Final Report

**Submitted to the Embedded Systems Competition (ESC)
of SBESC 2024**

Project Title: Boia para Monitoramento e Prevenção de Inundações

Students: José Batista de Souza Júnior, Larissa da Silva Matos and Samuel Henrique Guimarães Alencar

Professor: Wagner Guimarães Al-alam

University: Universidade Federal do Ceará - Campus Quixadá

JEMS ID: 242989

Link to video: SBESC - Vídeo.mp4

Declaration of Originality

We hereby declare that this report and the work reported herein was composed and originated entirely by ourselves. Information derived from the published and unpublished work of others has been acknowledged in the text and a list of citations is given in the references section

Professor's Signature: ASSINADO DIGITALMENTE WAGNER GUIMARAES AL ALAM A conformidade com a assinatura pode ser verificada em: http://serpro.gov.br/assinador-digital SERPRO

Name (in Block Letters): WAGNER GUIMARÃES AL-ALAM

Student's Signature: José Batista de Souza Júnior

Name (in Block Letters): JOSÉ BATISTA DE SOUZA JÚNIOR

Student's Signature: gov.br Documento assinado digitalmente LARISSA DA SILVA MATOS Data: 20/11/2024 23:29:42-0300 Verifique em https://validar.iti.gov.br

Name (in Block Letters): LARISSA DA SILVA MATOS

Student's Signature: gov.br Documento assinado digitalmente SAMUEL HENRIQUE GUIMARAES ALENCAR Data: 20/11/2024 23:27:41-0300 Verifique em https://validar.iti.gov.br

Name (in Block Letters): SAMUEL HENRIQUE GUIMARÃES ALENCAR

Date: 20/11/2024

Abstract

Flood monitoring and prevention are critical in mitigating the impacts of extreme weather events, which are becoming more frequent due to climate change. In this work, a floating monitoring device is proposed to detect and respond to rising water levels in real time. Designed to operate autonomously in remote areas, the device integrates sensors for measuring water level and environmental parameters, using solar power for sustained energy efficiency. Data transmission is achieved through LoRaMESH, with optional integration to the Helium network for extended range when available. A Node-RED-based interface visualizes data through an intuitive dashboard, allowing local authorities and communities to monitor and respond promptly to flood risks. This system overcomes limitations in current monitoring infrastructure, particularly in regions without hydrometeorological stations or with delayed data processing. Initial results suggest that this approach provides a scalable and reliable solution for flood preparedness, enhancing early warning capabilities and improving community resilience against climate-induced flooding.

Keywords: flooding, real time, LoRaMESH, Helium Network

Content

1. Introduction

Climate change is increasingly contributing to the frequency and intensity of extreme weather events [1], emphasizing the urgent need to develop effective strategies for prevention and mitigation. In Brazil, these challenges are particularly evident in the context of rainfall, as the country's tropical climate brings high precipitation levels during certain seasons. The resulting impacts include infrastructure damage, landslides, floods, and inundations [2].

While Brazil has an extensive hydrometeorological network with approximately 11,000 stations, the Geological Survey of Brazil (SGB) acknowledges that only around 4,200 stations are part of the national basic network (RHN). This distribution leaves gaps in certain regions, particularly in smaller or more remote municipalities, where adequate monitoring is often lacking. Additionally, even in municipalities with monitoring stations, data is frequently processed in centralized locations, often far from where the stations are situated. This delay in data processing can hinder a swift response to calamity events, reducing the effectiveness of emergency measures [3].

This project offers a novel solution to the challenges posed by extreme rainfall and its consequences, particularly in flood-prone and remote areas. By integrating a floating device within a stationary pipe installed on the riverbed, the system measures water level fluctuations using a laser sensor at the top of the pipe. Although the prototype does not yet include Helium network connectivity and solar panels, these solutions have been designed for inclusion in a future version, highlighting the project's potential for expansion. Once fully implemented, the device will be powered by solar energy and capable of transmitting real-time data via the Helium network, ensuring accessibility even in areas with minimal infrastructure. A user-friendly dashboard enables local authorities and communities to respond effectively to changing conditions, improving disaster preparedness and resilience.

This solution is particularly impactful for underserved communities in Brazil, where floods often lead to severe consequences such as loss of life, displacement, and property damage [4]. Many smaller municipalities lack efficient monitoring systems, leaving authorities without the real-time data needed to act swiftly during crises [5]. By offering an autonomous, low-cost monitoring device that provides early warnings, the system enables preventive measures like evacuations and flood barrier installations. Its reliance on solar power ensures functionality in off-grid areas, making it a sustainable, scalable, and cost-effective tool for improving disaster response in vulnerable regions [6].

2. System Description

This section presents all system descriptions, including requirements, technologies and components used, as well as the adopted hardware and software architecture.

2.1 System Requirements

Some requirements were defined to guide the development of functionalities and to ensure the quality of the system. The functional requirements describe the features the software must provide to meet the needs of the proposed solution. Table 2-1 shows the defined functional requirements.

Table 2-1: Functional Requirements of the System

Functional Requirements	Description	Classification
FR1. Measure the river water level difference	The system must measure the river water level difference using a distance sensor.	Essential
FR2. Measure ambient temperature and humidity	The system must measure ambient temperature and humidity using a waterproof sensor.	Important
FR3. Measure float stabilization	The system must determine the float's stabilization using an accelerometer.	Important
FR4. Transmit data via LoRa	The system must connect and transmit data to a receiver node using LoRaMESH.	Essential
FR5. Monitor and report sensor failures	The system must monitor the sensors' operation and report failures in their operation or communication.	Important
FR6. Real-time remote monitoring	The system must send data to the gateway in real time and enable remote monitoring of the data.	Essential

On the other hand, non-functional requirements define the overarching characteristics necessary for the solution to work effectively. Table 2-2 outlines the key non-functional requirements for the system.

Table 2-2: Non-Functional Requirements of the System

Non-Functional Requirements	Description	Classification
NFR1. Autonomous Power Supply	The system must operate independently, powered by clean and renewable energy sources.	Important
NFR2. Weather Resistance	The system must withstand harsh environmental conditions, including heavy rain, strong winds, and prolonged water exposure.	Essential
NFR3. Performance and Low Latency	The system must deliver efficient task execution, ensuring minimal delay in data transmission.	Essential
NFR4. Cost Efficiency	The system should be built using affordable components and open-source hardware/software to enable flexibility and collaboration.	Essential
NFR5. Reliability	The system must function dependably, ensuring data integrity during collection and transmission.	Essential
NFR6. Energy Efficiency	The system must consume minimal power while maintaining optimal performance.	Important

NFR7. Scalability	The system should allow for easy expansion, supporting the addition of more sensors and communication methods.	Important

2.2 Selected Components and Sensors

To meet the requirements and achieve the objective of the proposed solution, components and sensors were selected based on specific criteria, forming our prototype. These criteria include: comprehensive documentation, affordable cost, ease of configuration, fast response time, low power consumption, reliability, water resistance, and the availability of implementation examples or pre-developed libraries for these devices.

