INTERNSHIP PROJECT REPORT

“Smart Humidity and Temperature Monitoring System”

BACHELOR OF TECHNOLOGY IN ELECTRONICS AND COMMUNICATION ENGINEERING

Submitted by

Shaik Abdul Khadar Tawfiq(239X5A04L6)

Internship Organization

LEARNSQUARE

Department of Electronics and Communication Engineering

G. PULLA REDDY ENGINEERING COLLEGE (AUTONOMOUS)

(Affiliated to JNTUA, ANANTAPURAMU)

KURNOOL-518007

2024 – 2025

INTERNSSHIP PROJECT REPORT

Submitted in accordance with the requirement for the degree

Title of the Report : Smart Humidity and Temperature Monitoring System

Name of College : G PULLA REDDY ENGINEERING COLLEGE

Department : Electronics & Communication

Duration : From 05-05-2025 to 31-05-2025

Name of Student :S. Abdul Khadar Tawfiq

Program of study : FOUR YEARS B. Tech program

Year of Study : IV

Register Number : 239X5A04L6

CERTIFICATE OF COMPLETION

This is to certify that S. Abdul Khadar Tawfiq, Roll No. 239X5A04L6, student of Electronics and Communication Engineering, G Pulla Reddy Engineering College, has successfully completed a four-week internship on the project titled **"Develop a Web Dashboard with Python to exchange data between Raspberry pi and the dashboard by using any sensor."** from 05-05-2025 to 31-05-2025 by Learn Square Technology.

During the internship, the student actively engaged in technical activities, demonstrated problem- solving abilities, and showed commitment to project objectives. This certificate acknowledges the successful and satisfactory completion of the assigned work.

Authorized Signatory
[Name, Designation]
[Organization Seal]
Date: _____

ACKNOWLEDGEMENT

I am grateful to our beloved Principal **Dr. B. Sreenivasa Reddy**, for providing requisite facilities and helping us in providing such a good environment.

I am thankful to our Head of the Department **Dr. K. Suresh Reddy**, for his whole hearted support and encouragement during the project session.

We Would like to express our sincere gratitude to **Learn Square Technologies** for their guidance and continued support throughout our project.

I would also like to thank the faculty and staff of the institute for their support.

I wish to convey our acknowledgements to all information the staff members of the ECE department for giving the requirements needed for our project work.

Finally, I wish to thank all our friends and well-wishers who have helped us directly or indirectly during the course of this project work.

CONTENTS

S. No	Chapters	Page. No
1	Executive Summary	1
2	Introduction	2-4
3	Organization Profile	5
4	Internship Activities	6-7
5	Technical Details	8-10
6	Learning Outcomes	11
7	Conclusion	12
8	Reference	13
9	Appendices	14-20

1. EXECUTIVE SUMMARY

This project focuses on developing a lightweight and efficient web-based dashboard using Python to facilitate real-time monitoring and data exchange between a Raspberry Pi and a DHT11 sensor. The DHT11 is a low-cost digital temperature and humidity sensor, which is ideal for small-scale IoT applications.

The core of this project involves reading environmental data from the DHT11 sensor connected to the Raspberry Pi and transmitting it to a responsive web dashboard for live display and logging. The dashboard is built using Python's Flask web framework, providing a RESTful interface that allows bi-directional communication. The collected data can be visualized through dynamic charts and logs, enhancing usability for users interested in environmental monitoring.

The solution is scalable, cost-effective, and suitable for educational, agricultural, or home automation projects. Security, usability, and real-time performance have been considered to ensure practical deployment.

2. INTRODUCTION

Introduction:

With the advancement of the Internet of Things (IoT), embedded systems like the Raspberry Pi have become increasingly popular for environmental sensing and automation. One of the most basic yet useful sensors is the DHT11, which measures temperature and humidity. These metrics are essential in a wide variety of domains including weather stations, greenhouses, HVAC systems, and smart homes.

However, the challenge lies not in collecting the data, but in effectively communicating and visualizing it in real-time. This project addresses this need by developing a Python-based web dashboard that acts as a bridge between the sensor data and the user interface.

