1. Embedded Device

The Raspberry Pi Pico W is configured to collect temperature and humidity data from a DHT11 sensor and send it to the backend server using HTTP POST requests. It ensures a reliable connection by verifying internet availability through Google pings.

2. Resource Allocation

The system optimizes resources by leveraging the Pico W for edge processing and a DigitalOcean VPS for cloud hosting. This ensures cost-effective and scalable infrastructure for continuous data collection, storage, and processing.

3. Cloud Service

The DigitalOcean Ubuntu server hosts the backend using Node.js 23, stores data in InfluxDB v2.7.10, and serves the user interface via Nginx 1.24.0. This combination ensures a robust and secure IoT pipeline.

4. Cloud Service Configuration

The server is configured to operate 24/7, with public ports (80, 443) properly routed and firewall rules in place to allow safe traffic flow. Backend and database services are integrated seamlessly for efficient data handling.

5. Backend Collecting Data

The Node.js backend receives sensor data, sanitizes it, and writes it into InfluxDB. Real-time logging ensures operational transparency and facilitates debugging, ensuring the system remains reliable.

6. User Interface

The dashboard, hosted by Nginx, provides a real-time display of temperature, humidity, and light control. It offers a simple and accessible interface for users to monitor and interact with the system.

Our project cost

total cost of our project is 12€ 1 euro for dht11 1 euro for relay and 10 for pico,but in the mass production this cost can be cut in 2 times due to the difference in the purchase price of a large quantity from the manufacturer.

Extra 4 euro per month for Digital ocean.

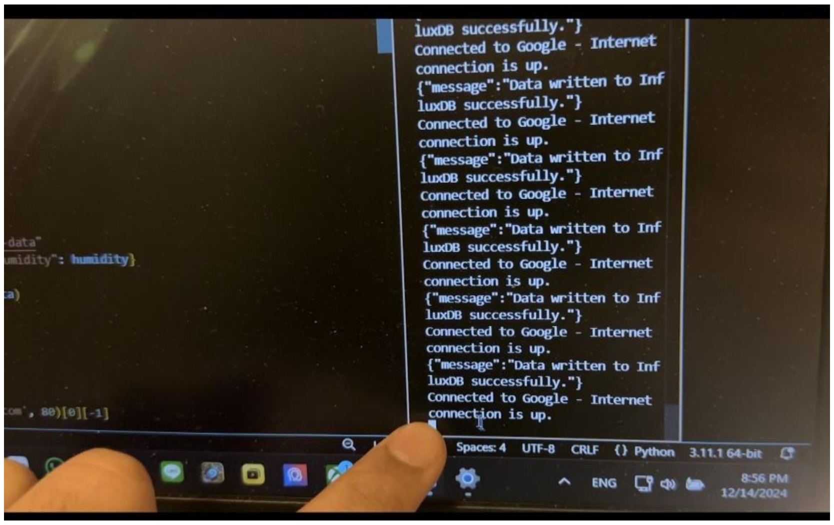
How the System Works?

Wi-Fi Connection: The Pico W connects to Wi-Fi using our network name and password.

Data Collection: Reads sensor data every 5 seconds.

Sending Data: Shares the temperature and humidity with the cloud.

Internet Check: Makes sure the device stays online.



```
luxDB successfully."}
Connected to Google - Internet
connection is up.
{"message":"Data written to Inf
luxDB successfully."}
Connected to Google - Internet
connection is up.
{"message":"Data written to Inf
luxDB successfully."}
Connected to Google - Internet
connection is up.
{"message":"Data written to Inf
luxDB successfully."}
Connected to Google - Internet
connection is up.
{"message":"Data written to Inf
luxDB successfully."}
Connected to Google - Internet
connection is up.
{"message":"Data written to Inf
luxDB successfully."}
Connected to Google - Internet
connection is up.
```

