Democratic and Popular Algeria

Ministry of Higher Education and Scientific Research
Constantine 2 University – AbdelHamidMehri

Faculty of New Information and Communication Technologies

Department of Software Technologies and Information Systems

Option : Data Science and Intelligent Systems

Theme

Smart Home IoT

Directed by :

Derradji Aymene Badreddine

Boumakh Mohamed

Haba Doua Kamar Ezzamane

Goutal Wafa

Directed By :

Dr.Blegidoum merime

6 janvier 2025

List of Figures

Introduction

The smart home project aims to integrate automated systems to improve the comfort, security and energy efficiency of homes. By using technologies such as sensors and connected devices, it allows various aspects of the home to be controlled remotely, while meeting the specific needs of the occupants. This project is not limited to optimizing daily life, but also contributes to reducing the ecological footprint by promoting responsible energy consumption. By addressing technological challenges and data security issues, this initiative represents a step towards a more sustainable and harmonious way of life, where technology and humans coexist to create smart living spaces.

The goal

The goal of the smart home project is to develop an integrated system that improves the functionality and security of the home using advanced technologies. This project focuses on automating various aspects of the home, such as opening and closing garage doors, monitoring room temperature. By integrating sensors and smart devices, the goal is to create a responsive environment that adapts to the needs of the occupants, while ensuring increased security through monitoring and alert systems. In addition, the project aims to raise awareness among users about the importance of sustainability and energy efficiency

Implémentation issues

The smart home project aims to demonstrate how IoT technologies can be used to improve security, energy management, and environmental monitoring. Using Packet Tracer to simulate the network infrastructure and Wokwi to simulate the IoT sensors, a complete system can be created. The collected data will be transmitted to AWS IoT Core for storage (DynamoDB), analysis, and visualization.

Overview of the Tools That Will Be Used

Packet Tracer

Packet Tracer will be used to model the network architecture of the smart home, including IoT sensors, gateways, and control devices. It will enable the simulation of communications between different devices and test connectivity.

Wokwi

Wokwi is an online platform that allows the simulation of electronic projects, particularly those using microcontrollers like the ESP32. It provides an environment to test and visualize code and hardware.

AWS IoT Core

AWS IoT Core is a cloud service that allows IoT devices to connect and communicate with applications. Sensor data is collected using the MQTT protocol, a lightweight publish/subscribe protocol, enabling devices

to send messages to AWS IoT Core securely and efficiently.

ThingSpeak

ThingSpeak is an open-source IoT platform that allows for the collection, storage, and visualization of sensor data. It offers tools for data analysis and real-time graph creation.

mobile app

dedicated mobile app that allows users to control and monitor their smart homes from their smartphones

Implementation Steps

1.Simulate Sensor Network and Devices

1.1 communication with objects

shows a network simulation in Cisco Packet Tracer, with various IoT devices connected in a smart home environment. Here is an overview of what you can see in this topology : Home Gateway (LTE 100) : A central gateway connecting all IoT sensors and devices to the Internet or the cloud.

IoT Sensors :

- Home Gateway
- Lawn Sprinkler (automatic sprinkler).
- Smart LED (IoT10) for lighting.
- Temperature Sensor (IoT15).
- Web Camera (IoT12).
- Street Lamp (IoT9) and other home automation devices.

Control Devices :

- A tablet (TabletPC-PT) to manage the systems.
- A garage door and automated windows.

FIGURE 1 – Smart Home IoT Network Simulation

1.2 configuration Some Sensors

1.2.1 configuration Home Gateway

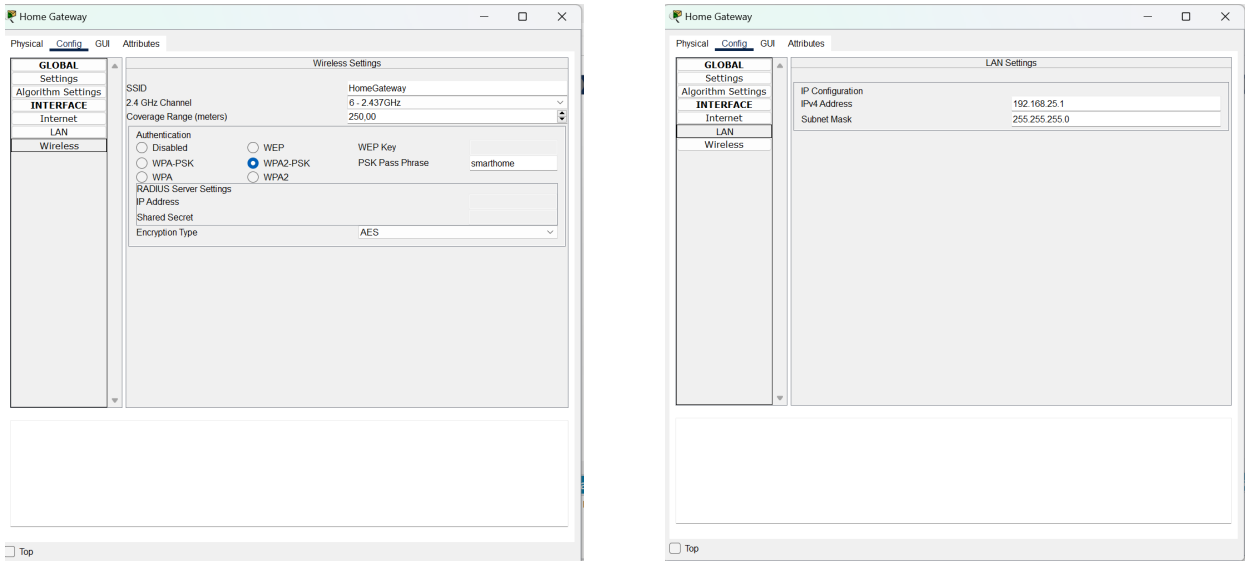


FIGURE 2 – configuration home Gateway

The configuration of the Home Gateway for a smart home project is essential to ensure network connectivity and secure communications between the different devices. Here are the main configurations carried out :

LAN configuration :

IPv4 address : 192.168.25.1, defining the router as the main gateway for devices connected to the local network.

Subnet mask : 255.255.255.0, allowing to define a subnet capable of managing up to **254** devices.

Wireless configuration :

SSID : HomeGateway, identifying the Wi-Fi network for connected devices.

2.4 GHz channel : Channel 6 (2.437 GHz) with a range of 250 meters, ensuring adequate coverage for devices in the home.

Authentication : WPA2-PSK with a secure "smarthome" passphrase, offering advanced protection against unauthorized access. Encryption type : AES, ensuring enhanced security for data exchanged over the network.

1.2.2 connect garage to home Gateway

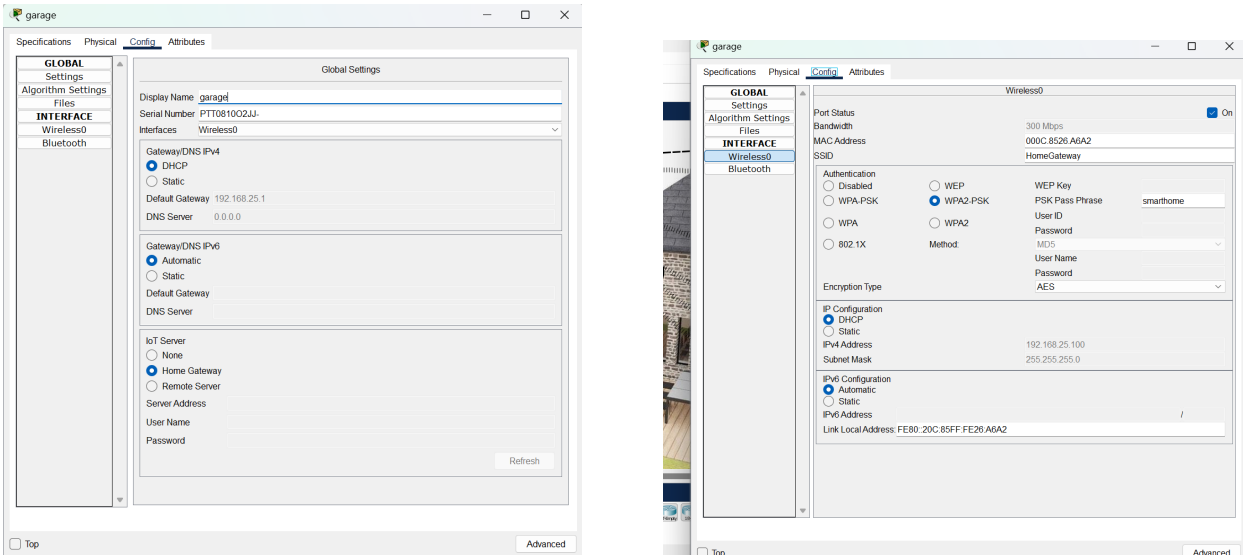


FIGURE 3 – connect garage to home Gateway

To connect the garage device to the HomeGateway network, configure the Wi-Fi interface by selecting the HomeGateway SSID with WPA2-PSK authentication and enter the smarthome security key. Make sure the encryption is set to AES to secure the connection. Enable IP configuration in DHCP mode to obtain an IP address automatically or configure a static address if necessary (e.g. **192.168.25.100** with a mask of **255.255.255.0**). For IPv6, choose Automatic or set a manual address if required. Finally, in the global settings, select Home Gateway as the IoT server and verify that the Wireless0 interface is enabled. Test the connection to confirm access to the network and the server. **192.168.25.100**