Objective

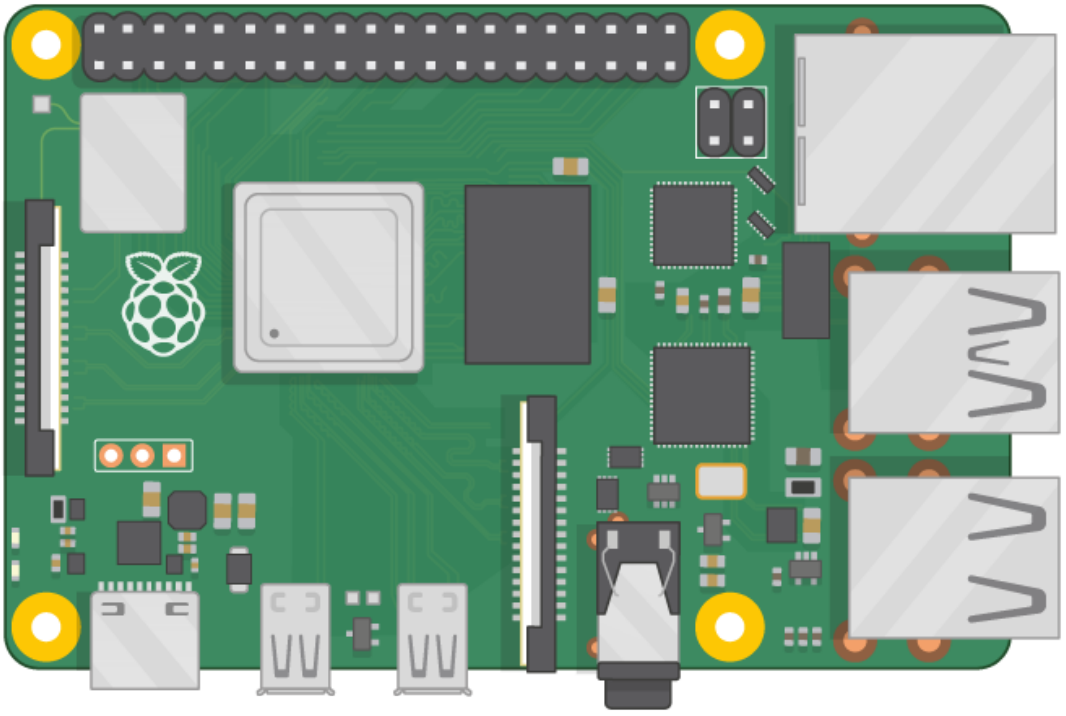
The primary objective of this project is to design and implement a Python-powered web dashboard that:

- Connects to a DHT11 sensor via a Raspberry Pi.
- Collects temperature and humidity data at regular intervals.
- Transmits the data to a web interface using Flask.
- Displays real-time data using graphs and indicators.
- Provides a scalable and user-friendly platform for environmental monitoring.

Scope

This project is limited to:

- Using the DHT11 sensor for data acquisition.
- Running the Python Flask server on a Raspberry Pi.
- Basic visualization and logging on the web dashboard.
- Local network access to the dashboard (with optional extension to remote access).

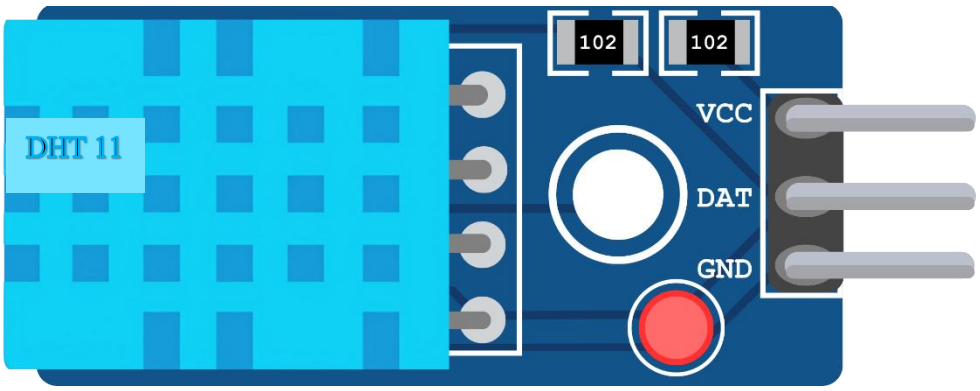


RaspberryPi 4 Module

RASBERRY PI MODULE

1. **Powerful Performance:**
   - Features a Quad-core ARM Cortex-A72 CPU @ 1.5GHz with options for 2GB, 4GB, or 8GB RAM.
   - Significantly faster than previous models, making it suitable for tasks like programming, media streaming, and even light desktop use.
2. **Rich Connectivity:**
   - Includes 2× USB 3.0, 2× USB 2.0, Gigabit Ethernet, Dual-band Wi-Fi, and Bluetooth 5.0.
   - Supports dual 4K micro-HDMI outputs, allowing
   - for dual-monitor setups.
3. **Versatile Applications:**
   - Used in IoT projects, robotics, media centers (like Kodi), retro gaming, and even small web servers.

