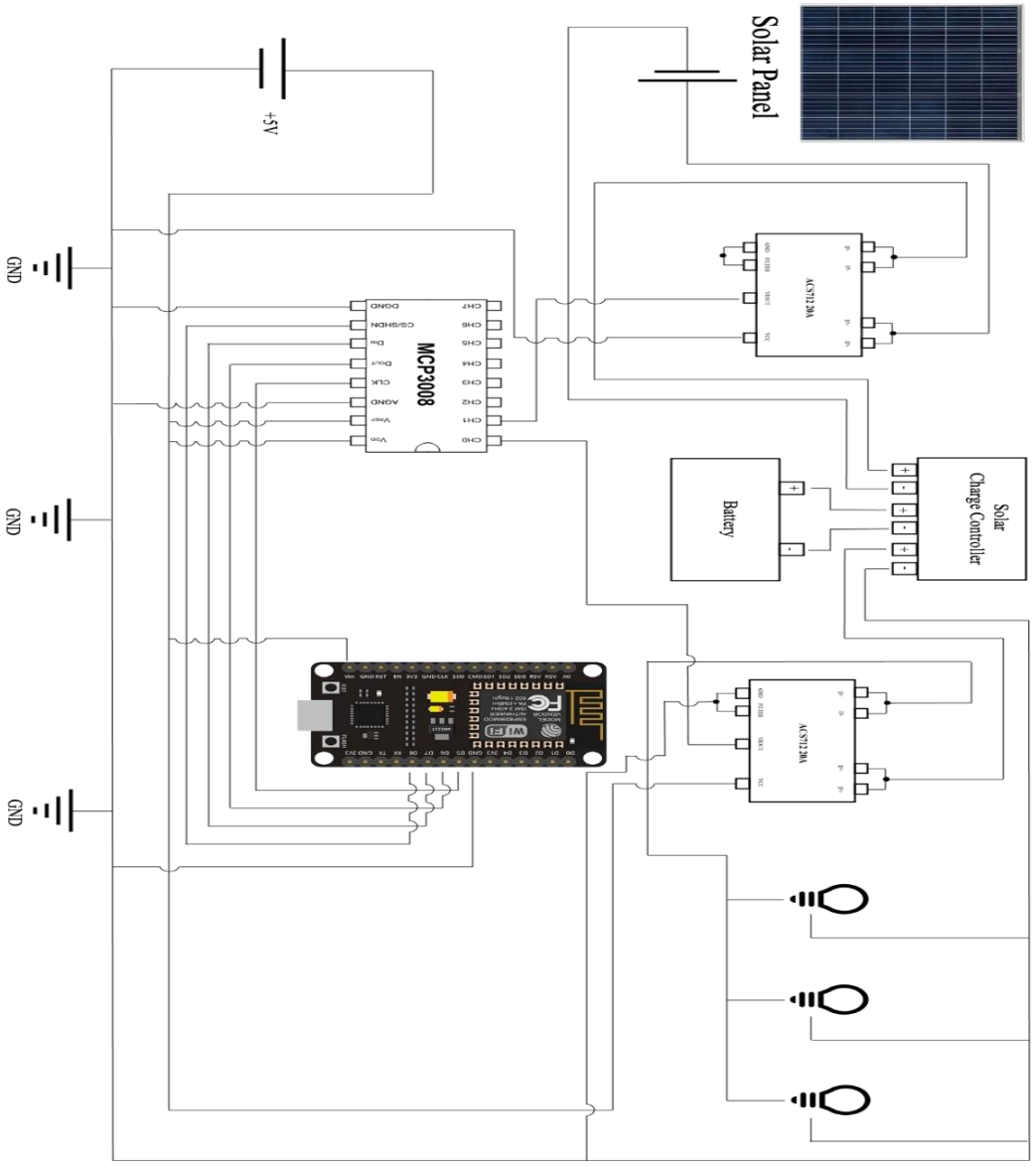

Figure 6 – Circuit diagram. Source: By the author (2021).

3.4.2 Development

As an alternative approach, the data was made accessible via the Internet through a web-based interface. One effective method to achieve this is by using the

Revista Multidisciplinar do Nordeste Mineiro, v.10, 2025
ISSN 2178-6925
DOI:10.61164/rmnm.v10i1.4031

ESP8266 development board, which was selected primarily for its flexibility and ease of integration, and secondarily for its built-in Wi-Fi capability. With this setup, the data flow — illustrated in Figure 7 — begins at the load, where the relevant data is initially generated and made available for transmission.