2.2.1 Laser Distance Sensor TOF10120

To meet the **FR1** requirement, the TOF10120 laser sensor was chosen for its precise and reliable long-range distance measurements, often used in autofocus applications. It supports both serial and I2C communication, giving developers flexibility in implementation. With a range of 100 to 1800 mm and an accuracy of ±5% in indoor environments, it's well-suited for the task. The sensor operates on a 3 to 5 V power supply and draws 35 mA during use [7].

2.2.2 SHT30 Temperature And Humidity Sensor

To meet the **FR2** requirement, the SHT30 sensor was selected for measuring ambient humidity and temperature. Manufactured by Sensirion, this sensor supports a wide supply voltage range from 2.4 V to 5.5 V and has a typical power consumption of 0.002 mA on average, ranging from 0.0002 mA in idle state (single-shot mode) to up to 1.05 mA during measurements. Additionally, it can measure temperatures within a range of -40°C to 125°C. The sensor features I2C communication, and the waterproof encapsulated version was chosen for this project [8].

2.2.3 MPU6050 Accelerometer and Gyroscope

To measure the stabilization of the pipe, as specified by the **FR3** requirement, the MPU6050 accelerometer was used, a sensor widely employed in prototypes due to its affordable cost and robust documentation. Communication between the sensor and the microcontroller is established via I2C. Measured acceleration values are used to calculate roll and pitch angles, allowing for effective stabilization monitoring. Typically, it operates at a voltage of 3.3 to 5V and consumes approximately 4 mA.

2.2.4 LoRaMESH Radioenge Module

To enable communication between two or more nodes via LoRa, it is essential to use a suitable module. In this context, the competition organization provided the Radioenge LoRaMESH module. This module allows for efficient preparation and transmission of packets to other nodes, with an average current consumption of approximately 30 mA, as calculated based on transmission,

reception, and idle mode cycles. It offers simplified configuration through a serial interface and employs a MESH network topology, where devices connect directly and communicate dynamically and flexibly with each other [10].

2.2.5 ESP32 Microcontroller

The ESP32 microcontroller plays a central role in managing the entire application, being responsible for reading sensors, processing data, preparing messages, and transmitting them to other nodes. This microcontroller is highly versatile, offering both Wi-Fi and Bluetooth Low Energy (BLE) connectivity, technologies widely used in Internet of Things (IoT) applications [11].

Additionally, the ESP32 features dual cores, enabling multiprocessing, which makes it ideal for applications requiring performance and efficiency. It is also supported by the ESP-IDF (Espressif IoT Development Framework), a robust development environment created by Espressif. This framework simplifies device programming and provides strong integration with C and FreeRTOS, enhancing its flexibility and capabilities [11].

2.3 Hardware Architecture

During the component selection process, the hardware architecture was simultaneously designed to ensure that all elements were properly connected to the appropriate peripherals, guaranteeing the integrated functionality of the system.

The ESP32 microcontroller provides a wide array of peripherals, a feature extensively leveraged in our solution. Two UART peripherals were utilized: one for communication with the LoRaMESH module and another for the TOF10120 distance sensor. Additionally, two I2C peripherals were employed: one for the MPU6050 accelerometer and another for the SHT30 temperature and humidity sensor.

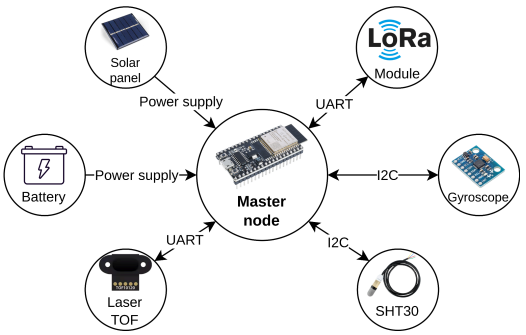


Figure 2-1: Hardware Architecture

Figure 2-1 illustrates the architecture adopted in our solution, detailing the connections between the components and the microcontroller. The diagram also highlights the communication protocols used and the data flow, represented by arrows.

An autonomous solar-powered solution was designed to supply energy to the system. The setup includes a solar panel integrated with a Li-Ion battery and a charging module. In this arrangement, under sunlight, the system would be powered directly by the solar panel while

simultaneously charging the battery. In the absence of sunlight, the system would switch to operating solely on the energy provided by the battery.

However, at the prototype stage, this solution was not implemented. To simplify initial validation, the system was powered using a power bank.

2.4 Software Architecture

The embedded software was developed in C, a language chosen for its lightweight nature and efficiency. Another key factor was the team's greater technical expertise in this language, which streamlined the development process and reduced implementation time.

Although more accessible frameworks like Arduino are available, ESP-IDF was chosen as the development framework for technical and strategic reasons:

- Full compatibility with the ESP32, leveraging all the hardware features.
- Low-level control over peripherals, enabling detailed and customized configurations.
- Enhanced performance efficiency, optimizing the use of Flash memory and RAM.
- Native integration with FreeRTOS, simplifying multitasking management and fully utilizing the ESP32's dual cores.

To manage the board's resources and implement multiprocessing, FreeRTOS was used - a real-time operating system widely recognized for its efficiency in embedded systems. It enables:

- Optimized task management, dividing responsibilities in a modular way.
- Greater control over system flow and resource utilization.
- Simplified maintenance and scalability, as each task has a well-defined responsibility.

2.4.1 Layered Architecture

The software architecture was designed following a layered pattern, widely used in embedded systems due to its modularity and separation of responsibilities. This model simplifies maintenance, code reuse, and future expansion. The architecture was divided into four layers, with each layer accessing only the one directly below it, as illustrated in Figure 2-2.

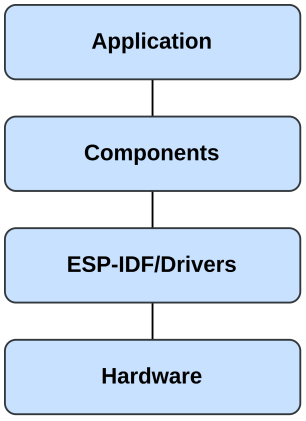


Figure 2-2: Software Architecture

At the base of the architecture is the hardware layer, which encompasses the ESP32 microcontroller, its peripherals, and the sensors connected to it. This layer directly handles physical signals and provides the foundation for the system's functionality.