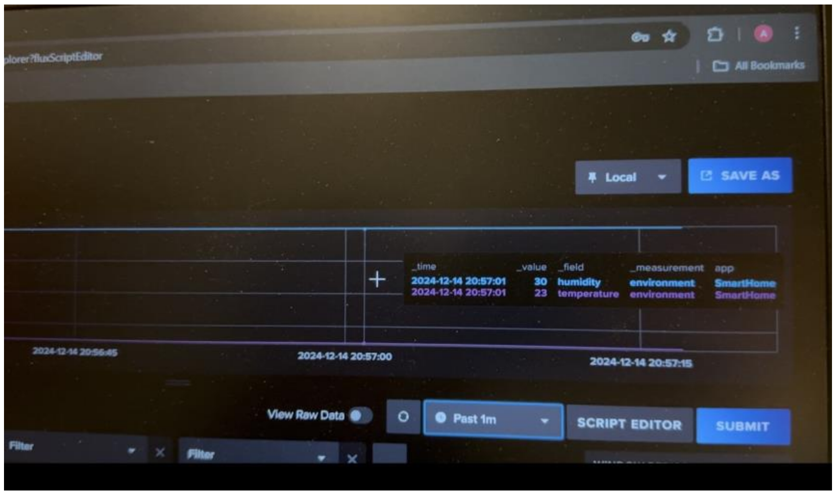
Cloud Server on DigitalOcean

Nginx: Handles internet traffic.

Node.js: Processes the data and saves it to InfluxDB, a special database for time-based data.

Runs all the time to keep the system working smoothly.

User Interface:A simple webpage shows the temperature and humidity in real time.



_time	_value	_field	_measurement	app
2024-12-14 20:57:01	30	humidity	environment	SmartHome
2024-12-14 20:57:01	23	temperature	environment	SmartHome

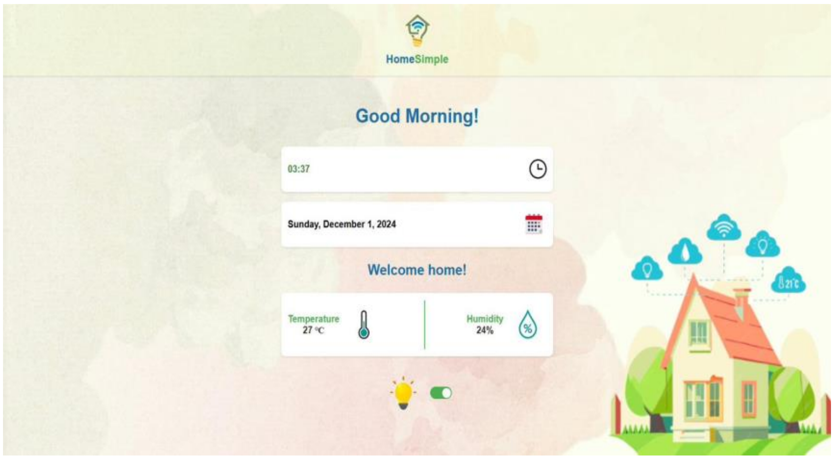
Why It's Useful?

Live Updates: See the latest data instantly.

Expandable: Add more devices or sensors easily.

Affordable: Uses low-cost devices and cloud hosting.

Reliable: Works nonstop with cloud support.