Note :

all devices are configured with the same method

1.3 communication with cloud

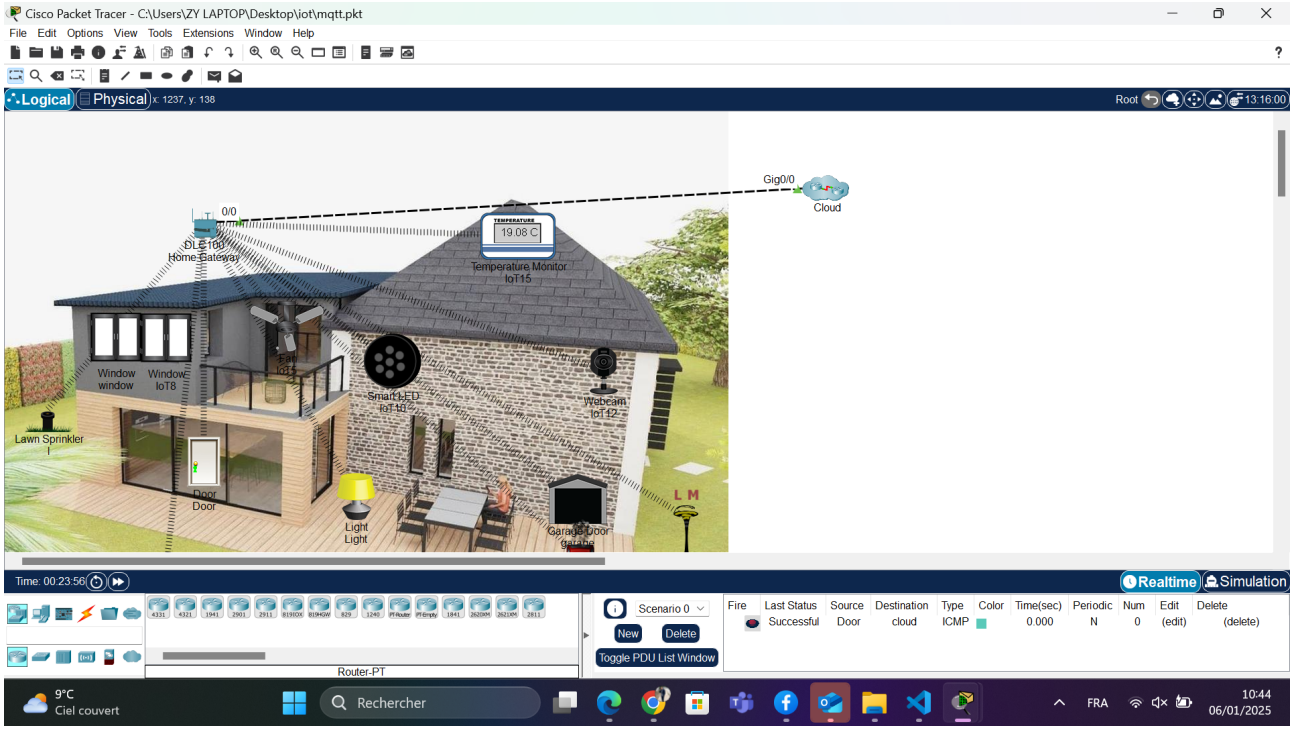


FIGURE 4 – Smart Home IoT Network Simulation connect with cloud

To configure a router for a distant connection via the cloud, start by activating and configuring the GigabitEthernet0/0 interface with the address IP 209.1.113.1 and the sous-réseau mask **255.255.255.0**. Check if the interface is active and functional in automatic mode for the watch and the duplex. Ensuite, use a route statistic to divert the traffic to the distant route **192.168.25.0/24** at the specific address of the driver (Next Hop) as **192.168.25.1.** This will ensure that all traffic due to this sous-Réseau will be corrected. These configurations allow the router to connect the local network to the distant network via the cloud. **209.1.113.1**

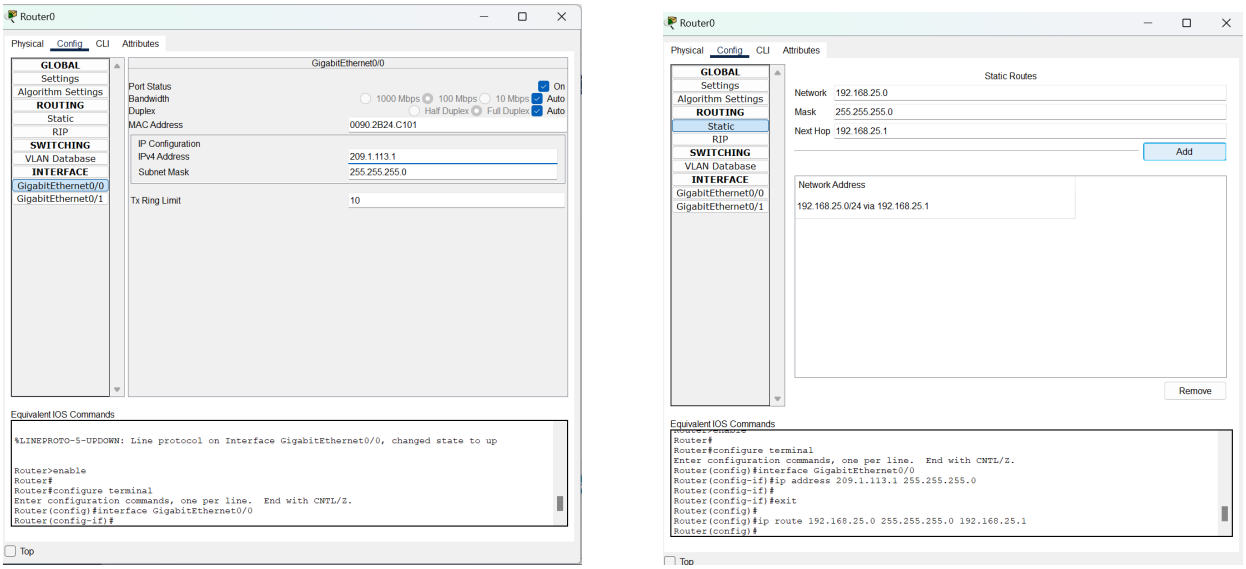


FIGURE 5 – configuration cloud

2.Simulation des Capteurs avec Wokwi

Use Wokwi to simulate an interactive control system integrating multiple devices such as temperature, humidity (DHT22) and motion (PIR) sensors. Wokwi allows you to program an ESP32 or Arduino microcontroller to manage the interactions between a matrix keypad, an LCD screen, a servo motor, LEDs and a buzzer. Using this simulation, code can be written to secure access with a password, display real-time sensor data on the LCD screen, and trigger actions such as opening or closing a door based on user data or interactions

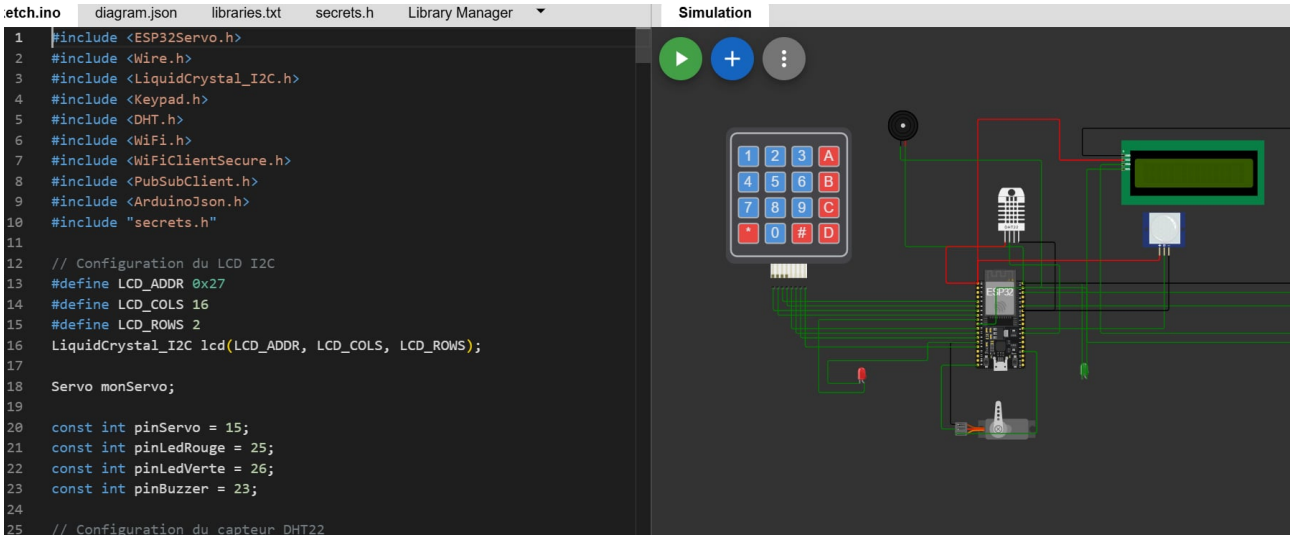


FIGURE 6 – Smelt DS Chapters Afk and Qawi

2.1 Features developed by wokwi code

2.1.1 Security with a password

The system is secured by a four-digit password that the user must enter via a 4x4 matrix keypad. This feature was implemented to limit access to critical actions, such as the simulated door opening by a servo motor. The user enters the password on the matrix keypad. Each digit is displayed in real time on the LCD screen. If the entered password matches the defined password **(default : "1111")**, access is granted and the door opens