Sitting above the hardware layer is the ESP-IDF, which provides drivers and abstractions for peripherals like UART and I2C. These drivers make it easier to manage peripherals while ensuring a consistent and reliable interface for sensor communication.

The components layer follows a common pattern in ESP-IDF, where specific libraries for the sensors used are implemented. This approach allows for:

- Modularity, organizing each sensor's code into reusable, self-contained components;
- Ease of integration, allowing sensors to be added or modified without affecting the rest of the system, including reusable drivers and abstractions.

At the top of the architecture is the application layer, which contains the system's core logic and all tasks managed by FreeRTOS. This layer is responsible for:

- Coordinating tasks such as sensor readings, data processing, and message transmission;
- Implementing the solution's specific logic to ensure the project requirements are met.

3. System Implementation

Initially, the system was designed to integrate with the Helium Network and use the Helium Console for data visualization. However, the Helium network's gateway operates on the LoRaWAN protocol, while the project was built on LoRaMESH. This led to a different network topology, causing communication issues and message format incompatibilities.

As a solution, the creation of a bridge node was considered. This node would receive LoRaMESH messages, convert them to the LoRaWAN format, and forward them to the nearest Helium gateway. However, due to the lack of Helium network coverage in the Sertão Central region of Ceará and the high cost of acquiring a Helium device, this approach was deemed unfeasible.

For prototyping and validation purposes, two LoRaMESH nodes were implemented: a master node and a receiver node, as shown in Figure 3-1.

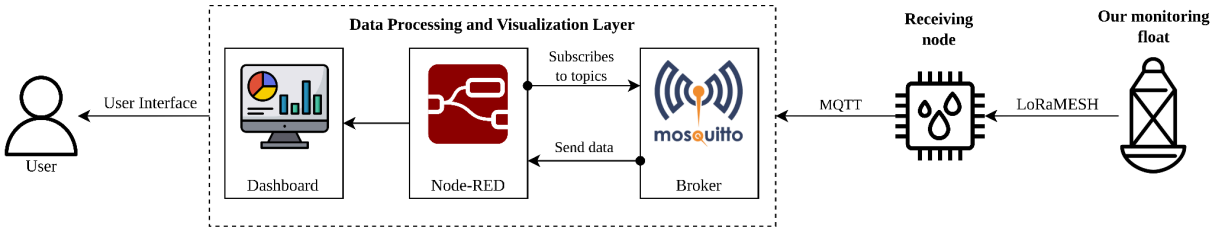


Figure 3-1: Diagrama de blocos do sistema

The monitoring float, detailed in Figure 2-1, functions as the master node, responsible for collecting sensor data, formatting messages, and transmitting them to the receiver node via

LoRaMESH. The receiver node, equipped with a LoRaMESH module, connects to Wi-Fi, processes the received messages, and forwards them via MQTT to the Mosquitto broker, which operates locally.

This setup enables the use of Node-RED, a browser-based visual development tool, to create information flows and design dashboards. The end user can effortlessly view the collected data and interact with the system through a simple and intuitive interface, whether accessing it locally or remotely. Node-RED connects to the broker's topics, retrieves the data, and displays it on visual dashboards, enabling efficient remote monitoring and providing a practical and user-friendly experience.

The source code for this project is available at: github.com/SamuelLost/sbesc-esc-2024.

3.1 Custom IDF Components

During the system implementation, it was necessary to develop new components to integrate the proposed solution effectively. These components, detailed below, ensure the system's functionality and performance.

The **accelerometer_sensor** component provides an abstraction for the accelerometer, managing initialization, configuration, data reading, and acceleration value processing. In contrast, the **i2c_driver** simplifies interaction with the ESP32's I2C bus by offering an intuitive interface for initialization, deinitialization, data writing, and reading, while also abstracting the ESP-IDF's driver/i2c component.

Moreover, the **laser_sensor** component handles the laser distance sensor, including its initialization and distance measurement readings. Similarly, the **lora_module** initializes and configures the LoRaMESH module, while also managing the preparation, transmission, and reception of packets.

In terms of communication, the **mqtt_driver** facilitates the connection to the MQTT broker by handling message publishing and topic subscription. Meanwhile, the **temperature_sensor** is responsible for initializing the temperature and humidity sensor, reading data, and processing the values.

Furthermore, the **uart_driver** abstracts the UART peripheral, providing a straightforward interface for initialization, deinitialization, and data operations. Additionally, the **utils** component supports system operations by offering macros, constants, and utility functions. Finally, the **wifi_driver** abstracts the ESP-IDF Wi-Fi component, simplifying the configuration and management of Wi-Fi connectivity.

Together, these components ensure the system is modular, maintainable, and capable of delivering streamlined functionality.

3.2 Application Startup

Proper initialization of the board and devices is a critical step to ensure the system functions as intended. Failures during this stage can impact performance, prevent the achievement of requirements, and compromise the solution's overall reliability. To mitigate this, a structured approach

was implemented to ensure a safe and efficient initialization process. Figure 3-2 illustrates the sequence of steps performed during the system initialization. The process follows these steps:

1. **Sensor and Module Check:** The system verifies whether all sensors and modules have been initialized correctly. If any issue is detected, the ESP32 microcontroller automatically restarts after a 7-second interval. This approach ensures that temporary failures can be resolved without human intervention.
2. **Queue and Mutex Creation:** Next, the FreeRTOS queues and mutexes are created and initialized. These are essential for task communication and synchronization. If this step fails, the board will restart to prevent system inconsistencies.
3. **Task Creation:** After configuring the synchronization resources, the system initializes the five main tasks developed for the project. If any task fails to initialize, the system automatically restarts to ensure all functionalities are correctly loaded.
4. **Task Execution:** Once all tasks are created, the system begins execution. Tasks are scheduled according to the FreeRTOS scheduling algorithm and the priority level assigned to each, ensuring that critical operations are handled at the appropriate time.

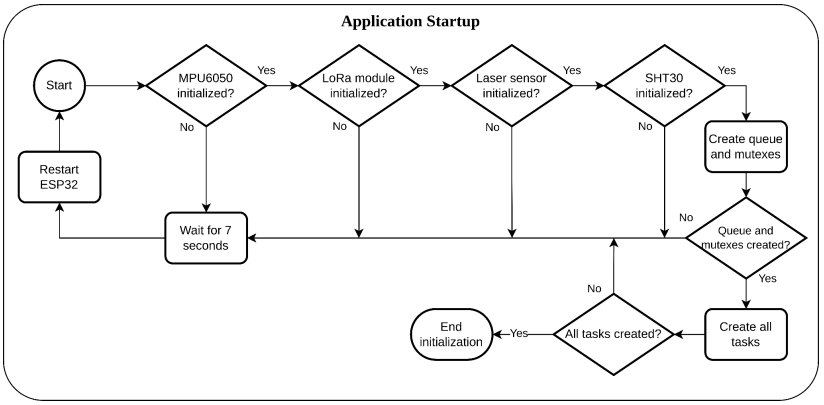


Figure 3-2: Application startup flowchart

This structured process not only enhances the system's robustness but also simplifies the identification and resolution of issues during development and operation.