- Runs a variety of operating systems, including Raspberry Pi OS (Linux), Ubuntu, and even Windows IoT Core.



1. **Measures Temperature and Humidity:**
   - The DHT11 is a digital sensor that provides temperature (0– 50°C) and humidity (20–90%) readings.
   - It's commonly used in weather stations, home automation, and environmental monitoring projects.

2. **Digital Output with Easy Interface:**
   - It uses a single-wire digital interface, making it simple to connect with microcontrollers like Arduino or Raspberry Pi.
   - Requires minimal external components, ideal for beginners and quick prototyping.
3. **Affordable and Widely Available**
   - Known for its low cost and reliability in basic applications
   - Perfect for educational and hobbyist projects, though less accurate than advanced sensors like the DHT22.

3. Company/Organization Profile

Learn Square Technology is a technical development and training firm in Hyderabad. It was involved in the skill development of engineering students and young professionals in advanced technologies. It was started in 2017 by a team of engineers and teachers with the aim of providing practical and useful learning to students in India.

Learn Square started as a small training facility that dealt with Embedded Systems and Arduino training. It grew in phases to offer a vast array of subjects such as Internet of Things (IoT), Artificial Intelligence, Machine Learning, Python Programming, and Full Stack Web Development. The growth of the center has been driven by the growing need for experiential learning and employability skills among engineering graduates.

Learn Square Technology has previously collaborated with numerous schools to conduct campus training programs, industrial internships, and project assistance. The organization is well known for providing structured, experiential learning through workshops, internships, and certification programs.

Mission:

At Learn Square Technology, the mission is to empower students, graduates, and professionals by delivering high-quality, hands-on technical training that is aligned with the latest industry trends and technological advancements. Learn Square Technology aims to bridge the gap between academic concepts and real-world industrial applications. The organization is committed to fostering a practical, project-based learning environment where individuals can gain the skills, confidence, and experience necessary to thrive in the real-world tech industry.

Core Activities:

- Conducting internship programs on real-time projects
- Offering certification courses in IoT, Embedded C, Python, AI/ML, and more
- Organizing hands-on workshops and seminars in engineering colleges
- Providing industry-aligned mini and major project guidance
- Developing IoT and embedded system prototypes as live projects

4. INTERNSHIP ACTIVITIES

During my internship at Learn Square Technology, I worked on a real-time project titled "**Develop a Web Dashboard with Python to exchange data between Raspberry pi and the dashboard by using any sensor**". The project aimed to monitor room temperature and humidity using a Raspberry pi and a DHT11 sensor. The following activities were undertaken:

- **Development Environment Setup:** Raspberry Pi OS Lite was installed for optimal performance. Python 3, Flask, and Adafruit-DHT libraries were set up for development. Flask was configured as an HTTP server with GPIO access enabled.

- **Hardware Integration:** The DHT11 sensor was connected to the Raspberry Pi GPIO pins. Wiring was done carefully for power, ground, and data communication. Additional modules like DS3231 RTC and TFT LCD were explored but not used.

- **Firmware Development:** Python scripts were written to read data from the DHT11 sensor. A RESTful API was created using Flask to serve sensor data. Data was updated periodically using Python scheduling methods.

- **Dashboard Integration:** A dashboard was designed using HTML, CSS, and JavaScript. Was used to display of temperature and humidity. AJAX calls fetched data from the Flask API for real-time updates.

- **Testing and Debugging:** Ensured reliable Wi-Fi connection, validated sensor accuracy, and optimized dashboard refresh behavior.

Tools and Technologies Used

- **Microcontroller**: Raspberry Pi (Model 3/4 or Zero W with Wi-Fi support)
- **Sensor**: DHT11 (Temperature and Humidity sensor)
- **Software**: Python 3, Flask framework, Thony (Compiler/ IDE) and Raspbian OS (Lite version)
- **Dashboard Platforms**: Custom-built Web Dashboard using Flask, HTML, JavaScript, and Chart.js
- **Programming Language**: Python
- **Communication**: Wi-Fi (802.11) via Raspberry Pi's built-in wireless module

Challenges Faced and Solutions Implemented:

- **Wi-Fi Connection Problems:** The Raspberry Pi occasionally experienced unstable Wi-Fi connections. This was resolved by configuring network settings properly and ensuring the device was within range of a reliable Wi-Fi signal.