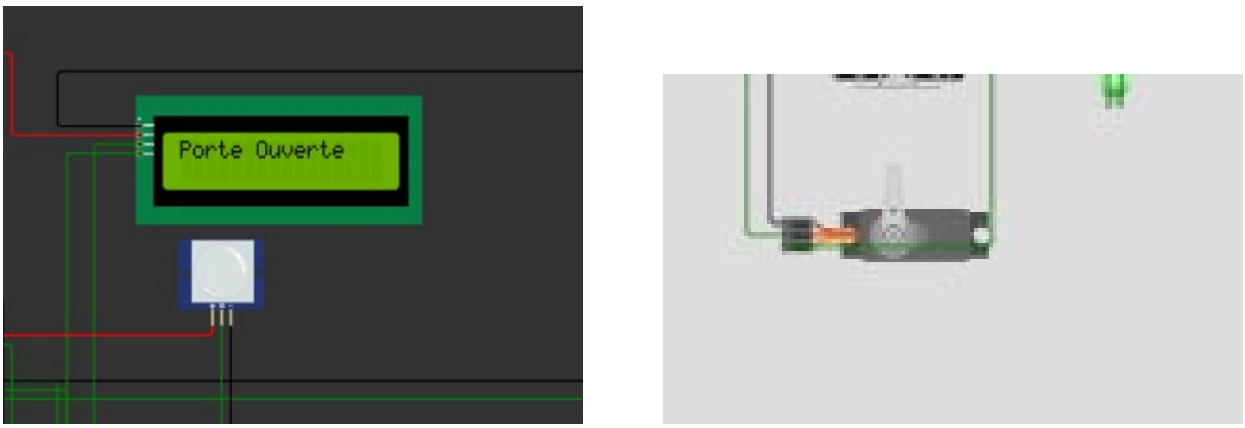


FIGURE 7 – Door simulation

If the password is incorrect, an alert is displayed on the LCD screen and an audible signal is emitted via the buzzer.

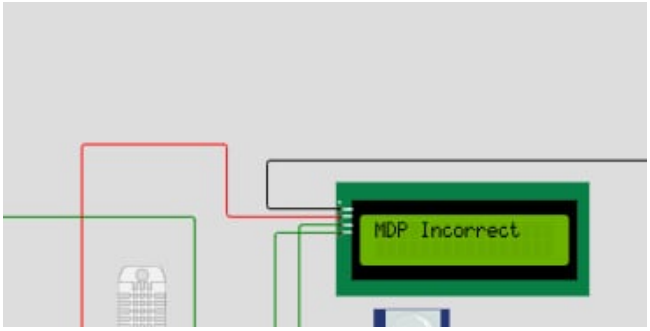


FIGURE 8 – mot de passe

2.1.2 Temperature and Humidity

In this project, the temperature and humidity measurement is done using a DHT22 sensor, which is a digital sensor that can provide data on ambient temperature and humidity. This data is then displayed on an I2C LCD screen for the user to view in real time

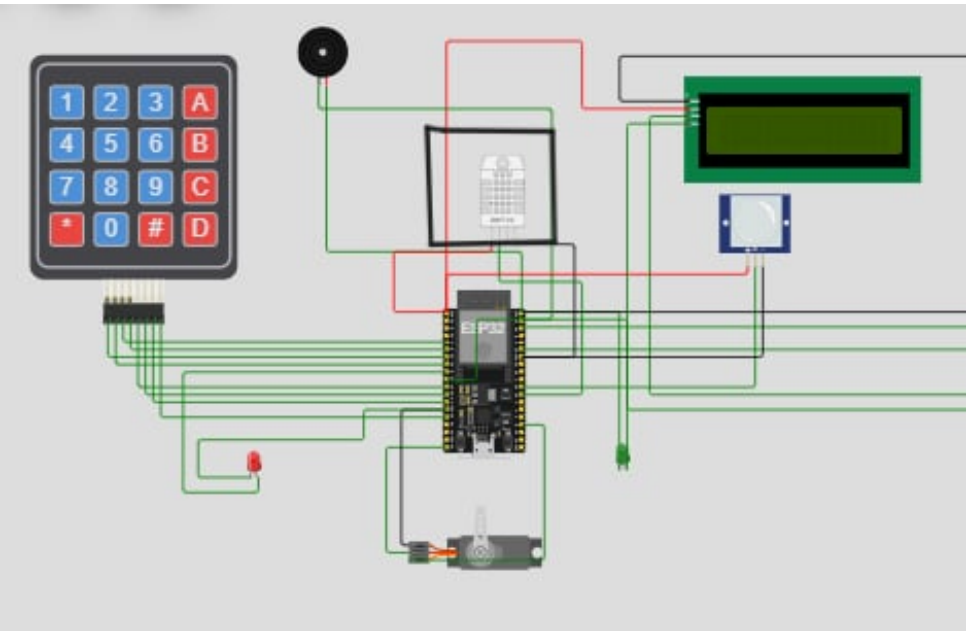


FIGURE 9 – Temperature and humidity simulation

If the sensor readings are correct, the temperature and humidity are displayed on two lines of the LCD :

— The first line shows the temperature in degrees Celsius

— The second line shows the humidity in percentages.

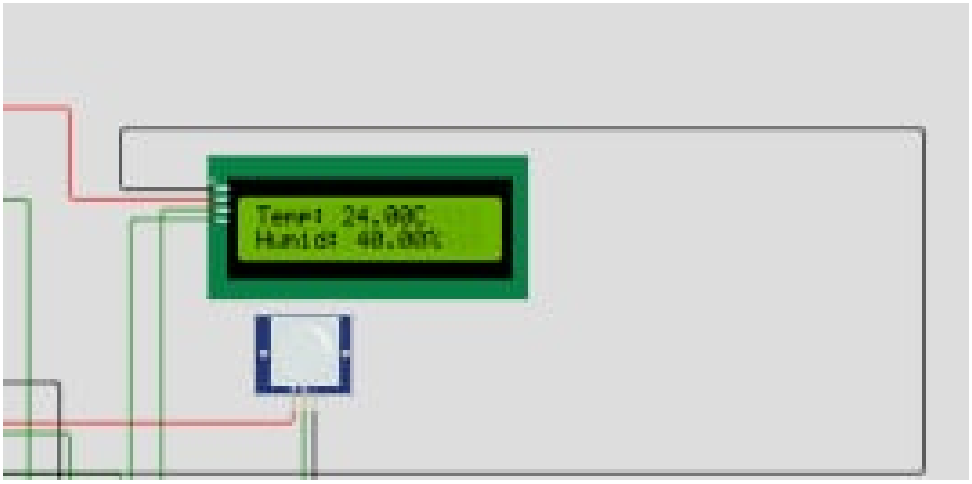


FIGURE 10 – Temperature and humidity

2.1.3 Motion Detection

Motion detection is achieved using a PIR (Passive Infrared Sensor), which can detect movement in a defined area. The PIR sensor senses infrared variations (heat) emitted by moving objects (such as a human) and sends a signal when motion is detected. This feature is used to indicate if someone is present in the monitored area, which can be integrated with other security or home automation systems

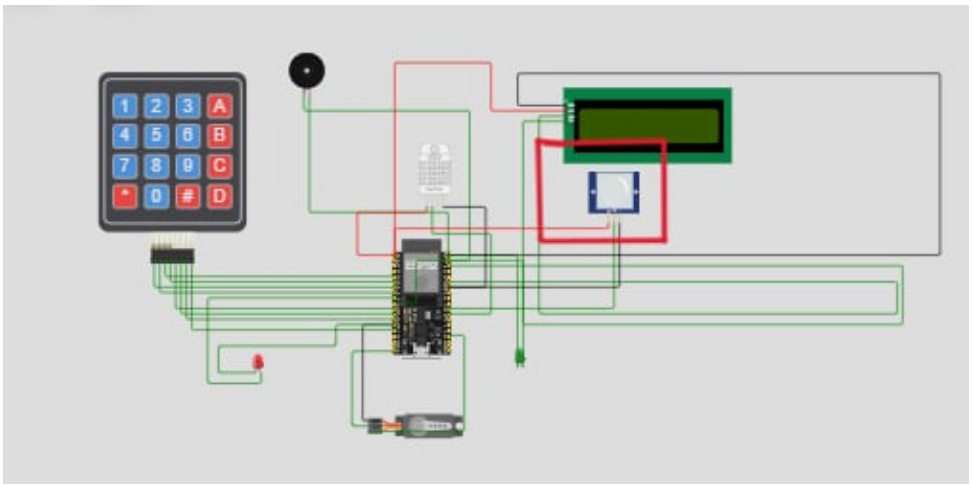


FIGURE 11 – Motion Detection

The LCD display is used to inform the user of the status of the motion detection. When the motion detection mode is activated (by the 'C' key on the matrix keyboard), the display shows :

— "Motion detected" if the sensor detects a presence.

— "No motion" if no infrared variation is detected.

Once the information is displayed, the system waits 2 seconds before returning to the main screen with the password prompt.