3.3 Overview of Developed Tasks

The master node is composed of five main tasks: vTaskAccelerometer, vTaskLaserSensor, vTaskLora, vTaskTemperature, and vTaskHeartbeat, each with a distinct responsibility. This design ensures modularity and simplifies maintenance. The tasks format messages as: “Sensor ID,Device ID,data1,data2,...,dataN”, where 0 represents the master node's ID.

3.3.1 Accelerometer Task

Figure 3-3 illustrates the steps of vTaskAccelerometer, which is responsible for measuring acceleration using the MPU6050 and calculating the roll and pitch angles. This task runs every 4 seconds, has a priority level of 3, and is assigned to the ESP32's application core (core 1). During

execution, it attempts to acquire the mutex protecting the sensor. Once the mutex is acquired, it measures the values, prepares the LoRa packet, and sends it to a queue for packet storage. Afterward, it releases the mutex and enters a blocked state for 4 seconds. The generated message follows the format: “ACC,0,pitch value,roll value”.

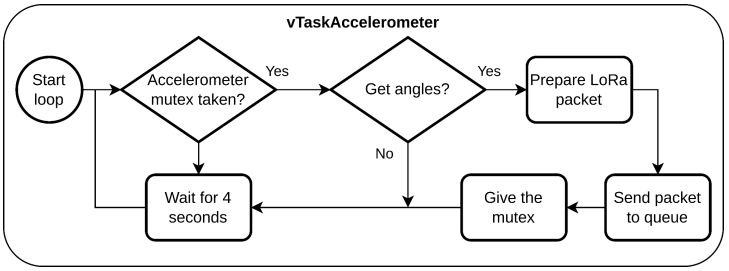


Figure 3-3: vTaskAccelerometer flowchart

3.3.2 Temperature Task

Figure 3-4 outlines the steps performed by the task responsible for measuring humidity (in percentage) and temperature (in Celsius). This task runs every 10 seconds, has a priority level of 3, and is assigned to the application core (core 1). During execution, the task attempts to acquire the mutex protecting the sensor. If successful, it reads the data. If the reading is successful, the task prepares a LoRa packet, sends it to the packet queue, and releases the mutex; otherwise, the task enters a blocked state. The generated message follows the format: “*TMP,0,temperature,humidity*”.

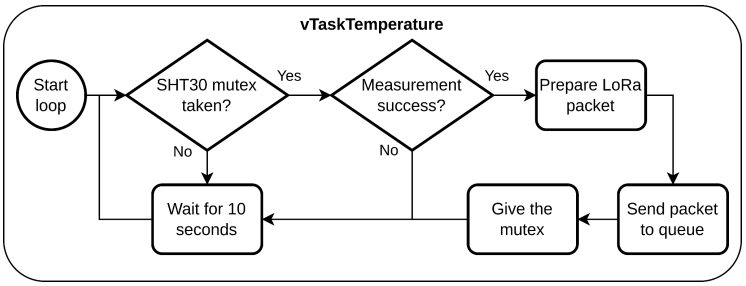


Figure 3-4: vTaskTemperature flowchart

3.3.3 Laser Task

Figure 3-5 depicts the flowchart of **vTaskLaserSensor**, tasked with measuring distance using the TOF10120 sensor. This task runs every 3 seconds, has a priority level of 3, and is assigned to core 1.

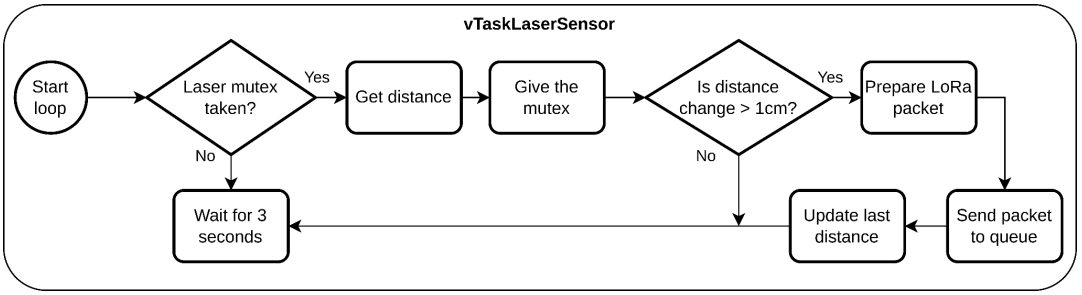


Figure 3-5: vTaskLaserSensor flowchart

During execution, it attempts to acquire the mutex protecting the sensor. Once acquired, it reads the distance in millimeters and releases the mutex. It then checks for a minimum variation of 1 cm compared to the previous reading. If such a variation is detected, a new message is formatted, sent to the message queue, and the distance value is updated. Finally, the task enters a blocked state. The generated message follows the format: "*LS,0,distance value*".

3.3.4 Heartbeat Task

The **vTaskHeartBeat**, illustrated in Figure 3-6, runs every 15 seconds, has a priority of 1, and is assigned to the protocol core (core 0). Its purpose is to check the sensors' status, monitor the free memory used by each task, prepare a LoRa message with this information, and send it to the message queue. The message follows this format: "*HB,0,accelerometer task memory,laser task memory,temperature task memory,LoRa task memory,heartbeat task memory,uptime,MPU6050 status,TOF10120 status,SHT30 status*".

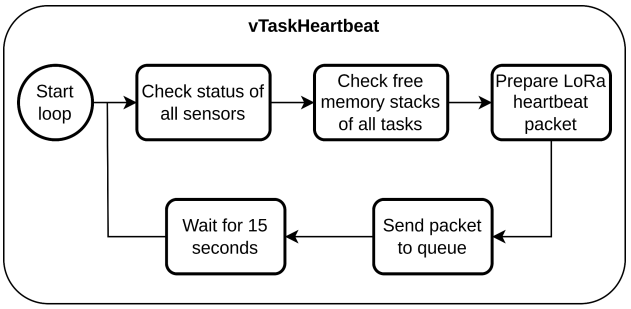


Figure 3-6: vTaskHeartBeat flowchart

3.3.5 LoRa Task

Figure 3-7 illustrates the task responsible for sending LoRa messages to the MESH network. With the highest priority among all tasks (priority 24), it runs every 2 seconds on the protocol core (core 0). The task continuously consumes the LoRa packet queue and, upon receiving a message, attempts to acquire the mutex before transmitting the message via LoRa.