How It All Works Together

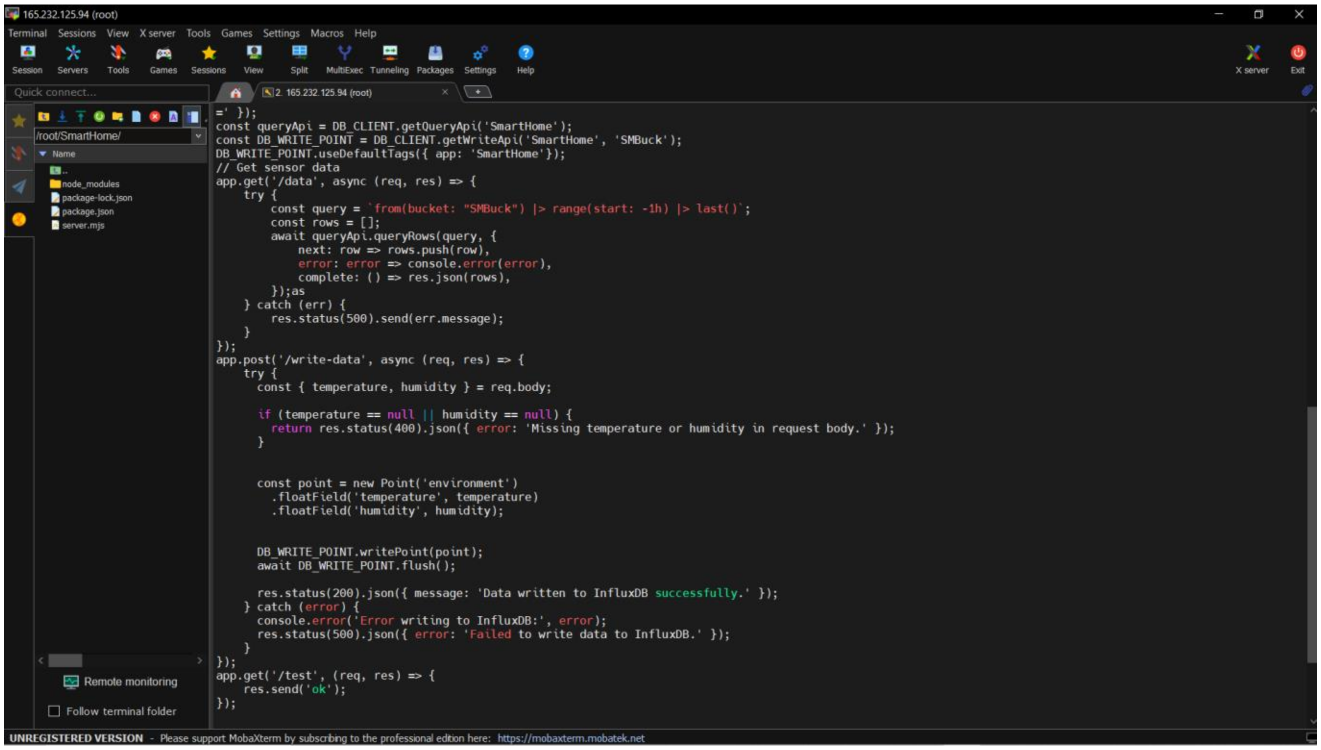
The sensor collects data → Pico W sends it to the cloud → The cloud saves it → The webpage shows it live.

This system is simple, easy to use, and perfect for monitoring your home!

Here is the Link to the youtube video.

Here is the github repository which consist the data flow of embedded system and frontend **SmartHomeM**

Note: We have used the direct approach for Backend coding where we directly code the system by getting inside the system using ssh by **MobaXtrem** which is helpful as server side code editor. So, Unfortunately we are not able to submit the backend code using github.



```
=' });
const queryApi = DB_CLIENT.getQueryApi('SmartHome');
const DB_WRITE_POINT = DB_CLIENT.getWriteApi('SmartHome', 'SMBuck');
DB_WRITE_POINT.useDefaultTags({ app: 'SmartHome'});
// Get sensor data
app.get('/data', async (req, res) => {
    try {
        const query = `from(bucket: "SMBuck") |> range(start: -1h) |> last()`;
        const rows = [];
        await queryApi.queryRows(query, {
            next: row => rows.push(row),
            error: error => console.error(error),
            complete: () => res.json(rows),
        });as
    } catch (err) {
        res.status(500).send(err.message);
    }
});
app.post('/write-data', async (req, res) => {
    try {
      const { temperature, humidity } = req.body;

      if (temperature == null || humidity == null) {
        return res.status(400).json({ error: 'Missing temperature or humidity in request body.' });
      }


      const point = new Point('environment')
        .floatField('temperature', temperature)
        .floatField('humidity', humidity);


      DB_WRITE_POINT.writePoint(point);
      await DB_WRITE_POINT.flush();

      res.status(200).json({ message: 'Data written to InfluxDB successfully.' });
    } catch (error) {
      console.error('Error writing to InfluxDB:', error);
      res.status(500).json({ error: 'Failed to write data to InfluxDB.' });
    }
});
app.get('/test', (req, res) => {
    res.send('ok');
});
```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net