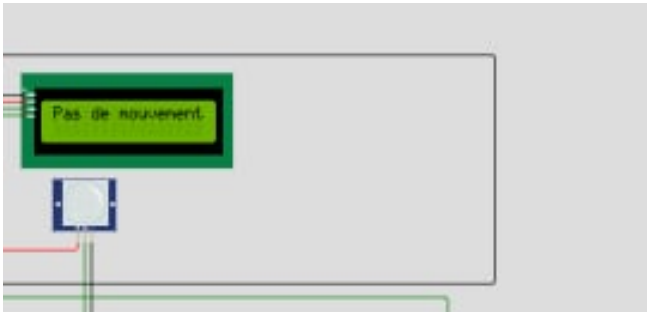


FIGURE 12 – motion detected" i

3.Collect and Transmit Data with MQTT and AWS IoT Core Configuration

3.1 create things

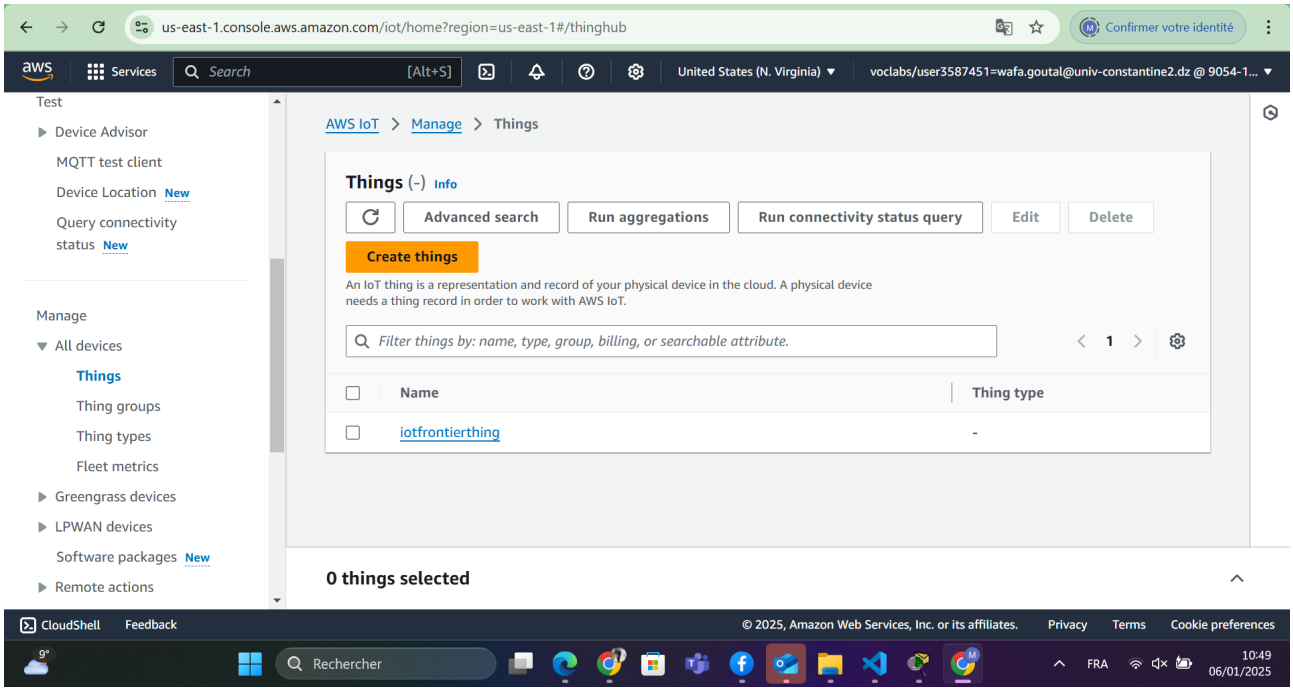


FIGURE 13 – Create Thing

Create a Thing : Click on the "Create Things" button.

Provide Details : You'll need to provide information about the new device, such as its name, type, and potentially other attributes.

Configure Thing : Once created, you can configure the thing by defining its attributes, shadow (which stores the current state of the device), and other settings.

Connect the Device : Establish a connection between your physical device and the AWS IoT cloud. This typically involves using protocols like MQTT or HTTPS.

Send Data : Your device will then be able to send data to the AWS IoT cloud using the established connection. This data can be anything relevant to the device's operation.

3.2 Create a certificate and key for the device

In the "Security" section of the AWS IoT Console, you can create the credentials required for securely connecting your IoT device to AWS IoT Core. This involves generating an X.509 certificate, a private key, and downloading the CA (Certificate Authority) file. These files must be downloaded immediately and stored securely on your PC, as the private key cannot be retrieved later,These security files are then loaded onto your device and referenced in the code to establish a secure connection using protocols like MQTT.

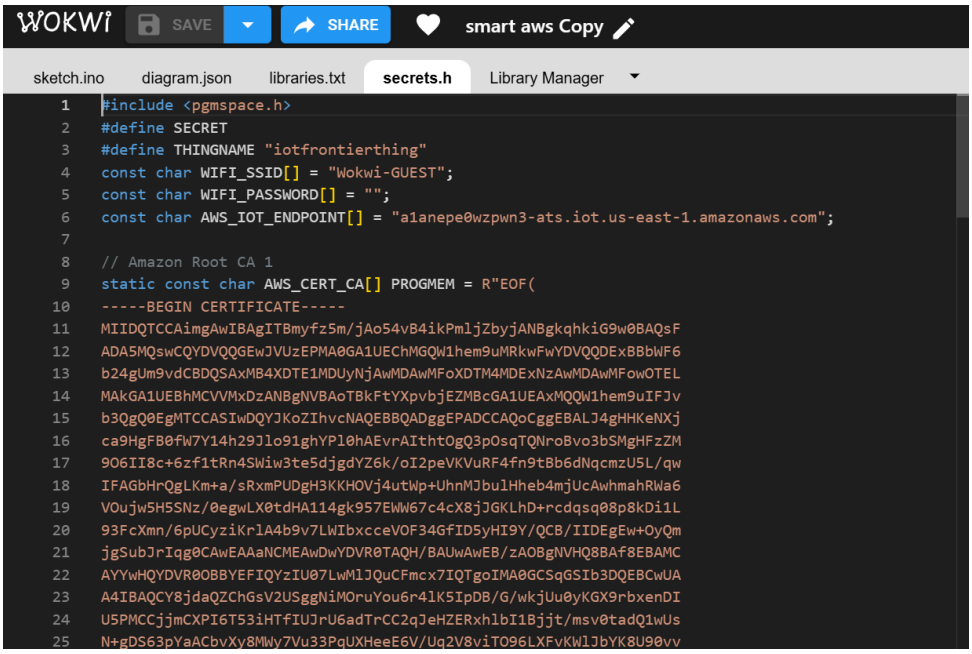


FIGURE 14 – Security-h

3.3 Send Data

Since the data has already been sent, you can verify it through the “MQTT Test Client” section in the AWS IoT Core console.

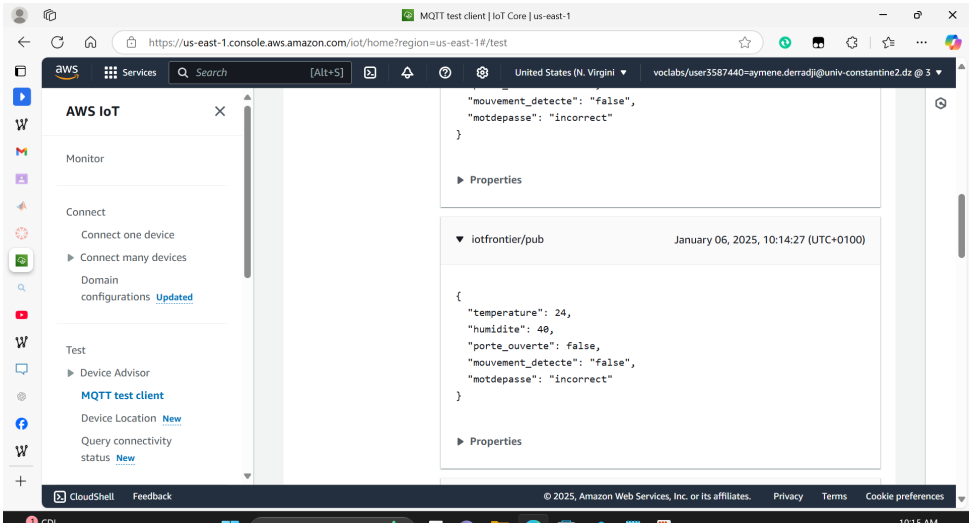


FIGURE 15 – Data sent

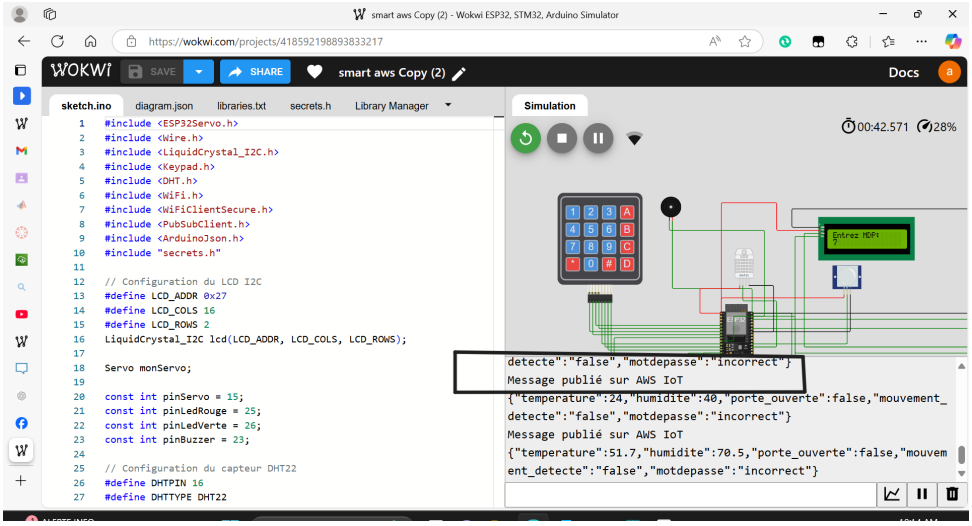


FIGURE 16 – message publish on our aws

4. Storage of data in dynamoDB

In this stage of our work, we aimed to demonstrate how data collected from IoT sensors can be effectively stored and visualized using AWS services. Storing data in DynamoDB ensures centralized, secure, and scalable management of data, while visualizing the data in ThinkSpeak provides a user-friendly interface for real-time analysis. These steps are crucial for building a reliable and functional IoT solution.