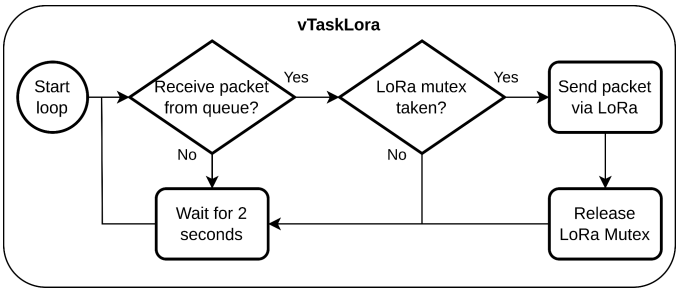


Figure 3-7: vTaskLora

3.3.6 Receiver Node

On the receiver side, there is a single task responsible for processing all incoming messages. When a LoRaMESH packet arrives, the task checks the message field to identify the command. Each sensor is identified by a specific command: "0x11" for angles, "0x12" for temperature, "0x13" for

distance, and “0x14” for the heartbeat. The string message for each sensor is reconstructed using the “sscanf” function, enabling the extraction of individual data. Once extracted, the data is published to the corresponding MQTT topic.

For distance-related messages, the received value is processed on the receiver side to calculate the height of the water column. Figure 3-8 illustrates the internal structure of the prototype’s pipe, where “**c**” represents the total length of the pipe (a constant), “**b**” is the distance measured by the sensor, “**d**” is the height of the floating device, and “**a**” is the calculated water column height.

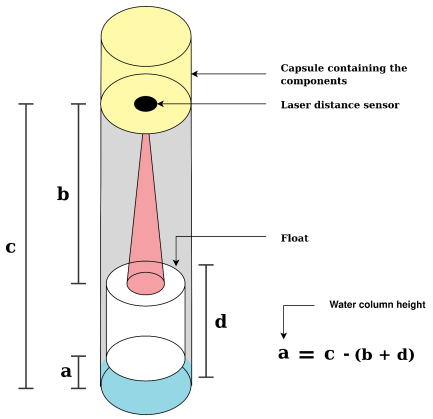


Figure 3-8: Scheme of system calculations

Based on the equation (3 - 1),

$$a = c - (b + d) \quad (3 - 1)$$

The water column’s height is calculated, and after processing, the resulting data is sent to the corresponding topic on the broker via MQTT, enabling remote visualization and monitoring.

3.4 LoRaMESH Communication

To enhance the security of the network, a password was implemented to ensure that only authorized devices can participate in communication. This measure helps protect the integrity of the data being exchanged and prevents unauthorized access. For the LoRaMESH protocol to function correctly, the buffer must be filled with the following fields in order:

Table 3-1: LoRaMESH packet

ID (LSB)	ID (MSB)	CMD	Payload (N bytes)	CRC (LSB)	CRC (MSB)

The **ID** identifies the device that will receive the message. The **CMD** determines the action or type of data contained in the message. The **Payload** carries the useful information, while the **CRC** ensures data integrity by verifying that no corruption occurred during transmission. This structure is essential to guarantee reliable communication between nodes in the LoRaMESH network.

3.5 Estimated Total Power Consumption

As outlined in Section 2.2, the sensors' current consumption values were identified, allowing the estimation of the system's approximate total current draw in active mode:

Table 3-2: Estimated Total Power Consumption

Component	**Current (mA)**
ESP32	240
SHT30 Sensor	1.05
LoRaMESH Node	30
MPU6050 Accelerometer	4
TOF 10120 Sensor	35
Total	**310.05**

Since the system operates on a 5v power supply, the power consumption in active mode is:

$$P = V \times I = 5V \times 0.31005A = 1.55025\,W$$

Using this result, the energy consumption for a typical day is calculated to be approximately 36.8 Wh.

4. Results

This section discusses the results obtained during the development of the system and prototype, along with a brief description of the tests performed.

4.1 Circuit Diagram

The circuit diagram of the prototype, shown in Figure 4-1, was created using Fritzing, an open-source tool for electronic hardware design. The diagram details the connections of the ESP32 with the project's sensors and modules, including the LoRaMESH, SHT30, MPU6050, and TOF10120, as well as the power peripherals. Components not available in Fritzing were represented illustratively and thoroughly documented. This diagram serves as a valuable reference for assembly, debugging, and future hardware modifications.

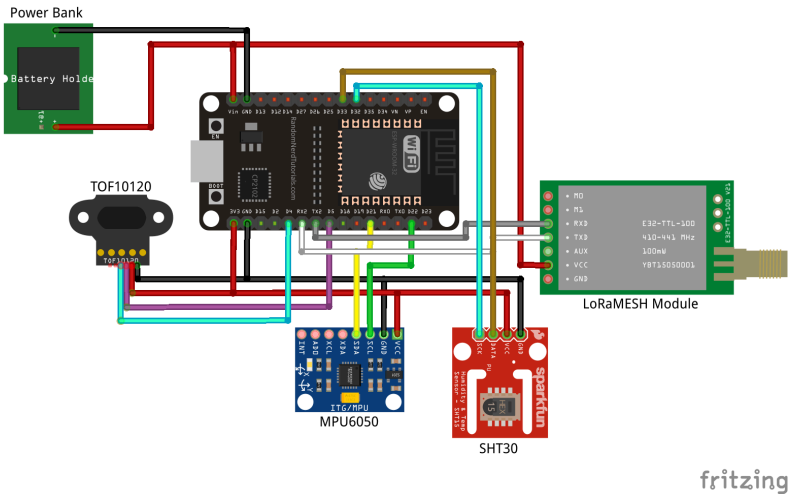
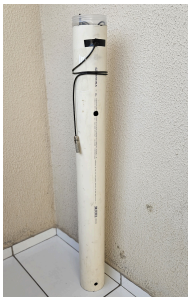


Figure 4-1: Prototype circuit diagram

4.2 Prototype

The system prototype was constructed using a PVC pipe and a specially designed capsule to house the components. This design ensures modularity and provides easy access to the microcontroller, batteries, and other system parts, facilitating future updates or improvements. The pipe features two strategically placed holes: one near the top for air release and another close to the base for water intake. These holes allow a floating object inside the pipe to move vertically as the water level within the pipe changes.



a)

b)

Figure 4-1: The pipe a) and capsule b) with components

4.3 Dashboards

The dashboard developed using Node-RED was designed to be intuitive and facilitate the visualization of information. It consists of two tabs: one dedicated to data collected from the sensors and another focused on system monitoring.