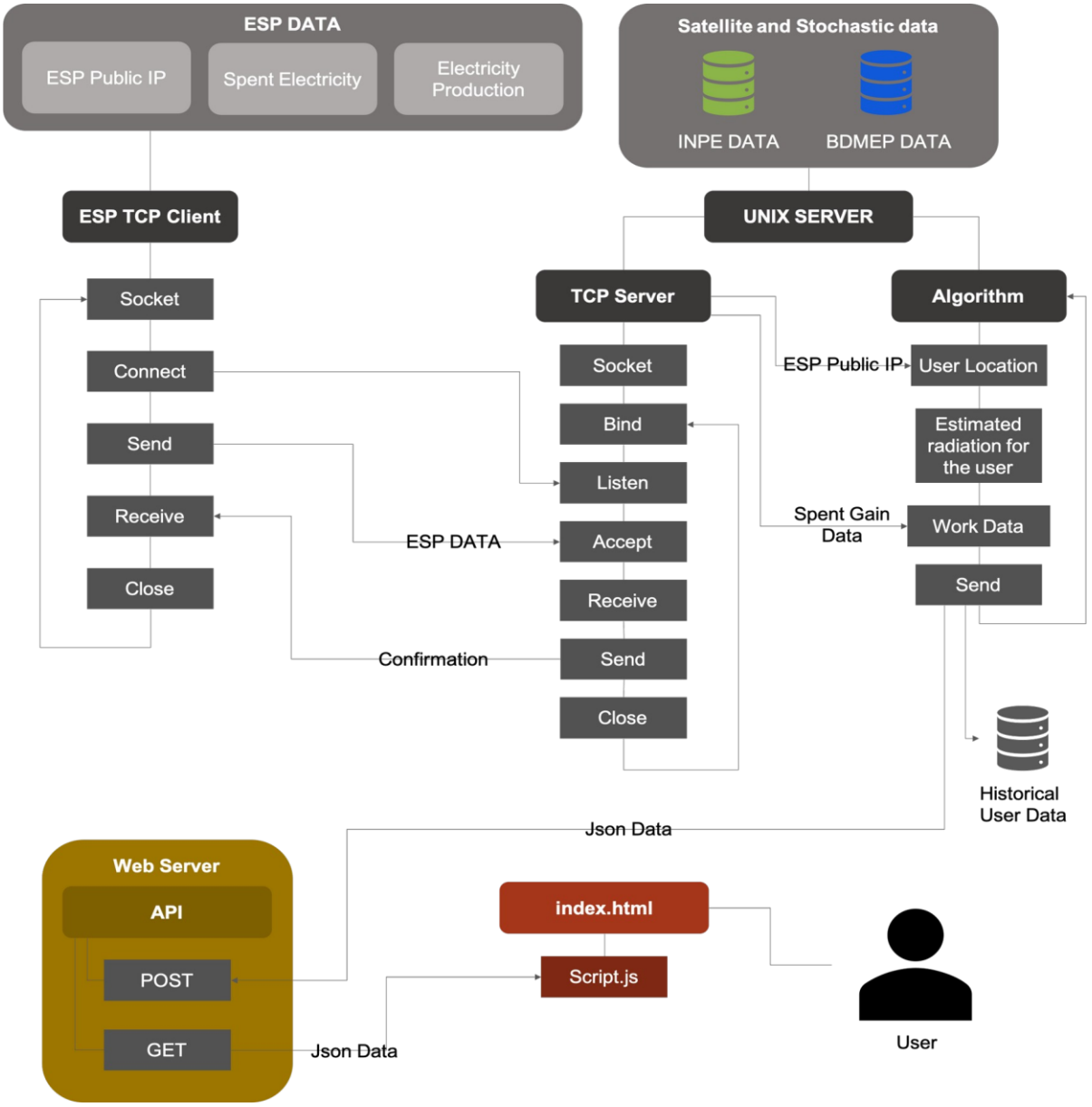

Figure 7 – Data flow. Source: Author (2021).

3.4.2.1 Development Environment

Revista Multidisciplinar do Nordeste Mineiro, v.10, 2025
ISSN 2178-6925
DOI:10.61164/rmnm.v10i1.4031

For this project, the **Visual Studio Code Integrated Development Environment (IDE)**, version 1.41.1, was chosen to edit all scripts and the main application running on both the server and client sides **(VISUAL STUDIO CODE, 2020)**. Although several other **IDEs** could have been considered, *Visual Studio Code* was selected due to its ease of use, suitability for the project's requirements, and strong community support, which provides extensive documentation, examples, and troubleshooting resources. Additionally, its cross-platform availability ensures compatibility with multiple operating systems.

Another tool used during development was **Screen**, a terminal multiplexer that enables the execution of remote shell sessions via **SSH**. Screen is natively available on both **macOS** and Linux systems. In practical terms, its function in this project was similar to that of an **IDE** for *Python* — allowing the execution of **MicroPython** commands directly on the **ESP8266**.

3.4.2.2 Data Acquisition Using the ESP8266

As illustrated in Figure 7, the **ESP8266** microcontroller is responsible for transmitting data related to solar energy generation, electrical energy consumption, and the device's public IP address. The IP information can be used to approximate the system's geographical location — without the need for a GPS module — by employing geolocation services such as the ***geocoder*** library on the TCP server side.

Both solar radiation acquisition (energy input) and electrical energy consumption (load usage) are measured using two **20A ACS712** current sensors from Allegro, as previously described in Section 3.4.1. These sensors are interfaced with an **MCP3008** analog-to-digital converter (ADC), which allows the **ESP8266** to process multiple analog inputs. Thus, to accurately interpret the readings from the **ACS712** sensors, the **MCP3008** must be properly calibrated. For this purpose, a custom driver was developed to handle the ADC readings, convert the sensor output into meaningful current values, and forward the processed data through a designated port to the **ESP8266**.

Revista Multidisciplinar do Nordeste Mineiro, v.10, 2025
ISSN 2178-6925
DOI:10.61164/rmnm.v10i1.4031

The **ACS712** 20A model measures currents in the range of **-20A to +20A**. It outputs a voltage of approximately 0.5V for -20A, 2.5V for 0A (no current), and 4.5V for +20A. The **MCP3008** converts these analog voltages into digital values ranging from 0 to 1023, corresponding linearly to an input voltage range of 0V to 5V [2]. Accordingly, the **MCP3008** driver includes a calibration function that translates the ADC values into current readings based on Equation (4).

$$Res = \frac{ACSout - 511{,}5}{20{,}46} \quad \textbf{(4)}$$

The MCP3008 returns to the **ESP8266** a **resolution value (Res)**, expressed in amperes, which is calculated based on the input voltage value from the **ACS712** sensor (**ACS_{out}**). This equation accounts for the raw digital value returned by the **MCP3008**. Since a value of approximately **511.5** corresponds to **0A** (i.e., 2.5V output from the **ACS712**), the theoretical range spans from around **102.3** (representing -20A, or ~0.5V) to **920.7** (representing +20A, or ~4.5V). However, testing revealed that the values returned by the **ACS712** can occasionally exceed the expected voltage range. While the **MCP3008**'s digital resolution remains consistent, any readings corresponding to voltages above 5V or below 0V will be lost, as they fall outside the ADC's measurable limits. These cases result in clipping at the extremes of the resolution range.