4.1 Reasons for Choosing DynamoDB

— High Performance and Scalability : DynamoDB is built to handle large volumes of data with low latency, making it ideal for IoT projects.

— Seamless Integration with AWS IoT : DynamoDB integrates perfectly with AWS IoT Core using IoT rules, enabling automated data storage.

— Flexible NoSQL Model : Its schema-less structure allows easy management and analysis of diverse data.

— Reliability and Security : DynamoDB ensures high availability and offers robust access controls for data security.

4.2 Launching the Wokwi Simulator

The Wokwi simulator was configured with a DHT22 sensor to generate temperature and humidity data.

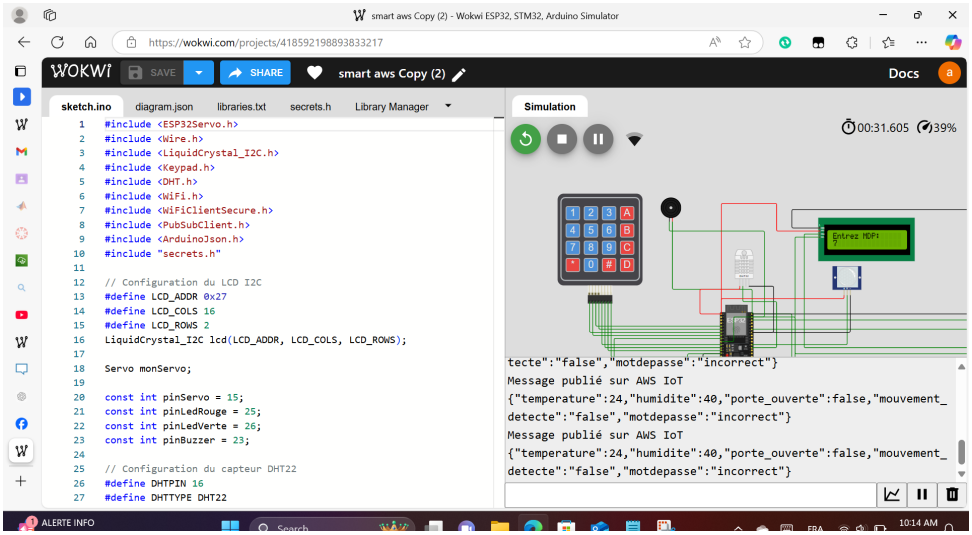


FIGURE 17 – data publish

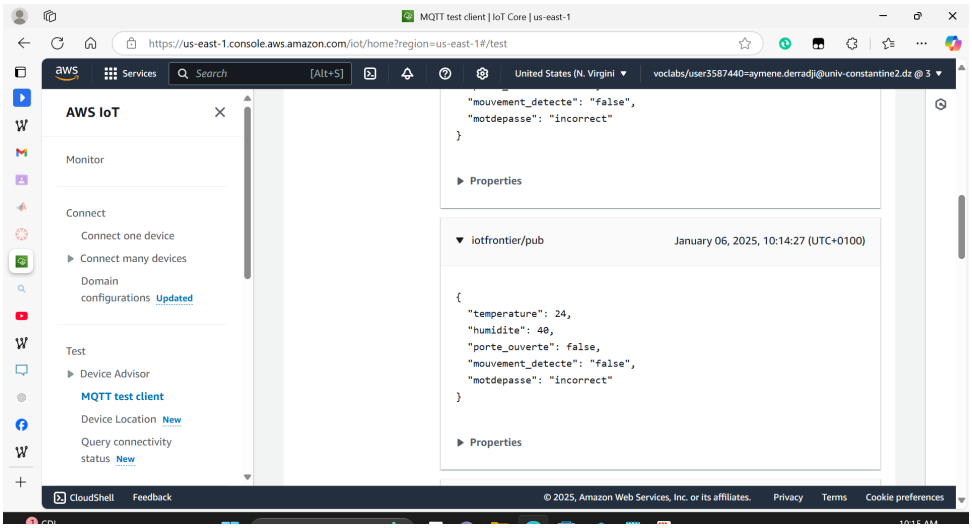


FIGURE 18 – data publish

These data points were sent to the AWS IoT Core MQTT broker for processing.

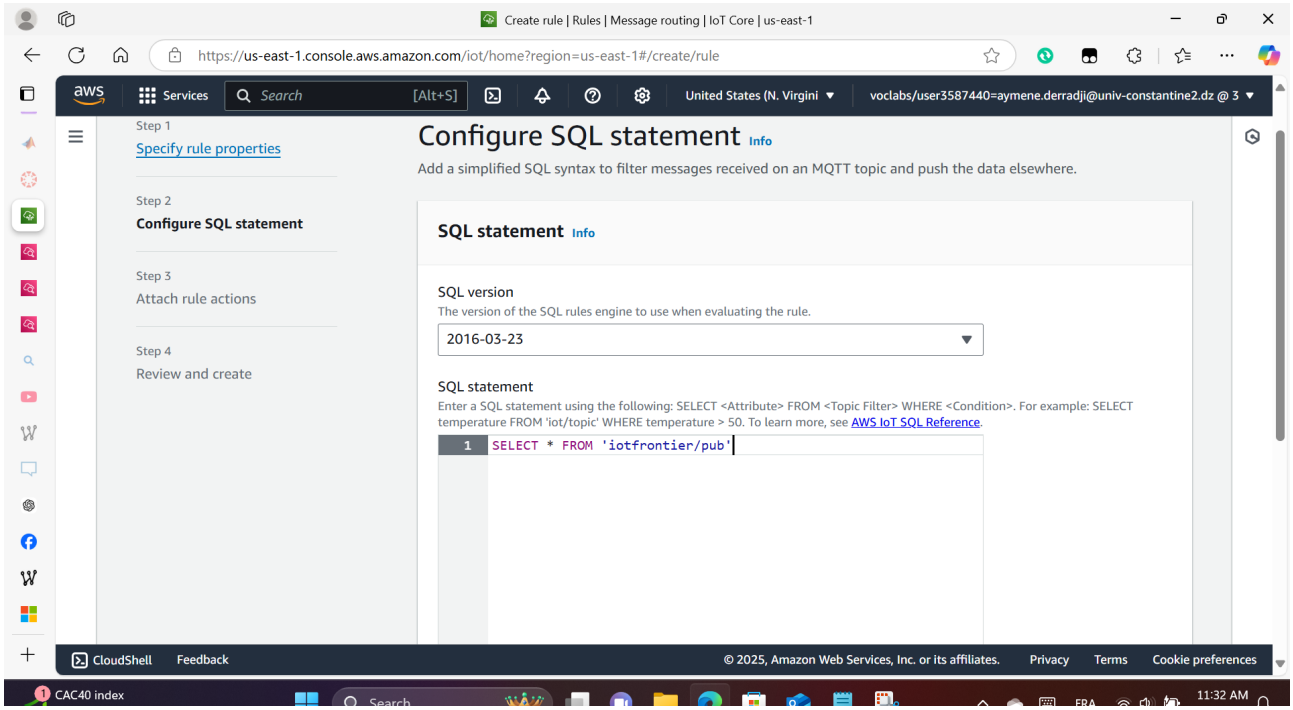
4.3 Validating Data in the MQTT Test Client

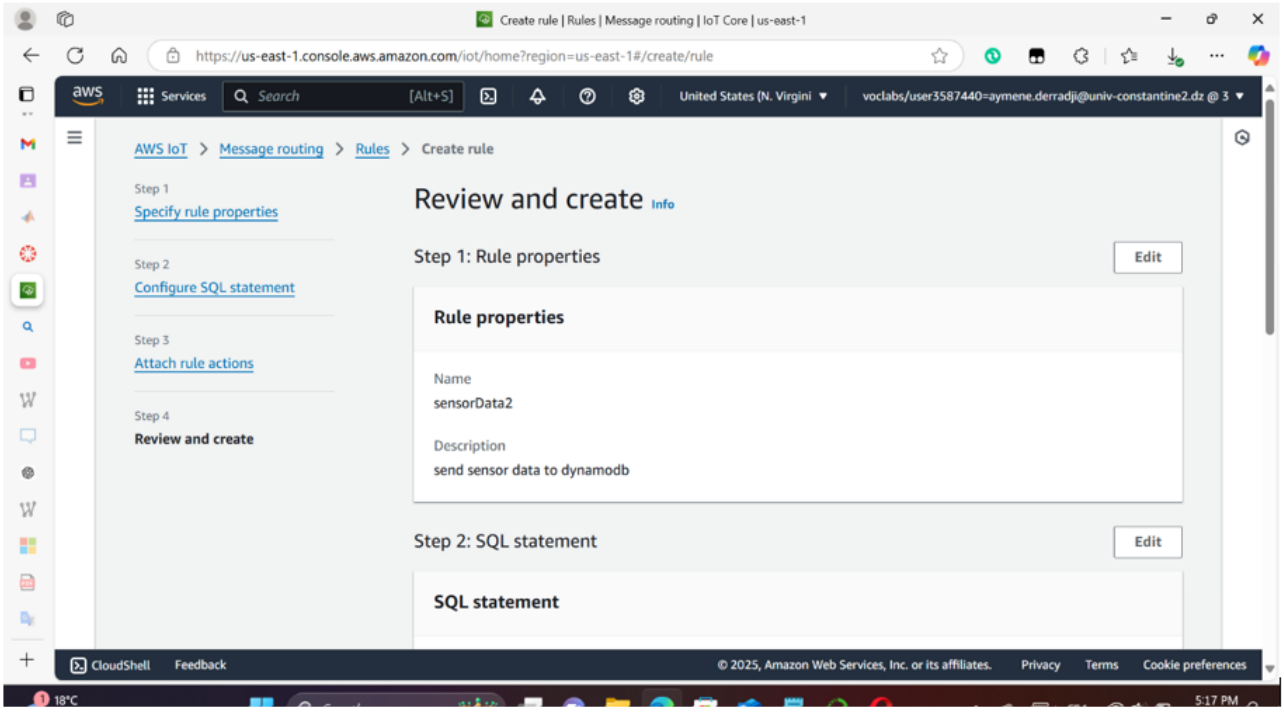
Once the simulator was running, the data was validated using the AWS IoT Core MQTT test client to ensure the data was being received correctly.