- **Sensor Inaccuracy:** The DHT11 sometimes gave incorrect readings due to fast data requests. We solved this by adding a small delay in the code and checking the data before displaying it.

- **Slow Dashboard Updates:** The web dashboard was not updating smoothly. We improved this by optimizing the refresh rate and avoiding unnecessary page reloads.

- **Performance Issues:** Running the web server and sensor reading code at the same time made the system slow sometimes. We improved this by writing efficient code and keeping the server simple.

- **Sensor Warm-Up Time:** The DHT11 sensor requires a few seconds to stabilize after powering on, which caused initial incorrect readings. Adding a short delay before the first data read ensured reliable measurements.

5. Technical Details

The system reads real-time temperature and humidity data from a DHT11 sensor connected to a Raspberry Pi. A custom-built local web dashboard fetches this data from a Python Flask server running on the Raspberry Pi and displays it using modern web technologies. The entire setup runs over the local Wi-Fi network without relying on external cloud platforms.

Raspberry pi Configuration:

- Reads temperature and humidity data from the DHT11 sensor using GPIO pins.
- Runs a Python script to collect and process the sensor data.
- Hosts a Flask-based web server that sends sensor data through HTTP in JSON format.

Dashboard Details:

- Built using HTML, CSS, and JavaScript for a responsive interface.
- Uses AJAX (JavaScript) to fetch live data from the Flask server via HTTP GET requests.
- Refreshes data every few seconds to show the latest temperature and humidity values in real-time.

Methodologies Applied:

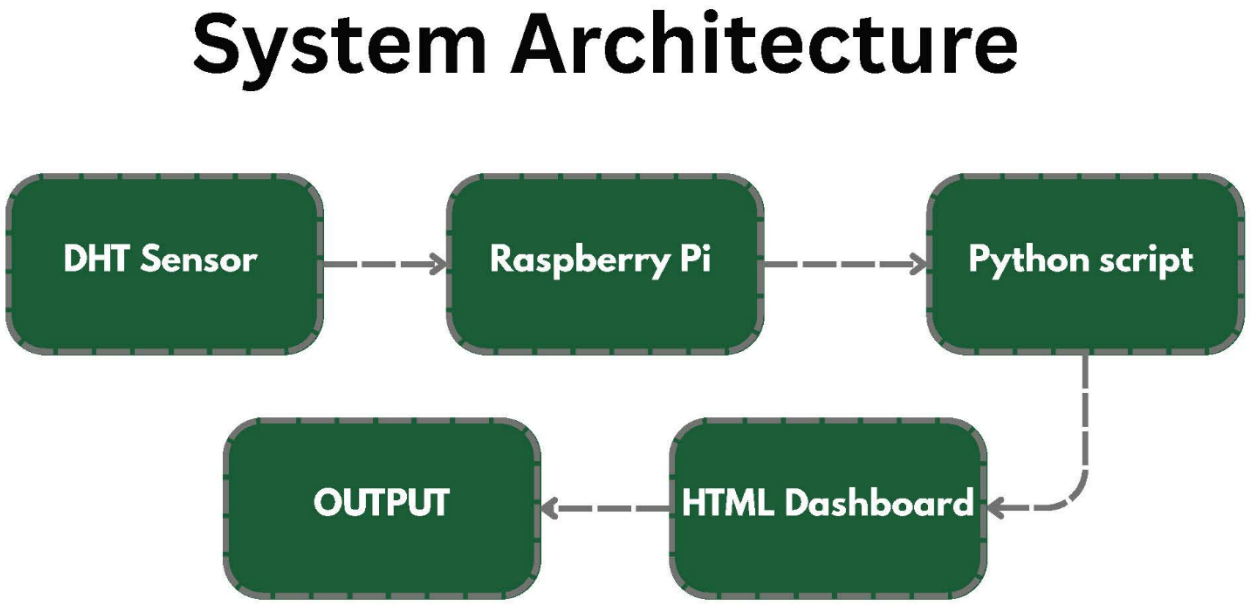
- Applied appropriate timing intervals between DHT11 reads to avoid inaccurate values.
- Used RESTful API design principles to serve sensor data from the backend.
- Implemented real-time graph updates on the frontend using Chart.js for smooth data visualization.