The resolution calculation is applied identically to the readings from both current sensors. The denominator in the calibration formula, **20.46**, is a correction factor derived from the volts-per-ampere sensitivity of the **ACS712**, based on Equation (5). In that equation, **0.1** represents the sensor's sensitivity constant (V/A), and **Vcc** is the power supply voltage used to determine the **MCP3008**'s resolution.

$$\frac{0{,}1}{\frac{Vcc}{1024 - 1}} \quad \textbf{(5)}$$

[2] It depends on how the MCP3008 is powered. Therefore, this work addresses the microchip's power supply with 5V, so the resolution is justified. In the case of a 3.3V power supply, for example, the resolution range would change to 0 to 3.3V.

Revista Multidisciplinar do Nordeste Mineiro, v.10, 2025
ISSN 2178-6925
DOI:10.61164/rmnm.v10i1.4031

3.4.2.3 Data Flow Between the ESP8266 and the TCP Server

The ACS712 sensors perform readings every second, and the data is temporarily stored in the ESP8266's cache until it is aggregated into an hourly average. This hourly data is recorded in the format: **[consumption (A), generation (A), timestamp (dd/mm/yyyy hh:mm)].**

Therefore, to ensure the **ESP8266** maintains accurate internal time, it periodically synchronizes with the **worldtimeapi API**. This is necessary because, upon each reset, the **ESP8266** defaults to Unix Epoch time. As a result, when midnight is reached, a new text file is created (or updated, if it already exists) on the **ESP8266**, containing the daily average in the format: **[consumption (A), generation (A), date (dd/mm/yyyy)].** Thus, if no hourly records are available for that day, the file is not generated.

The main program (main.py) running on the **ESP8266** is written asynchronously. In addition to creating a clock-like object that manages time synchronization and a task that performs second-by-second data collection, the code also defines a **TCP client** (as illustrated in Figure 7). This client continuously attempts to establish a connection with a Unix-based TCP server using the connect() function from Python's socket library. The connection is made using a tuple **(IP, port)**, where the first element is the server's IP address (as a string), and the second is the port number (as an integer). Besides that, once the handshake is successfully completed and the devices are connected, the **ESP8266** retrieves its public IP address using the **icanhazip.com** service. It then transmits this IP to the TCP server, which uses it to estimate the **ESP8266**'s geographic coordinates (latitude and longitude). After this exchange, the **ESP8266** closes the socket and asynchronously waits for 10 seconds before resuming operations.

On the **server side**, once the client's geographic coordinates are obtained, the system begins generating a solar radiation forecast. This forecast spans a one- week window — covering four days prior, the current day, and two days ahead. Meanwhile, after the 10-second asynchronous delay, the **ESP8266** checks for the existence of a daily average **.txt** file. If the file is found, the device attempts to

Revista Multidisciplinar do Nordeste Mineiro, v.10, 2025
ISSN 2178-6925
DOI:10.61164/rmnm.v10i1.4031

reconnect to the TCP server. Upon successful handshake, it transmits the data — whether it contains a single day's record or multiple entries — then closes the socket, deletes the **.txt** file, and enters a 24-hour wait period before initiating a new connection. If no daily average file is found, the **ESP8266** simply waits, performs another check, and repeats the process. Consequently, each day, the **ESP8266** also attempts to send its public IP address, which the server uses to initiate a new radiation forecast specific to the device's location. On the **TCP server**, upon receiving daily average records, the first action is to store the data in a **.csv** file for empirical logging and analysis. The server is designed to support multiple simultaneous connections, ensuring scalability. It distinguishes between different clients using unique IDs assigned to each **ESP8266** device.

3.4.2.3 Database Selection Guidelines

This section outlines the criteria used for selecting and implementing the databases, specifically the **INPE** and **BDMEP** sources, and explains how each was integrated into the system. The **BDMEP** database organizes its meteorological stations by Brazilian state, as illustrated in Figure 8. An inspection of the webpage's underlying code reveals that states are identified using standard two-letter acronyms (e.g., SP for São Paulo, PA for Pará). In the data structure used, the state information is typically found at index 3 in the format: **[latitude, longitude, city, state]**, where the final element corresponds to the state's acronym.

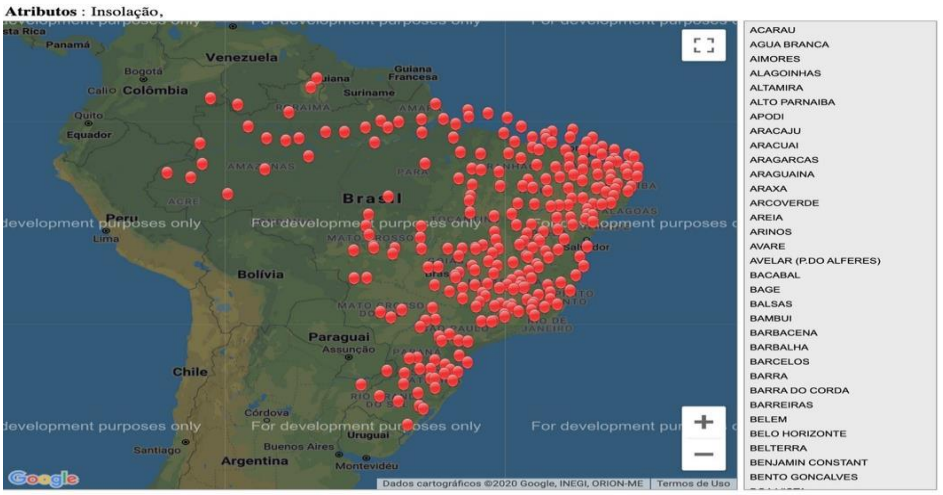

Figure 8 – Map of BDMEP meteorological stations. Source: (INMET, 2020).