4.4 Creating an IoT Rule in AWS IoT Core

An IoT rule named SensorData2 was created in AWS IoT Core to process the incoming data. The SQL statement used in the rule configuration was :

SELECT * FROM 'iotfrontier/pub'

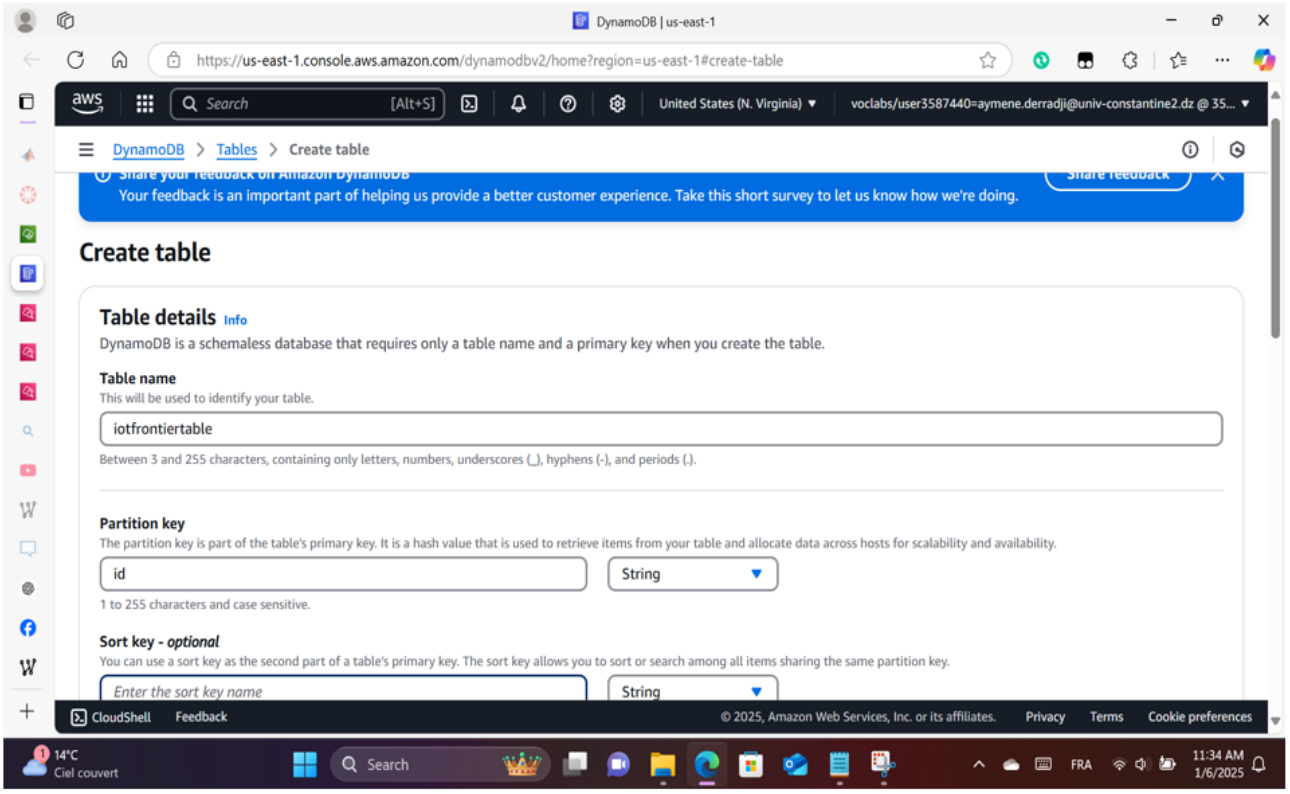


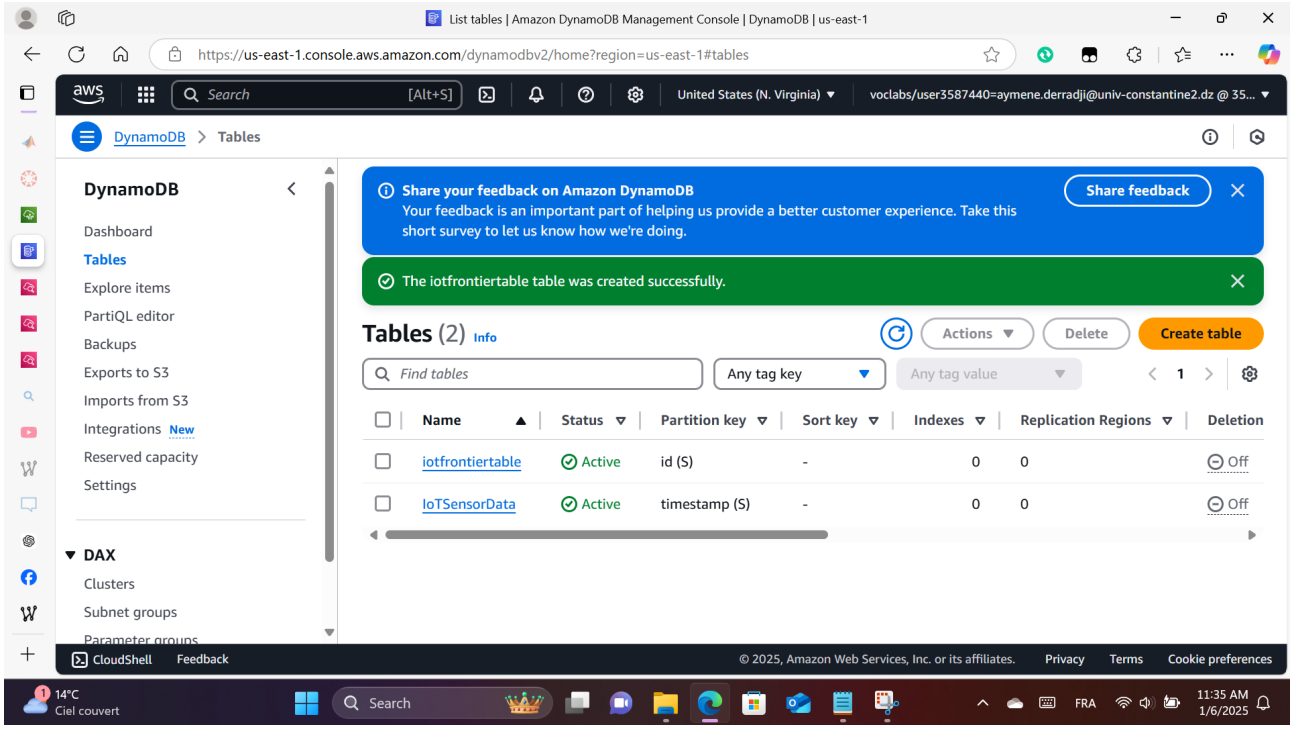


This SQL statement captures all data published on the MQTT topic iotfrontier/pub.

4.5 Creating a DynamoDB Table

A DynamoDB table named iotfrontiertable was created to store the data. The table schema included attributes to store temperature and humidity values, along with a unique identifier for each record.

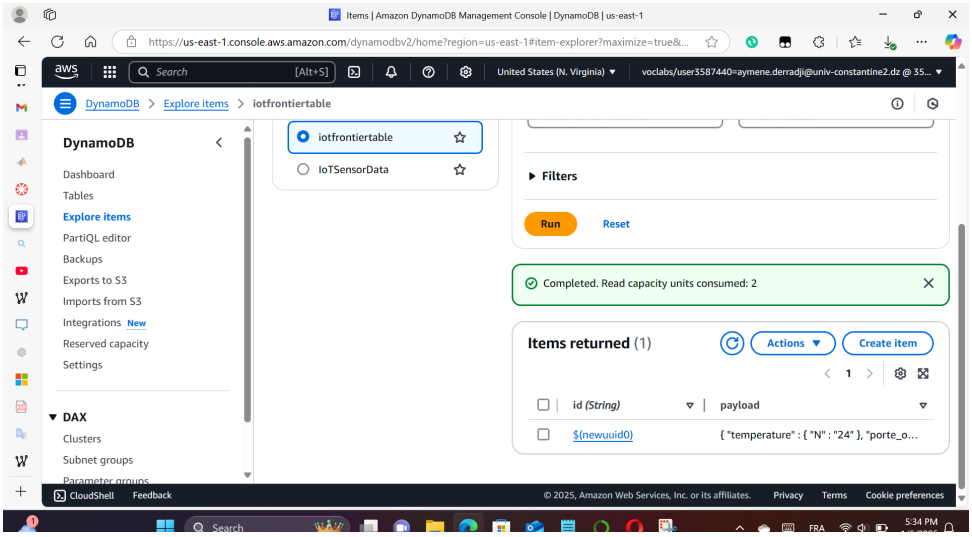




4.6 Configuring Rule Actions

The IoT rule SensorData2 was configured to route the data to the DynamoDB table. Table details, including the table name and necessary permissions, were set up in the rule actions.

4.7 Verifying Data Storage After configuring the setup, the iotfrontiertable was explored in the DynamoDB console. The data generated by Wokwi, such as temperature and humidity, was successfully stored and displayed in the table, as evidenced by the attached screenshots.