Figure 4-2 presents a screenshot of the main tab, which displays the most critical system information. The interface highlights two gauges: one for temperature, with the sensor's reading range spanning from -40 ºC to 125 ºC, and another for humidity, ranging from 0% to 100%. Next to the humidity gauge is an interactive graph monitoring the water level in millimeters, represented in blue. As the water level rises, the blue "liquid" progressively fills the circle.

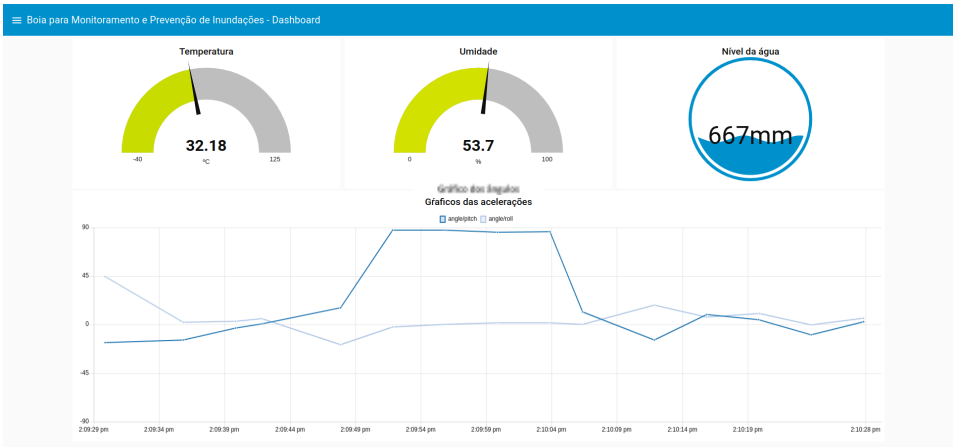


Figure 4-2: Main dashboard

At the bottom of the interface is a line graph illustrating the variation of roll (light blue) and pitch (dark blue) angles over a one-minute interval. This graph is essential for monitoring the device's stability, indicating whether it is tilted or upright. Tests were conducted to simulate device inclinations along the X and Y axes, as reflected in the recorded angle variations.

Figure 4-3 provides a screenshot of the system status monitoring tab. This tab includes multiple gauges that display the amount of free bytes available in the memory stack for each task. Additionally, it features text indicators that show whether a specific sensor is connected or not. The tab also tracks and displays the uptime of the ESP32, measured in seconds.

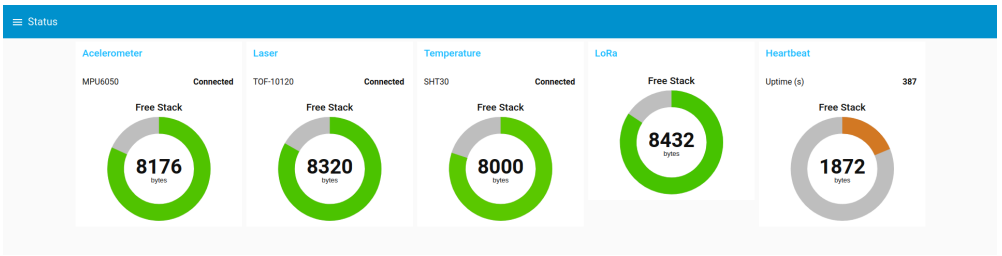


Figure 4-3: Status dashboard

4.4 Validations and tests

The system underwent both unit and integration testing to ensure functionality and reliability. Each sensor was individually tested to verify accurate readings under controlled conditions. For instance, the laser distance sensor was tested using varying water levels, confirming its ability to measure distances within the expected range. Similarly, the temperature and humidity sensor was evaluated across its operational range, producing consistent and reliable data.

Integration tests were performed to assess the system as a whole. The LoRaMESH communication module was tested for data transmission between the master and receiver nodes, demonstrating stable performance over varying distances. The accelerometer was validated by simulating inclinations along the X and Y axes, with roll and pitch angles recorded and visualized on the Node-RED dashboard. The dashboard itself was tested for real-time data updates and user interaction, ensuring a seamless and intuitive monitoring experience.

Stress testing was also conducted to evaluate the system’s performance under continuous operation. The microcontroller handled simultaneous task executions, including sensor readings, data processing, and message transmission, without failures or significant delays. While solar power and Helium network connectivity were not implemented in the prototype, simulations confirmed their feasibility for future integration. These tests indicate the system is robust, scalable, and ready for deployment in flood-prone areas.

5. Innovation

The proposed project introduces innovative features for environmental monitoring, particularly in remote river ecosystems, setting it apart from conventional buoy-based systems. While existing

solutions use sensor-equipped buoys to monitor variables like oceanographic, meteorological, and water quality data, this project integrates advanced technologies to address specific challenges.

The system leverages the Helium platform, combining LoRaWAN technology with a cryptocurrency-based economic model (HNT). This enables long-range communication at significantly lower costs, creating an economically self-sustained and efficient solution for areas with limited infrastructure. Additionally, 3D printing at the Federal University of Ceará's Quixadá campus facilitates localized, customizable manufacturing, allowing for rapid prototyping and flexibility in design.

Energy efficiency and scalability are key focuses of the project. Solar panels and rechargeable batteries power the low-consumption device, ensuring sustainable operation in off-grid locations with minimal maintenance. Real-time data analysis is managed by FreeRTOS, an open-source operating system that processes and transmits sensor data in a deterministic way, enabling rapid detection of environmental changes critical for early warning systems.

LoRa technology enhances communication capabilities by enabling long-range, low-energy data transmission, ideal for hard-to-reach areas. Furthermore, the system's open-source hardware design allows customization for various sensors and applications, fostering collaboration and extending the project's impact.

6. Conclusion

This project offers a solution to mitigate climate-related disasters, whether caused by natural events or human activities. It was inspired by the severe impact of floods and heavy rainfall in Brazil, a country still in need of greater preparedness to address climate emergencies. This work aims to foster new ideas and improvements in this area, helping to reduce the adverse effects of rainfall on vulnerable communities.

During development, various challenges arose but were quickly resolved thanks to a well-structured framework of meetings, schedules, and requirements established at the project's outset. Weekly meetings facilitated idea exchanges, clarified doubts, and guided critical decisions that advanced the system's development. Many key decisions were made with the support of the professor and advisor of the group, highlighting the importance of mentorship in academic projects.

The results achieved closely align with the project's initial objectives, even as adjustments were made along the way. The functional prototype successfully implemented the proposed concept: water levels were monitored as the floating device moved within the pipe, climatic variables such as humidity and temperature were captured to identify environmental changes, and the pipe's position was accurately tracked by the accelerometer. Additionally, data was effectively transmitted and presented using the LoRaMesh communication system proposed in the competition.