Revista Multidisciplinar do Nordeste Mineiro, v.10, 2025
ISSN 2178-6925
DOI:10.61164/rmnm.v10i1.4031

Based on the considerations above, three metrics were established to determine how to proceed using the user's location data. The **first metric** attempts to identify whether a **BDMEP** station exists in the user's city. This is done by comparing the city name at **index 2** of the station data list. However, due to the limited number of **BDMEP** stations compared to the vast number of cities in Brazil, this check often fails. In such cases, the **second metric** becomes more appropriate. It involves calculating the geographical distance between the user's coordinates and the stations using trigonometric formulas. This is based on the latitude and longitude values found at **indexes 0 and 1** of the station list. The result is a vector map (illustrated in Figure 9) showing the relative distances between the user and nearby stations. To optimize performance, the algorithm uses the **state abbreviation** from the user's location to restrict the search to only the **BDMEP** stations within that state. It then identifies the station corresponding to the **shortest distance vector** (measured in kilometers) and selects it as the closest match. It is important to note that this approach is only feasible because **each Brazilian state has at least one BDMEP station**, and the algorithm is designed to operate **within the bounds of the specified state**.

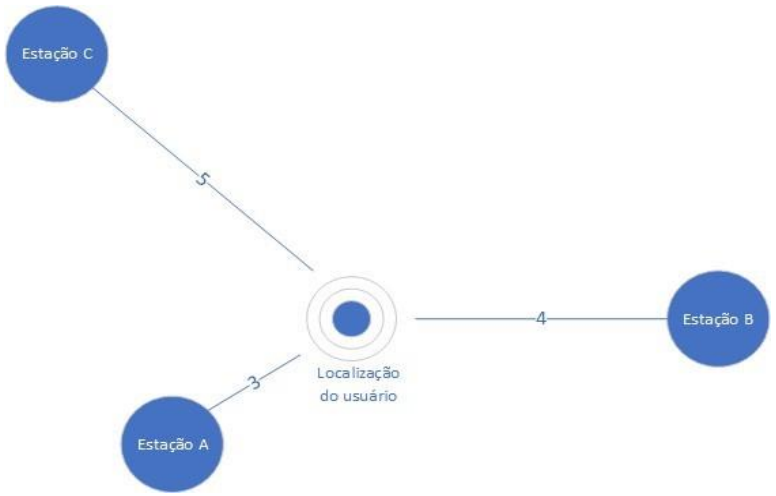

Figure 9 – Example of vector creation based on the distance between the user and the BDMEP databases. Source: Author (2020).

However, there may be cases where no station within the user's state provides consistent or sufficient data. In such situations, the algorithm expands its search to include **all stations** returned from a direct query to the **BDMEP** website. If

ISSN 2178-6925
DOI:10.61164/rmnm.v10i1.4031

the **closest station** does not contain at least **30% usable data**, the algorithm proceeds to the **next closest station**, based on ascending vector distance, and continues this process until a suitable station is found or no more options remain. If **none** of the stations in the state have reliable data, the system resorts to the **third metric**, known as **Empirical Associations**. For example, in this study, the user's location includes the city **Barra do Bugres (MT)** as its third list item. The closest **BDMEP** station identified was **Diamantino (MT)**; however, due to insufficient data availability, the system selected the **next closest station**, **Cáceres (MT)**, whose data met the consistency threshold.

The criteria for selecting **INPE** satellite data points differ from those used for **BDMEP** stations. This is primarily because **INPE** provides **multiple satellite- derived data points**, and the analysis involves **two types of solar radiation**: **Direct** and **Diffuse**. Both the **BDMEP** and **INPE** datasets were manually downloaded and added to the system as preloaded resources. This was necessary because:

- **BDMEP** has restricted web scraping access by updating its data access protocols.
- The **Brazilian Solar Energy Atlas – 2nd Edition** (from **INPE**) provides only empirical average values derived from satellite observations and does not offer real-time updates.

As a result, **INPE data must be downloaded in advance** before applying the corresponding selection metrics. Conversely, **BDMEP** data are **updated daily** at their respective stations. Therefore, any future updates to the system must account for and accommodate the dynamic nature of **BDMEP** data. Furthermore, **LABREN** utilizes **Direct Radiation** and **Diffuse Radiation** data files. To align these datasets with the user's location, the geographical coordinates (latitude and longitude) provided in indexes 0 and 1 of the user's input were used to identify the **four closest spatial points** within both radiation files. These points were selected based on a maximum allowable deviation of **0.1º** from the user's specified latitude and longitude. Once identified, a spatial interpolation method was applied: the values

ISSN 2178-6925
DOI:10.61164/rmnm.v10i1.4031

from the four points in each file (Direct and Diffuse) were averaged to generate **two consolidated radiation values** (one for each dataset). These values were then summed (Direct + Diffuse) to produce a **total solar irradiance estimate** for the user's location.

Subsequent data processing steps — including statistical averaging and validation — were performed to refine the results. The final output is designed to be compared against the user's power grid data, after which the processed information is transmitted via the **API** for further analysis or system integration.