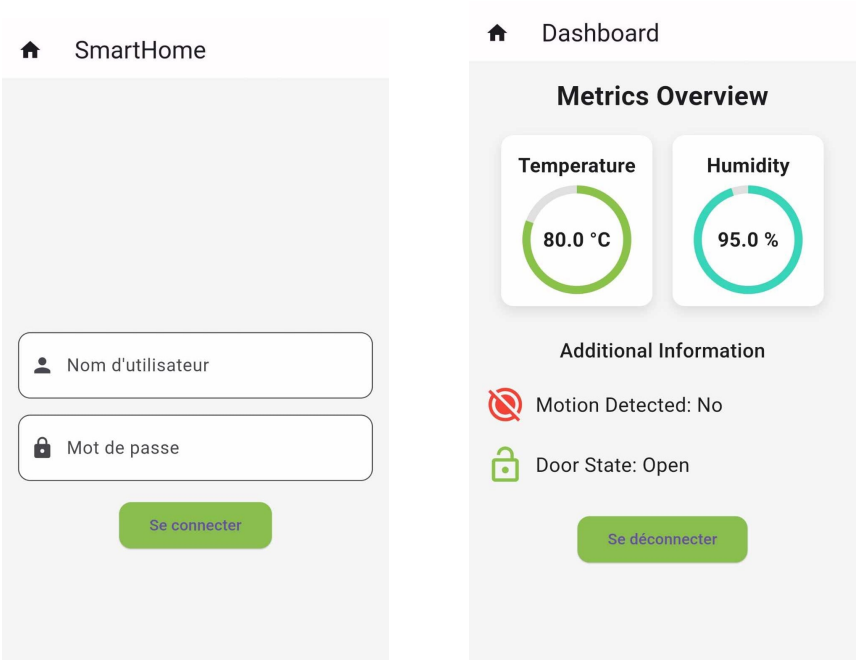


5. Visualization

Data visualization is the process of representing complex information in interactive graphs or charts to make it more understandable. In this project, parameters such as temperature, humidity, detected motion, and door status (open or closed). These visualizations can include curves to track changes over time, gauges to indicate critical thresholds, or real-time graphs for events such as door movement or opening.

5.1 visualization with mobile application

This mobile application integrates a login system to secure access to IoT data. Users must log in with their credentials to ensure that only authorized people can access the information. Once logged in, the application allows real-time visualization of data such as temperature, humidity, movement, and door status. The data is sent directly from IoT Core to the mobile application via HTTP requests. This information is then displayed in the form of clear and interactive graphs for easy interpretation. In addition, the application can generate real-time notifications to alert users in the event of critical events (e.g. an open door or detected movement). This approach ensures efficient and secure monitoring by leveraging the communication between IoT Core and the application via HTTP.

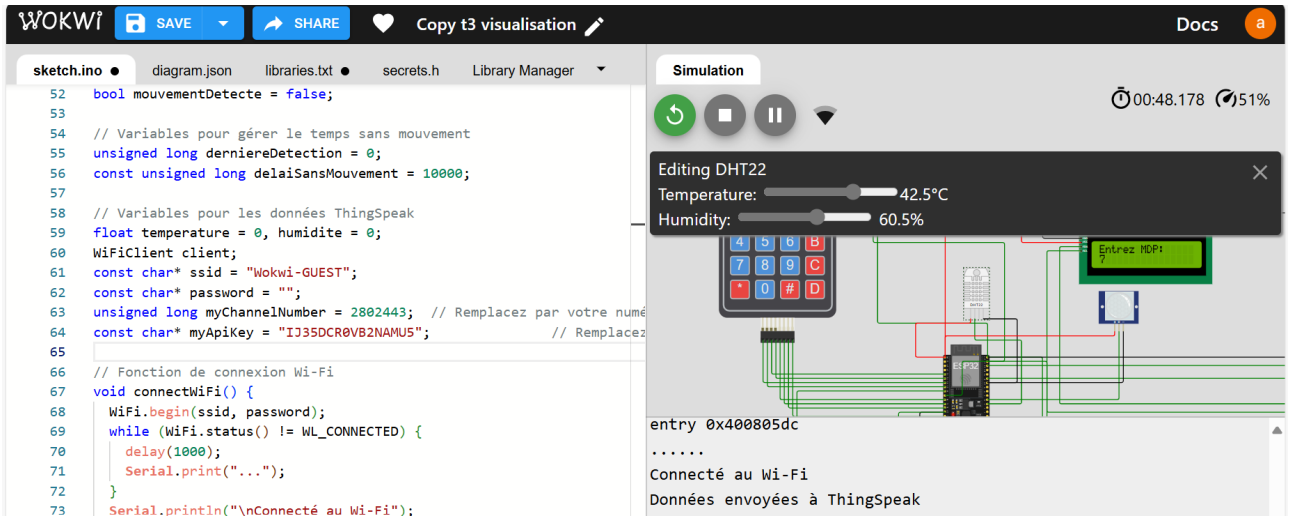


5.2 Visualisation using ThinkSpeak

The visualization was achieved using ThinkSpeak, a platform designed for IoT data monitoring and analysis. The goal was to ensure that changes in temperature and humidity values in the Wokwi interface are accurately reflected in ThinkSpeak's graphs and gauges, providing a real-time interactive visualization.

5.2.1 Steps to Achieve Data Visualization

Step 1 : Configuring Wokwi for Data Generation The Wokwi simulator was used to emulate an IoT environment with a connected DHT22 sensor generating temperature and humidity data. Changes to the sensor values (e.g., setting the temperature to 60°C or adjusting humidity) were instantly reflected in the simulation. The generated data was transmitted in real-time to AWS IoT Core via the MQTT protocol, specifically publishing to the MQTT topic iotfrontier/pub. The simulation was continuously monitored to ensure proper data publishing and real-time updates.

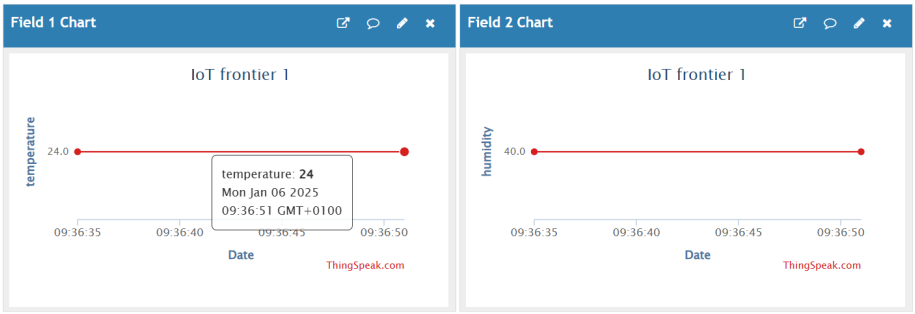


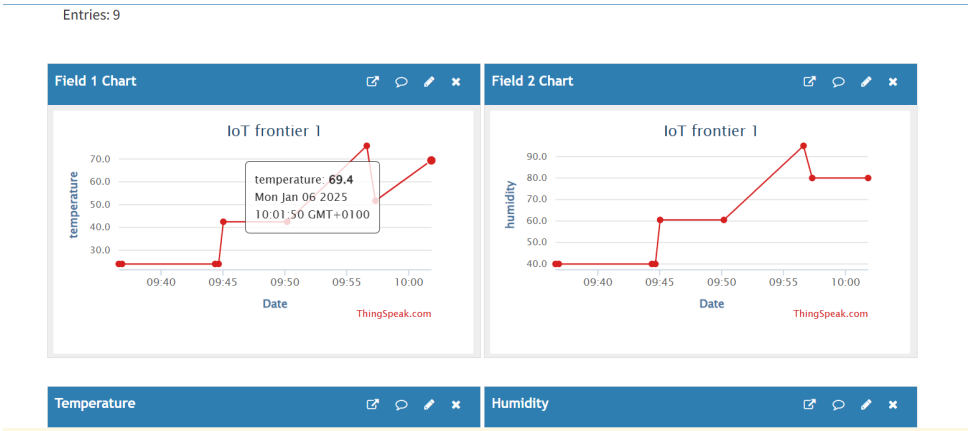
Step 2 : Sending Data to ThinkSpeak ThinkSpeak Channel Configuration

A ThinkSpeak channel was created with the following fields : Field 1 : Temperature (°C) Field 2 : Humidity (API keys (write and read) were generated to securely communicate with the ThinkSpeak server. Integration of AWS IoT Core with ThinkSpeak

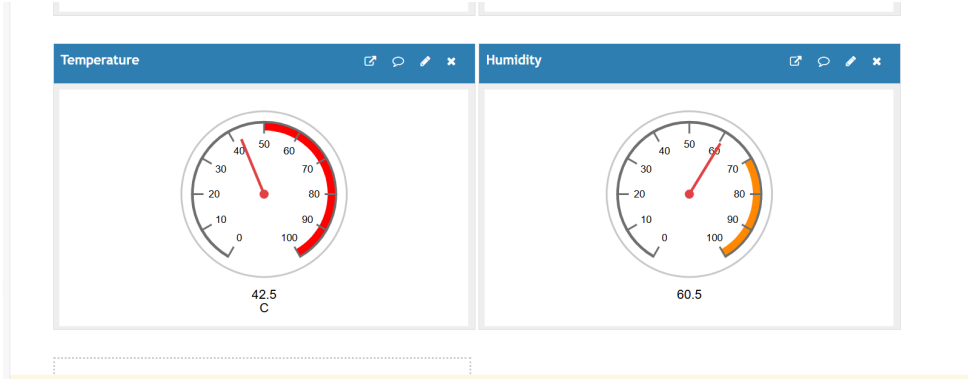
Data from the Wokwi simulator was routed through AWS IoT Core, formatted appropriately, and then sent to ThinkSpeak using HTTP requests or MQTT. A script or rule action was configured to ensure seamless data transmission.