Lastly, solutions that could not be integrated into this prototype, such as Helium network connectivity and solar panels, were designed for inclusion in a future version of the product, underscoring the project's potential for expansion.

References

[1] Painel Intergovernamental sobre Mudanças Climáticas (IPCC): IPCC. (2021). *AR6 Climate Change 2021: The Physical Science Basis.* Geneva: Intergovernmental Panel on Climate Change. Available at: https://www.ipcc.ch/2021/08/09/ar6-wg1-20210809-pr/. Accessed on November 15.

[2] Springer, S., & Torres, L. (2024). Brazil Thought It Was Safe From Natural Disasters. Then Came the Flood. *The Wall Street Journal.* Available at: https://www.wsj.com/world/americas/brazil-thought-it-was-safe-from-natural-disasters-then-came-the-flood-247522a8. Accessed on November 15.

[3] Sistema Nacional de Informações sobre Recursos Hídricos. (2005). Apresentação. Available at: https://www.snirh.gov.br/hidroweb/apresentacao. Accessed on November 16, 2024.

[4] G1 Rio Grande do Sul. (2024). Um mês de enchentes no RS: veja cronologia do desastre. Available at: https://g1.globo.com/rs/rio-grande-do-sul/noticia/2024/05/29/um-mes-de-enchentes-no-rs-veja-cronologia-do-desastre.ghtml. Accessed on November 16, 2024.

[5] Fundação Republicana Brasileira. (2024). *A falta de infraestrutura nos pequenos e médios municípios*. Available at: https://fundacaorepublicana.org.br/a-falta-de-infraestrutura-nos-pequenos-e-medios-municipios-conjuntura-republicana-ed-no-170/. Accessed on: 16 de novembro de 2024.

[6] U. Shantha Kumar, Chetan Kumar, Dhanraj N., Vidya B., Kuri Pampapathi. (2024). *Solar Powered Flood Alert System*. Journal of Emerging Technologies and Innovative Research (JETIR), Volume 11, Issue 5. ISSN: 2349-5162. Available at: https://www.jetir.org/papers/JETIR2405684.pdf. Accessed on November 16, 2024.

[7] Shenzhen Hongcheng Technology Co., Ltd. (n.d.). *TOF10120 Time-of-Flight Ranging Sensor Datasheet*.

[8] Sensirion. (2022). *Datasheet SHT3x-DIS: Humidity and Temperature Sensor*. Version 7, December.

[9] InvenSense Inc. (2012). *MPU-6000 and MPU-6050 Product Specification*. Document Number: PS-MPU-6000A-00, Revision 3.3, Release Date: May 16.

[10] Radioenge. (2023). *LoRaMESH Module User Manual*. Revision July 2023.

[11] Espressif Systems. (2024). *ESP-IDF Programming Guide for ESP32*. Available at: https://docs.espressif.com/projects/esp-idf/en/stable/esp32/get-started/index.html. Accessed on November 17, 2024.

ATESTADO DE MATRÍCULA

Período Letivo:	**2024.2**	Nível:	**GRADUAÇÃO**
Matrícula:	**494223**	Vínculo:	**REGULAR**
Nome:	**LARISSA DA SILVA MATOS**		
Curso:	**ENGENHARIA DE COMPUTAÇÃO - MT**	Cidade:	**QUIXADÁ**
Formação:	**BACHARELADO**		

TURMAS MATRICULADAS: 4
ATIVIDADES MATRICULADAS: 1

Componentes Curriculares/Docentes	Status	Horário
QXD0176 - APRENDIZADO DE MÁQUINA CRISTON PEREIRA DE SOUZA **Tipo:** DISCIPLINA **Local:** Campus Quixadá **Ano/Periodo:** 2024.2 **Turma:** 01A **Duração da Turma:** (21/10/2024 a 07/03/2025)	MATRICULADO	QUI 15:30-17:30 SEX 15:30-17:30 (21/10/2024 - 07/03/2025)
QXD0232 - EDUCAÇÃO AMBIENTAL PAULO ARMANDO CAVALCANTE AGUILAR **Tipo:** DISCIPLINA **Local:** Campus Quixadá **Ano/Periodo:** 2024.2 **Turma:** 01A **Duração da Turma:** (21/10/2024 a 07/03/2025)	MATRICULADO	QUA 15:30-17:30 SEX 13:30-15:30 (21/10/2024 - 07/03/2025)
QXD0256 - INTERAÇÃO HUMANO-COMPUTADOR MARCELO MARTINS DA SILVA **Tipo:** DISCIPLINA **Local:** Campus Quixadá **Ano/Periodo:** 2024.2 **Turma:** 02A **Duração da Turma:** (21/10/2024 a 07/03/2025)	MATRICULADO	TER 10:00-12:00 QUA 10:00-12:00 (21/10/2024 - 07/03/2025)
QXD0225 - TÓPICOS ESPECIAIS EM TECNOLOGIA DE INFORMAÇÃO E COMUNICAÇÃO CRISTIANO BACELAR DE OLIVEIRA **Tipo:** DISCIPLINA **Local:** Campus Quixadá **Ano/Periodo:** 2024.2 **Turma:** 01A **Duração da Turma:** (21/10/2024 a 07/03/2025)	MATRICULADO	QUI 10:00-12:00 SEX 10:00-12:00 (21/10/2024 - 07/03/2025)
QXD0220 - TRABALHO DE CONCLUSÃO DE CURSO II **Tipo:** ATIVIDADE **Ano/Periodo:** 2024.2	MATRICULADO	--

Após os períodos de ajustes de matrículas, procure a coordenação do seu curso para confirmação de solicitações de matrícula em ATIVIDADES, ESTÁGIOS e TRABALHOS DE CONCLUSÃO DE CURSO.