3.4.2.4 Data Reliability and Validation

Challenge → The **BDMEP dataset** occasionally contains gaps or records with **zero photoperiod values**, rendering them unusable for calculating the **Daily Average Insolation (ImD)**. Such inconsistencies can compromise the reliability of solar energy estimates.

Solution Workflow → To address this, the data processing module's main class incorporates error-handling logic structured as follows:

1. **Apply BDMEP Guidelines**: Validate dataset compliance with predefined standards;
2. **Acquire Data**: Retrieve records from the selected **BDMEP** station (e.g., the closest station, *Cáceres*);
3. **Filter Invalid Entries**: Discard data points with null or zero photoperiod values;
4. **Compute Average Insolation**: Apply Equation (1) to calculate **ImD** for all valid days of the year.

Validation Protocol: While the automated filtering and calculation processes yield statistically sound results, manual verification is critical to ensure **code robustness** and prevent logical errors. To facilitate this:

- An **interim code component** generates a .csv file during the **ImD** calculation phase;
- This file aggregates daily station records into monthly lists, aligning with the

Revista Multidisciplinar do Nordeste Mineiro, v.10, 2025
ISSN 2178-6925
DOI:10.61164/rmnm.v10i1.4031

code's data treatment workflow (see **Figure 10** for implementation details);

- Validation is performed by comparing code-generated averages against manual Excel-based calculations. Consistency between the two confirms the algorithm's accuracy.

Rationale for Implementation: The .csv generation step was embedded within the **ImD** calculation method to streamline data segmentation by month, simplifying manual review and ensuring alignment with the station's geographic prioritization logic.

For instance, the Cáceres station dataset spans **8,866 daily records** from March 25, 1990, to March 25, 2021. Following the **data filtering process** (removal of entries with invalid or zero photoperiod values), **7,349 valid records** remained, covering the period from March 25, 1990, to July 2, 2014. To validate the integrity of this filtering, the total number of retained records was cross-referenced against the **automatically generated** .csv file, which confirmed the same count of 7,349 entries. This file is **preserved in its original state** to ensure consistency and reliability during comparative analyses, minimizing the risk of discrepancies in validation workflows.

```
187     with open('intdata.csv', 'w') as csvfile:
188         spamwriter = csv.writer(csvfile, delimiter=';', quoting=csv.QUOTE_MINIMAL)
189         spamwriter.writerow(['Photoperiod', 'Day', 'Month', 'Year'])
190         for item in jan:
191             spamwriter.writerow(item)
192         for item in feb:
193             spamwriter.writerow(item)
194         for item in mar:
195             spamwriter.writerow(item)
196         for item in apr:
197             spamwriter.writerow(item)
198         for item in may:
199             spamwriter.writerow(item)
200         for item in jun:
201             spamwriter.writerow(item)
202         for item in jul:
203             spamwriter.writerow(item)
204         for item in aug:
205             spamwriter.writerow(item)
206         for item in sep:
207             spamwriter.writerow(item)
208         for item in oct:
209             spamwriter.writerow(item)
210         for item in nov:
211             spamwriter.writerow(item)
212         for item in dec:
213             spamwriter.writerow(item)
214
```

Figure 10 – Code that creates the spreadsheet for all records of the BDMEP station in Cáceres. Source: By the author (2020).

Revista Multidisciplinar do Nordeste Mineiro, v.10, 2025
ISSN 2178-6925
DOI:10.61164/rmnm.v10i1.4031

To validate the algorithm's accuracy, **six randomly selected days** were analyzed. For each day, the average insolation was calculated independently using two methods:

1. **Manual calculation** via the .csv file (processed in Excel);
2. **Automated calculation** via the proposed software.

As illustrated in Figure 11, the results demonstrated **near-identical precision** between the manual and automated methods, confirming the algorithm's reliability. Thus, following this validation, the **BDMEP and INPE datasets were integrated** to estimate daily solar radiation. Using Python's datetime library, the system generates an **Estimated Radiation** value (in Wh/m²) for the user's specified location, centered on the current date. This includes:

- The current day;
- Five preceding days;
- Two subsequent days.

Therefore, this temporal scope enables a 9-day radiation profile, enhancing analytical flexibility for real-time and short-term solar energy forecasting.