Outcomes Achieved:

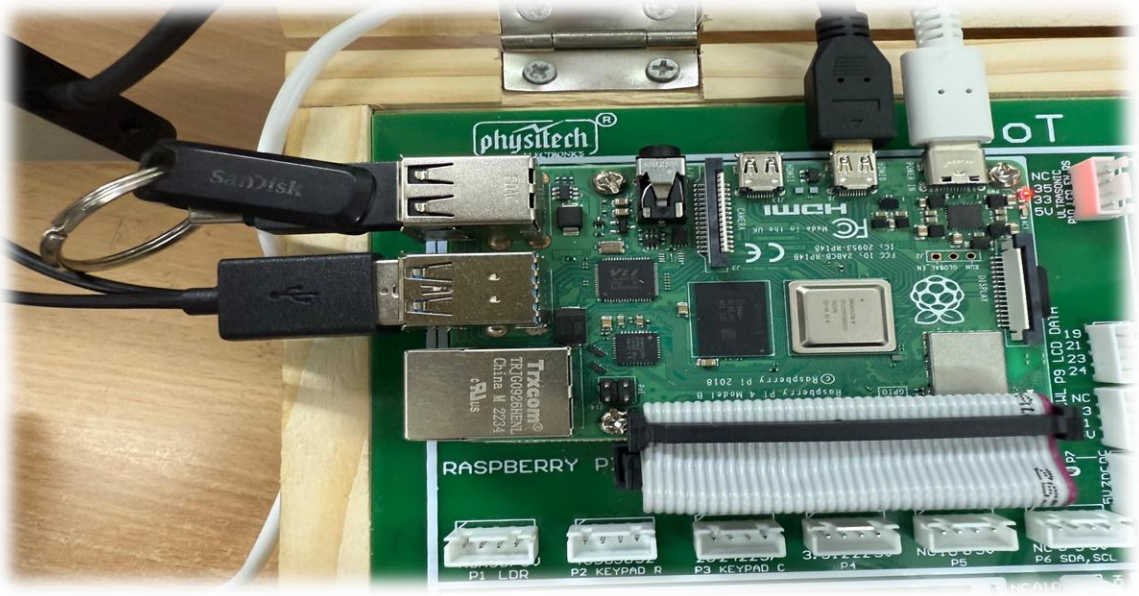
- Accurate environmental data collection.
- Achieved consistent and accurate room temperature and humidity monitoring.
- Enabled real-time data access from any device connected to the same Wi-Fi network.
- Ensured reliable system performance through efficient coding and local data handling.