ATENÇÃO

Para verificar a autenticidade deste documento acesse **http://www.si3.ufc.br/sigaa/documentos/** informando a matrícula, a data de emissão e o código de verificação **49a7cab64c**

SIGAA | Copyright © 2006-2024 - Superintendência de Tecnologia da Informação - UFC - (85) 3366-9999 - si3asprd02.ufc.br

ASSINADO DIGITALMENTE
WAGNER GUIMARAES AL ALAM
A conformidade com a assinatura pode ser verificada em:
http://serpro.gov.br/assinador-digital
SERPRO

Universidade Federal do Ceará
Sistema Integrado de Gestão de Atividades Acadêmicas

Emitido em 20/11/2024 13:04

Portal do Discente

Atestado de Matrícula

Período Letivo:	**2024.2**	Nível:	**GRADUAÇÃO**
Matrícula:	**472110**	Vínculo:	**REGULAR**
Nome:	**JOSE BATISTA DE SOUZA JUNIOR**		
Curso:	**ENGENHARIA DE COMPUTAÇÃO - MT**	Cidade:	**QUIXADÁ**
Formação:	**BACHARELADO**		

Turmas Matriculadas: 6
Atividades Matriculadas: 1

Componentes Curriculares/Docentes	Status	Horário
QXD0141 - INSTRUMENTAÇÃO LUIS RODOLFO REBOUÇAS COUTINHO **Tipo:** DISCIPLINA **Local:** Campus Quixadá **Ano/Periodo:** 2024.2 **Turma:** 01A **Duração da Turma:** (21/10/2024 a 07/03/2025)	MATRICULADO	QUA 13:30-15:30 QUI 13:30-15:30 (21/10/2024 - 07/03/2025)
QXD0110 - PROJETO DE PESQUISA CIENTÍFICA E TECNOLÓGICA EMANUEL FERREIRA COUTINHO **Tipo:** DISCIPLINA **Local:** Campus Quixadá **Ano/Periodo:** 2024.2 **Turma:** 01A **Duração da Turma:** (21/10/2024 a 07/03/2025)	MATRICULADO	QUA 08:00-10:00 (21/10/2024 - 07/03/2025)
QXD0126 - PSICOLOGIA E PERCEPÇÃO VALDEMIR PEREIRA DE QUEIROZ NETO **Tipo:** DISCIPLINA **Local:** Campus Quixadá **Ano/Periodo:** 2024.2 **Turma:** 01 **Duração da Turma:** (21/10/2024 a 07/03/2025)	MATRICULADO	SEG 08:00-10:00 TER 08:00-10:00 (21/10/2024 - 07/03/2025)
QXD0240 - ROBÓTICA I THIAGO WERLLEY BANDEIRA DA SILVA **Tipo:** DISCIPLINA **Local:** Campus Quixadá **Ano/Periodo:** 2024.2 **Turma:** 01A **Duração da Turma:** (21/10/2024 a 07/03/2025)	MATRICULADO	QUI 15:30-17:30 SEX 15:30-17:30 (21/10/2024 - 07/03/2025)
QXD0214 - SISTEMAS EMBARCADOS ANDRE RIBEIRO BRAGA **Tipo:** DISCIPLINA **Local:** Campus Quixadá **Ano/Periodo:** 2024.2 **Turma:** 01A **Duração da Turma:** (21/10/2024 a 07/03/2025)	MATRICULADO	SEG 15:30-17:30 TER 15:30-17:30 (21/10/2024 - 07/03/2025)
QXD0225 - TÓPICOS ESPECIAIS EM TECNOLOGIA DE INFORMAÇÃO E COMUNICAÇÃO CRISTIANO BACELAR DE OLIVEIRA **Tipo:** DISCIPLINA **Local:** Campus Quixadá **Ano/Periodo:** 2024.2 **Turma:** 01A **Duração da Turma:** (21/10/2024 a 07/03/2025)	MATRICULADO	QUI 10:00-12:00 SEX 10:00-12:00 (21/10/2024 - 07/03/2025)
QXD0219 - TRABALHO DE CONCLUSÃO DE CURSO I **Tipo:** ATIVIDADE **Ano/Periodo:** 2024.2	MATRICULADO	--

Após os períodos de ajustes de matrículas, procure a coordenação do seu curso para confirmação de solicitações de matrícula em ATIVIDADES, ESTÁGIOS e TRABALHOS DE CONCLUSÃO DE CURSO.

ATENÇÃO

Para verificar a autenticidade deste documento acesse **http://www.si3.ufc.br/sigaa/documentos/** informando a matrícula, a data de emissão e o código de verificação **f2c671fe68**

SIGAA | Copyright © 2006-2024 - Superintendência de Tecnologia da Informação - UFC - (85) 3366-9999 - si3asprd02.ufc.br

ASSINADO DIGITALMENTE
WAGNER GUIMARAES AL ALAM
A conformidade com a assinatura pode ser verificada em:
http://serpro.gov.br/assinador-digital
SERPRO

Universidade Federal do Ceará
PRÓ-REITORIA DE GRADUAÇÃO

07.272.636/0001-31 Campus Universitário - Reitoria, PICI
Fortaleza/CE - CEP: 60020-181 (85) 3366-9999 - Fax: (85) 3366-9999
- e-mail: dsi@sti.ufc.br

D E C L A R A Ç Ã O

Declaramos, para os fins a que se fizerem necessários, que **SAMUEL HENRIQUE GUIMARÃES ALENCAR** é aluno(a) desta universidade, sob o número **473360**, no curso de **ENGENHARIA DE COMPUTAÇÃO - QUIXADÁ - Presencial - MT - BACHARELADO - QUIXADÁ**.

Pró-Reitoria de Graduação da Universidade Federal do Ceará, em Fortaleza, 11 de Novembro de 2024.

ASSINADO DIGITALMENTE
WAGNER GUIMARAES AL ALAM
A conformidade com a assinatura pode ser verificada em:
http://serpro.gov.br/assinador-digital
SERPRO

Para verificar a autenticidade deste documento, escaneie o QR Code ou acesse **http://si3.ufc.br/sigaa/documentos** e preencha o formulário, informando o número de matrícula **473360**, a data de emissão **11/11/2024** e o código de verificação **b3cc62e33d**.

Federal University of Ceará
Office of the Vice Provost for Personnel Management
Coordinating Office for Personnel Management

Corporate Taxpayer Identification No: 07.272.636/0001-31

STATEMENT OF EMPLOYMENT

This is to certify that **WAGNER GUIMARAES AL ALAM** (employee registration no. **2186999**) holds the position of **HIGHER EDUCATION PROFESSOR** at the Federal University of Ceará. He was admitted to the **Quixadá Campus** on **23 January 2015**, in the municipality of **QUIXADÁ, Ceará, Brazil** (**full-time**, class **ADJUNCT PROFESSOR**, career level **603**). We further confirm that his employment hereunder is on an exclusive basis.

Fortaleza, 20 November 2024.

Av. da Universidade, 2853, Benfica
60020-181
Fortaleza – Ceará – Brazil

ASSINADO DIGITALMENTE
WAGNER GUIMARAES AL ALAM
A conformidade com a assinatura pode ser verificada em:
http://serpro.gov.br/assinador-digital
SERPRO

To verify the authenticity of this document, scan the QR Code or access **http://si3.ufc.br/sigaa/documentos**, and enter the enrollment number **2186999**, the date of issue **20/11/2024** and the verification code **191aa0b5ae**.