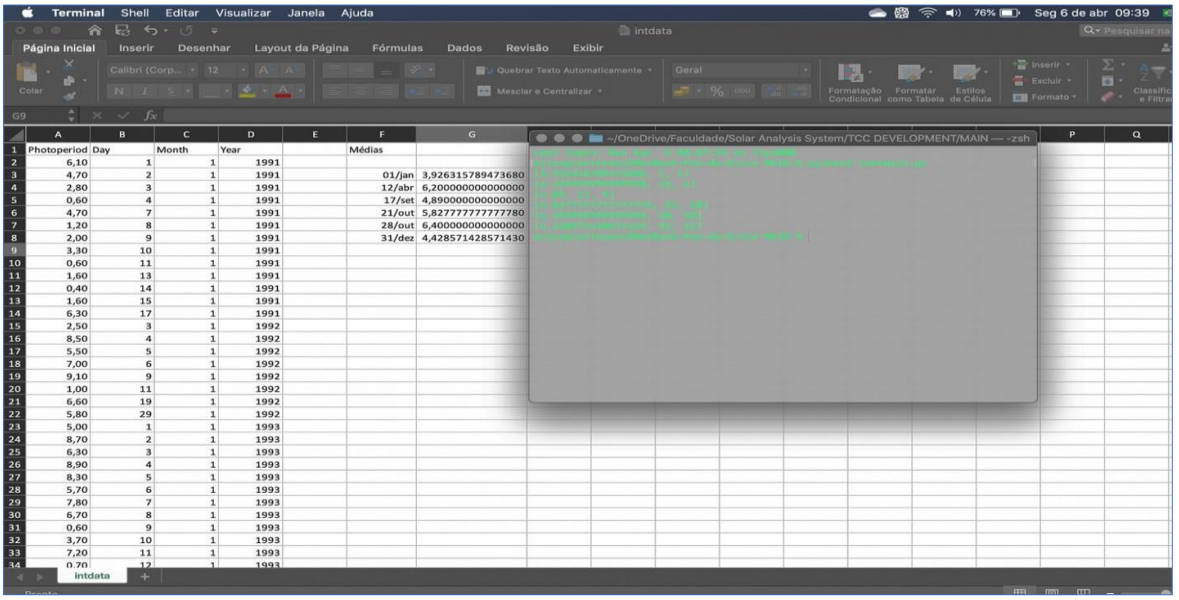

Figure 11 – Comparison of manual and automatic calculations of average photoperiod for certain days of the year. Source: Author (2020).

3.4.2.5 Presenting Information to the User

Revista Multidisciplinar do Nordeste Mineiro, v.10, 2025
ISSN 2178-6925
DOI:10.61164/rmnm.v10i1.4031

After the TCP server creates (or updates) empirical records, it gathers the **ESP8266** data and the radiation prediction data into a data dictionary. This dictionary is converted to **JSON**, encrypted, and sent via a **POST** method to an API developed specifically for this work, which decrypts it and makes the data publicly available for consumption via the GET method. This data is used by a JavaScript script on the website, and analyzed by a comparison algorithm to apply the most convenient approach.

The API was developed in Python using the Flask library. To allow external communication (access control), the **Flask-CORS** library was used, where **CORS** is added to the header of each POST. Thus, to keep the project free, the API was hosted on **PythonAnywhere**. This created a domain for the API, which makes the HTTP methods programmed in it available. The website was hosted for free on **GitHub** Pages. Figure 12 shows the presentation of this data to the user and details of the work. At the time, the day was the 17th, so there was still no acquisition or consumption for the day.

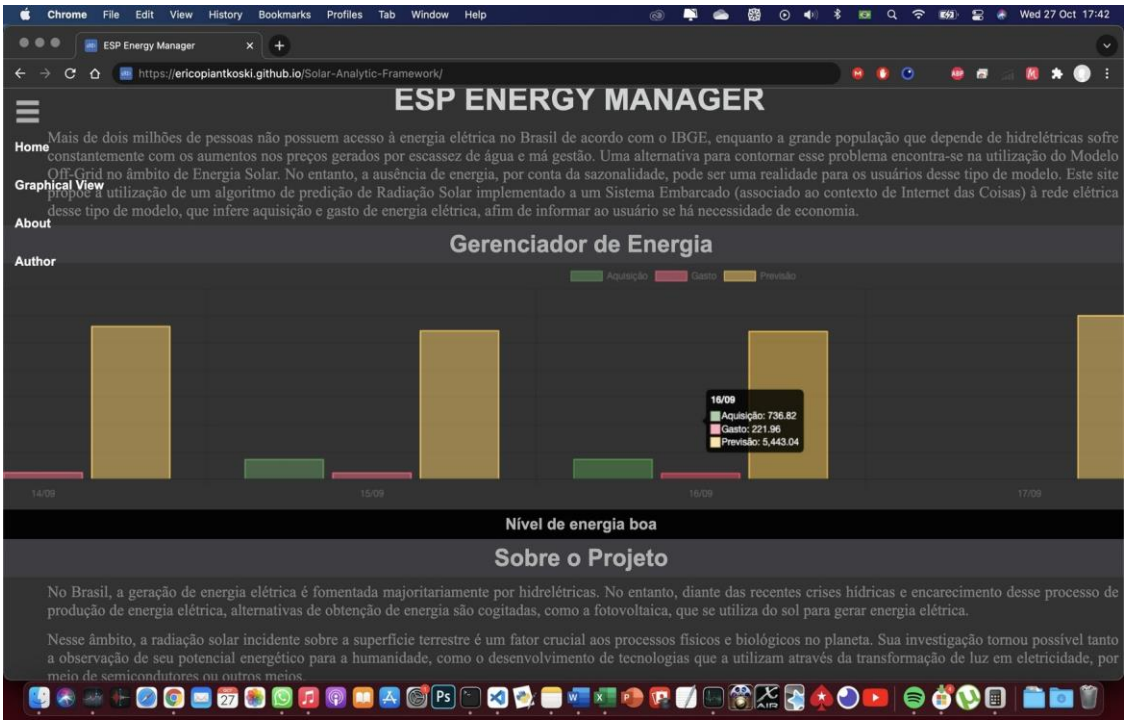

Figure 12 – Presentation of data to the user through a website. Source: Author (2021).

4. Results and Discussion

Revista Multidisciplinar do Nordeste Mineiro, v.10, 2025
ISSN 2178-6925
DOI:10.61164/rmnm.v10i1.4031

Based on the tests carried out during the month of September 2021, the data generated from the predictions based on the prediction algorithm, projected on the efficiency of the solar panel, consists of three types: **1.** Acquisition through the solar panel; **2.** Prediction acquired through the algorithm; **3.** Product of the prediction by the efficiency of the panel (estimated gain).

The user's acquisition data was saved in an .xlsx document as an empirical database, and used to produce the graphs in Figure 13 and Figure 14, which explain the accuracy of the algorithm.

Since the algorithm was executed on September 30, 2021 and does not have acquisition bases from the previous month, the calculation of the efficiency was therefore referred to the last value obtained (e.g., in this case, acquisition on September 30) and, therefore, the estimated gain will be exactly the same as the acquisition for that day.