Step 3 : Real-Time Visualization on ThinkSpeak Interactive Dashboard A custom dashboard was created in ThinkSpeak to display the incoming data in real-time. Graphical Components : Line Graphs : These were used to plot temperature and humidity over time, showing their fluctuations dynamically



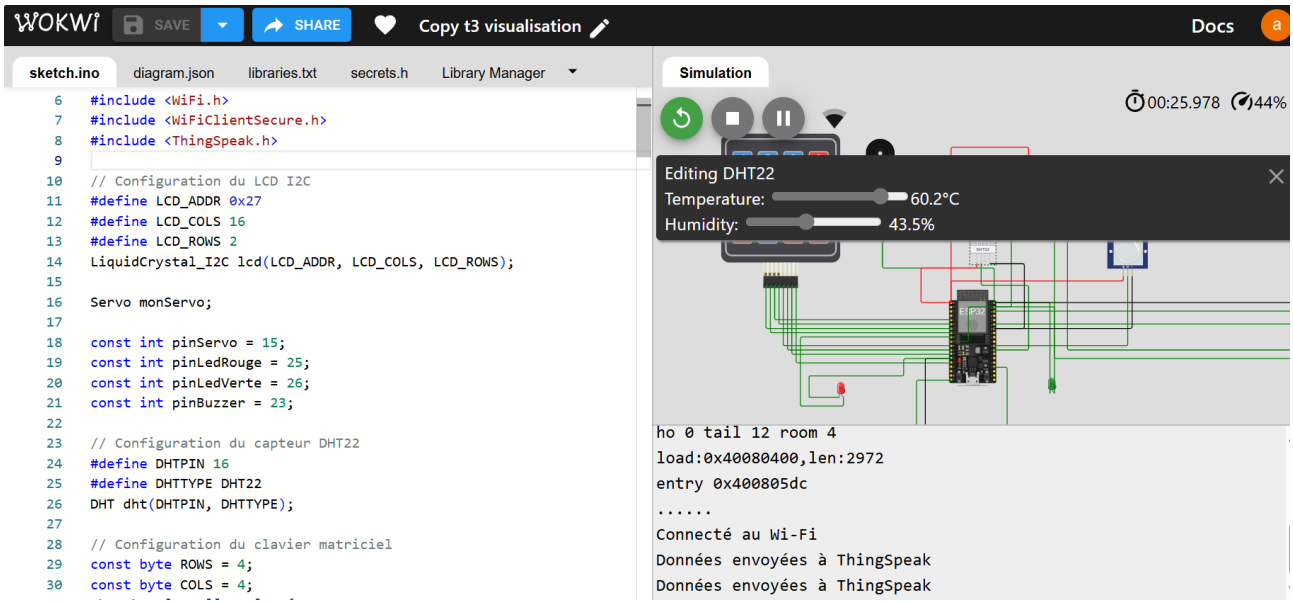


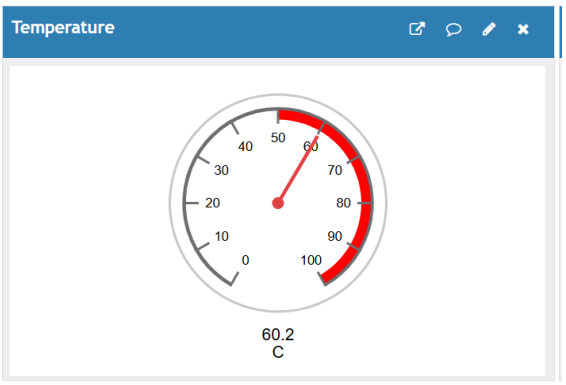
Gauges : Real-time gauge widgets displayed the current temperature and humidity values, making the data more interpretable.

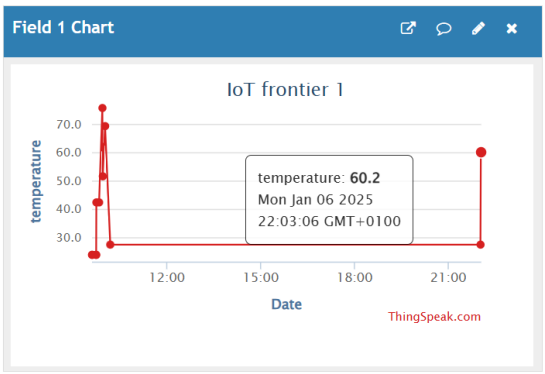


Testing Real-Time Changes

To test the setup : The temperature in the Wokwi DHT22 sensor was increased to 60°C and the humidity value was adjusted. These changes were immediately reflected in the ThinkSpeak graphs and gauges, confirming the successful real-time data transmission and visualization.







Results and Observations Accuracy of Data

The changes made in the Wokwi simulator (e.g., altering temperature to 60°C or humidity values) were accurately displayed in ThinkSpeak's graphs and gauges without delays. Real-Time Interaction

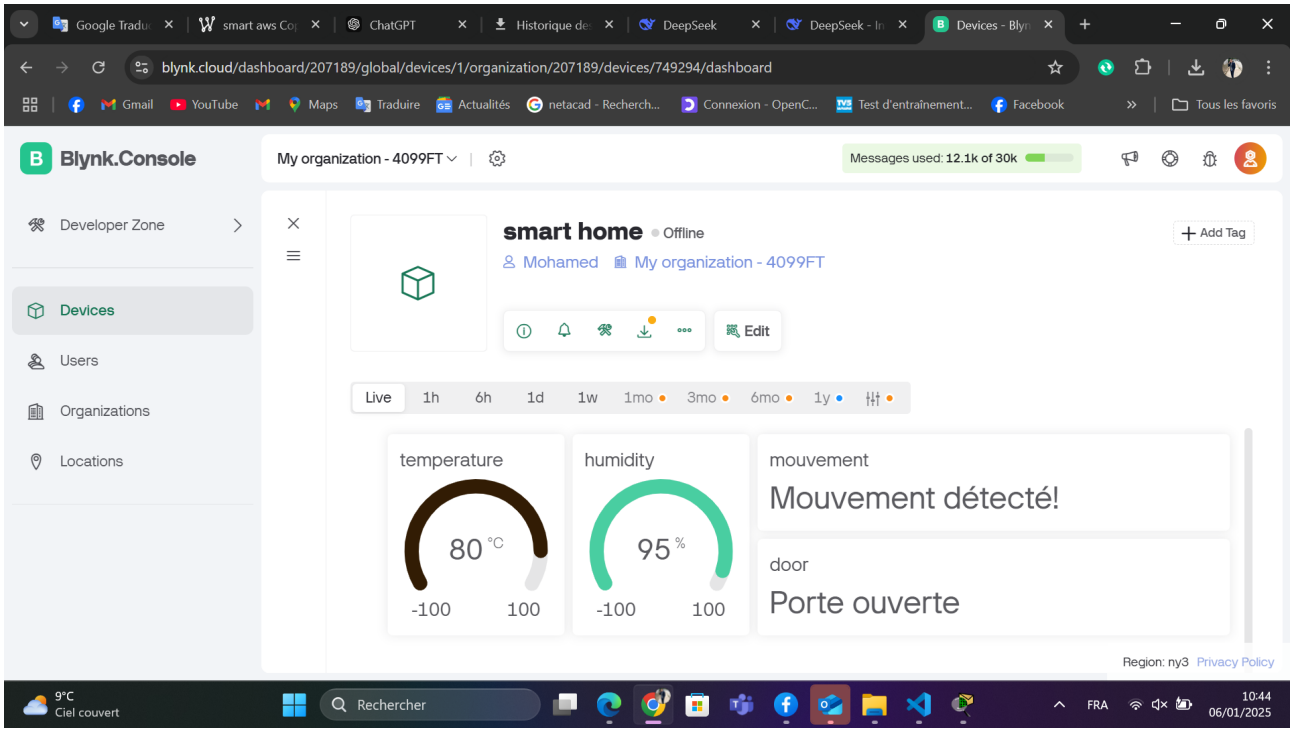
ThinkSpeak's real-time visualization allowed for immediate observation of environmental changes simulated in Wokwi. User-Friendly Interface

The ThinkSpeak interface, with its intuitive graphs and gauges, provided an easy-to-understand visualization of sensor data.

5.3 visualization with Blynk

To visualize the temperature, humidity and movement on the Blink on the Wokwi part, start by creating a projector on the Wokwi and simulating an ESP32 with capteurs, such as DHT for the temperature and humidification, and a PIR capteur for the movement. Ensuite, configure a projector on the Blynk application,

including widgets to monitor these données at real times. Click on the code to enter the Blynk library, configure the capteur brochures and send donnés from the capteurs to the Blynk application via Wi-Fi or through a communication module. The code uses a Blynk authentication token to set the connection. After you receive the code and connect your projector to the application, you can view the temperature values, humidity and movement of the Blink widgets at real temperatures



conclusion

The Smart Home IoT project successfully demonstrates how modern technologies like IoT, cloud computing, and data visualization tools can create more efficient, secure, and responsive living environments. By integrating sensors, emulators like Wokwi, and platforms such as AWS IoT Core, the project highlights the potential of automating daily tasks, improving energy efficiency, and ensuring environmental monitoring. Real-time data was effectively collected, transmitted, and stored using MQTT and DynamoDB, while visualization platforms like ThingSpeak and Blynk offered intuitive insights into the data. This project lays the groundwork for scalable and sustainable IoT solutions, addressing both technical challenges and user needs for smart, connected homes