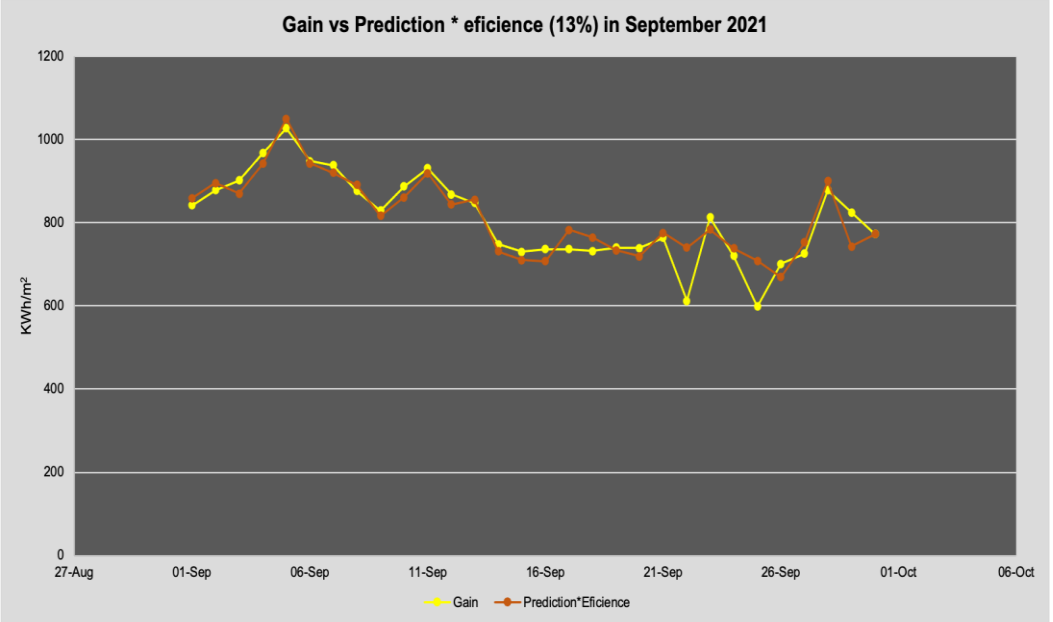

Figure 13 – Gain compared with the product of the Prediction by the efficiency of the solar panel Source: From the author (2021).

Given the above, if the efficiency algorithm were executed the following day (October 1, 2021), with the existing empirical bases (e.g., without it being executed previously), it would return the average efficiency (e.g., 13% also, also obtaining the

Revista Multidisciplinar do Nordeste Mineiro, v.10, 2025
ISSN 2178-6925
DOI:10.61164/rmnm.v10i1.4031

integer part).

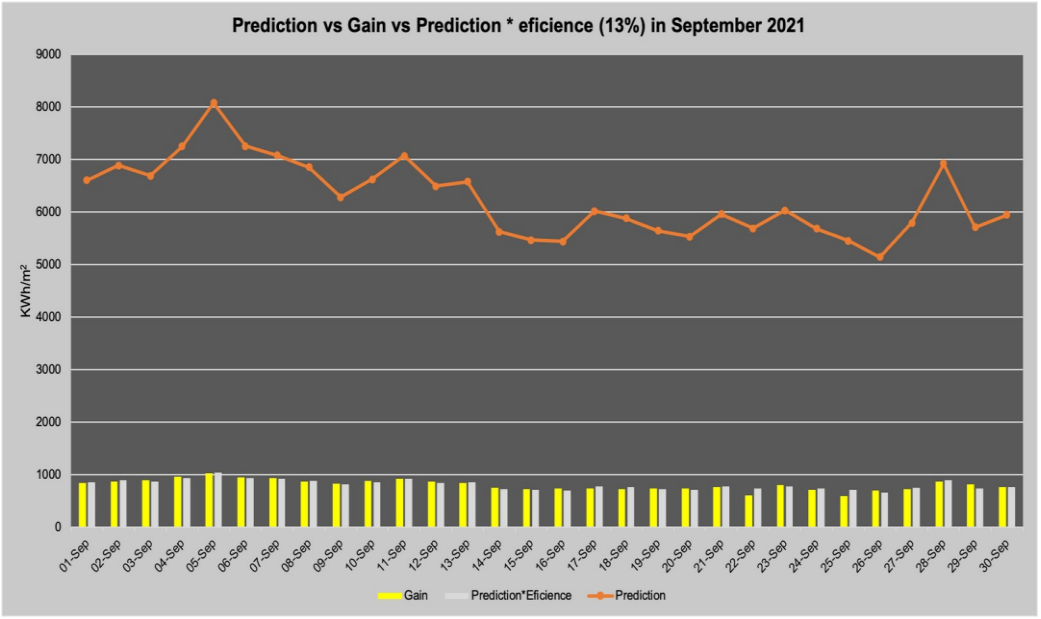

Figure 14 – Global view of prediction compared to user acquisition with estimated gain. Source: Author (2021).

Accordingly, if the algorithm were to be executed after it had already been executed previously (i.e., if there had already been an efficiency return of 13%), then the efficiency resulting from this execution would be compared to the return from the previous execution, and the lowest efficiency value would prevail (i.e., given that degradation always increases, therefore, efficiency will always fall over time, never the other way around). Thus, the yellow and green columns for each day in Figure 12, on the website, show the same prediction and gain as in Figure 14 (for the same days).

5. Conclusions and Future Works

This paper presents a proposal for a Solar Radiation prediction algorithm based on the *Off-Grid* model, implemented in an embedded system based on meteorological data and in the context of the **Internet of Things (IoT)**. Thus, several electronic components were used to develop the proposal, such as: **MCP3008** microchip modules, **ESP8266 V3 12F** module, **ACS712 Allegro 20A**

ISSN 2178-6925
DOI:10.61164/rmnm.v10i1.4031

module, **1mx1m MPrime M150P-12v Solar Panel**, among others. In addition, the proposal in question allows the end user to manage energy generation through a web interface that shows energy gain, expenditure, and an overall view of energy compared to the user's prediction. That is, the proposed system infers the acquisition and expenditure of electrical energy to inform the user if there is a need for energy savings. In this context, the results of radiation forecasting, when compared to energy acquisition, were close and, consequently, promising during the testing period, September 2021.

Among the main contributions of this work, the following stand out:

1. It has open source code developed in *Python*, *MicroPython*, *HTML*, and *JavaScript*;
2. The proposed algorithm improves the assertiveness of the *Off-Grid* system in relation to energy generation prediction;
3. The prediction technique employed, unlike others in the literature, used different strategies for choosing the meteorological station databases, diverging for the **INPE** and **BDMEP** databases, which makes our approach more efficient;
4. The results of estimated gains in the user's system proved to be convenient to help the user have an idea of the energy production in their *Off-Grid* system and, therefore, allow energy savings, since the data proved to be consistent when compared to the prediction.

Despite the advantages listed, the short testing period does not characterize the consistency of this system, requiring tests over longer periods, such as one or two years. Another observation is that the work depends on multiple variables, as observed throughout Section 3.

Finally, the results shown in Section 4 were satisfactory despite the short observation time and, consequently, were convenient to help the user have an idea of the energy production in their own *Off-Grid* system.

As future work, we intend to consider approaches for larger systems. For

example, in this work, we took into account a solar panel with the following dimensions m^2; however, another approach could be implemented for larger systems. This new approach would consist of creating an acquisition area (x square meters), multiplying it by the radiation acquisition estimate, and, in this way, comparing the estimated gain with a higher expense.

6. References

AMBARITA, H. **Development of software for estimating clear sky solar radiation in Indonesia**. Journal of Physics: Conference Series. Medan, Indonesia, 2017. Acesso em: 14 de Novembro de 2019. DOI: https://doi.org/10.1088/1742-6596/801/1/012093.

BOSO, A. C. M. R.; GABRIEL, C. P. C.; GABRIEL FILHO, L. R. A. **Análise de Custos dos Sistemas Fotovoltaicos On-grid e Off-grid no Brasil**. ANAP Brasil, 1904-3240, v.8, n.12, p.57-66, 2015. Acesso em: 14 de março de 2018. DOI: https://doi.org/10.17271/1984324081220151138.

FU, Pinde; RICH, Paul M. **A Geometric Solar Radiation Model and its Applications in Agriculture and Forestry**. Proceedings of the Second International Conference on Geospatial Information in Agriculture and Forestry. p.357-364, jan. 2000. Acesso em: 30 de outubro de 2019. DOI: https://doi.org/10.1016/s0168-1699(02)00115-1.

GUNJAL, P. et al. **An IOT Based Solar Power Robotic Tracking & Monitoring System**. International Conference on Communication and Information Processing (ICCIP). Acesso em: 10 de novembro de 2019. DOI: http://dx.doi.org/10.2139/ssrn.3416985.

HARTLEY, M.; CARLINI, L.; BELLOCCHI, G. **A Software Component for Estimating Solar Radiation**. ELSEVIER. p. 411-416, jul. 2005. Acesso em: 12 de novembrp de 2019. DOI: https://doi.org/10.1016/j.envsoft.2005.04.002.

MOOJEN, Thomaz M. B.; CAVALCANTE, Rosane B. L.; MENDES, Carlos A. B. **Avaliação da Radiação Solar com Base em Dados de Nebulosidade** . Geografia, v.21, n.3, p.41-55. Londrina-PR, set/dez. 2012. Acesso em: 20 de setembro de 2019. DOI: https://doi.org/10.5433/2447-1747.2012v21n3p41.

POP-VADEAN, A. et al. **Harvesting Energy an Sustainable Power Source, Replace Batteries for Powering WSN and Devices on the IoT**. IOP Conference Series: Materials Science and Engineering. p. 1-9, 2016. Acesso em: 15 de novembro de 2019. DOI: https://doi.org/10.1088/1757-899X/200/1/012043.

ISSN 2178-6925
DOI:10.61164/rmnm.v10i1.4031

RAMAN, A. P.; LI, W.; FAN, S. **Generating Light from Darkness**. Joule. p.1-8, Los Angeles-California, nov. 2019. Acesso em: 01 de novembro de 2019. DOI: https://doi.org/10.1016/j.joule.2019.08.009.

SPANIAS, Andreas S. **Solar Energy Management as an Internet of Things (IoT) Application**. 8th International Conference on Information, Intelligence, Systems & Applications (IISA). Lanarca – Cyprus, mar. 2018. Acesso em: 04 de novembro 2019. DOI: 10.1109/IISA.2017.8316460.

SUDDARD-BANGSUND, John et al. **Organic Salts as a Route to Energy Level Control in Low Bandgap, High Open-Circuit Voltage Organic and Transparent Solar Cells that Approach the Excitonic Voltage Limit**. Advanced Energy Meterials. p.1-10, East Lansing – MI, 2016. Acesso em: 27 de novembro de 2019. DOI: https://doi.org/10.1002/aenm.201501659.

Visual Studio Code. **Wikipedia**. Março de 2020. Acesso em: 31 Mar. 2020. Disponível em: https://en.wikipedia.org/wiki/Visual_Studio_Code.