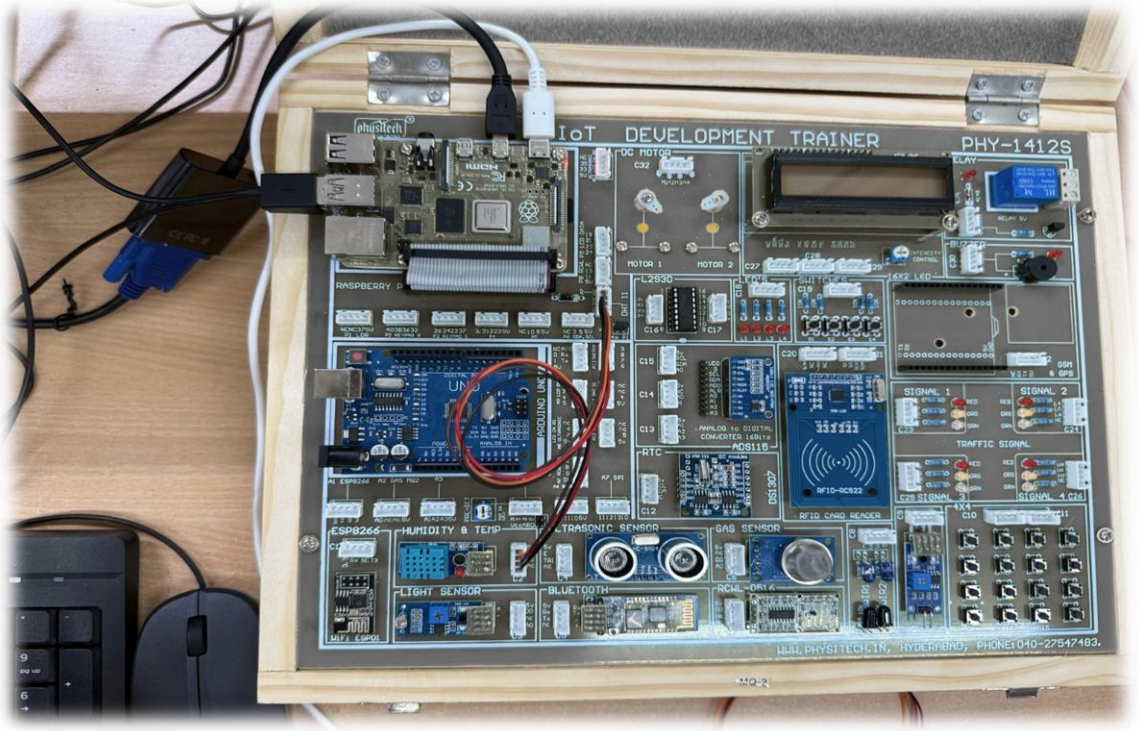
BLOCK DIAGRAM

This shows the basic working flow of the system — the **DHT11 sensor** collects temperature and humidity data, Raspberry Pi processes it, sends it over **Wi-Fi**, and displays it on a **mobile or web dashboard**.



CIRCUIT DIAGRAM:





DHT11 Sensor to Raspberry Pi:

- **VCC** (DHT11) → 5V (Pin 2 or 4 on Raspberry Pi)
- **GND** (DHT11) → GND (Pin 6 on Raspberry Pi)
- **DATA** (DHT11) → GPIO 17 (Pin 11 on Raspberry Pi**)** (or any available GPIO pin
- **POWER SUPPLY** →The Raspberry Pi is powered via USB power adapter (typically 5V, 2.5A or higher). The DHT11 sensor receives power directly from the Raspberry Pi's 5V and GND pins

6. LEARNING OUTCOMES

- **Embedded IoT Programming:**
  Gained practical experience in using Python on Raspberry Pi to read sensor data and host a local web server for real-time monitoring.

- **Sensor Interfacing:**
  Successfully connected and retrieved accurate temperature and humidity data from the DHT11 sensor using GPIO pins and Python libraries.

- **Local Network Integration:**
  Implemented a Flask server to distribute sensor data over a local Wi-Fi network, allowing real-time access through a browser.
- **Web Dashboard Creation:**
  Developed a functional and responsive web dashboard using HTML, JavaScript to visualize live sensor data

- **Debugging & Optimization:**
  Resolved issues related to data inconsistency, refresh lag, and Wi-Fi stability, improving the overall performance and reliability of the system.

7. CONCLUSION

The project successfully demonstrated how to create a web dashboard using Python to exchange real-time data between a Raspberry Pi and any connected sensor. By integrating Python, Flask, and web technologies like HTML, CSS, and JavaScript, a complete system was developed that collects sensor data and displays it in a user-friendly dashboard over a local Wi-Fi network.

This project improved my understanding of embedded system communication, web development, and IoT data handling. It also helped me gain practical experience in setting up real-time data exchange between hardware and software components. Overall, the project provided a strong foundation in building efficient and scalable IoT solutions using open-source tools and platforms.

8. REFERENCES

- "Internet of Things: A Hands-On Approach" – Arshdeep Bahga, Vijay Madisetti
- "Raspberry Pi Cookbook" – Simon Monk"
- "Python Programming for Raspberry Pi" – Tim Cox"
- "Sensor Data Acquisition and Monitoring Using Raspberry Pi and IoT" – ResearchGate
- – Adafruit DHT11 Python library guide
- https://randomnerdtutorials.com/esp32-dht11-dht22-temperature-humidity-web-server/

9. APPENDICES

Python Code:

```
from flask import Flask, request, render_template, redirect
import json
import os

app = Flask(__name__)

command_file = "command.txt"
data_file = "data.json"

@app.route("/")
def dashboard():
    # Read command
    command = "STOP"
    if os.path.exists(command_file):
        with open(command_file, "r") as f:
            command = f.read().strip()

    # Read sensor data
    temperature, humidity = "--", "--"
    if os.path.exists(data_file):
        with open(data_file, "r") as f:
            try:
                data = json.load(f)
                if command == "OK":
                    temperature = data.get("temperature", "--")
                    humidity = data.get("humidity", "--")
            except json.JSONDecodeError:
                pass

    return render_template("dashboard.html", temperature=temperature, humidity=humidity,
command=command)

@app.route("/command", methods=["POST"])
def update_command():
    cmd = request.form.get("cmd", "STOP")
    with open(command_file, "w") as f:
        f.write(cmd)
    return redirect("/")

if __name__ == "__main__":
    app.run(host="0.0.0.0", port=5000)
```

HTML, CSS, JavaScript Code:

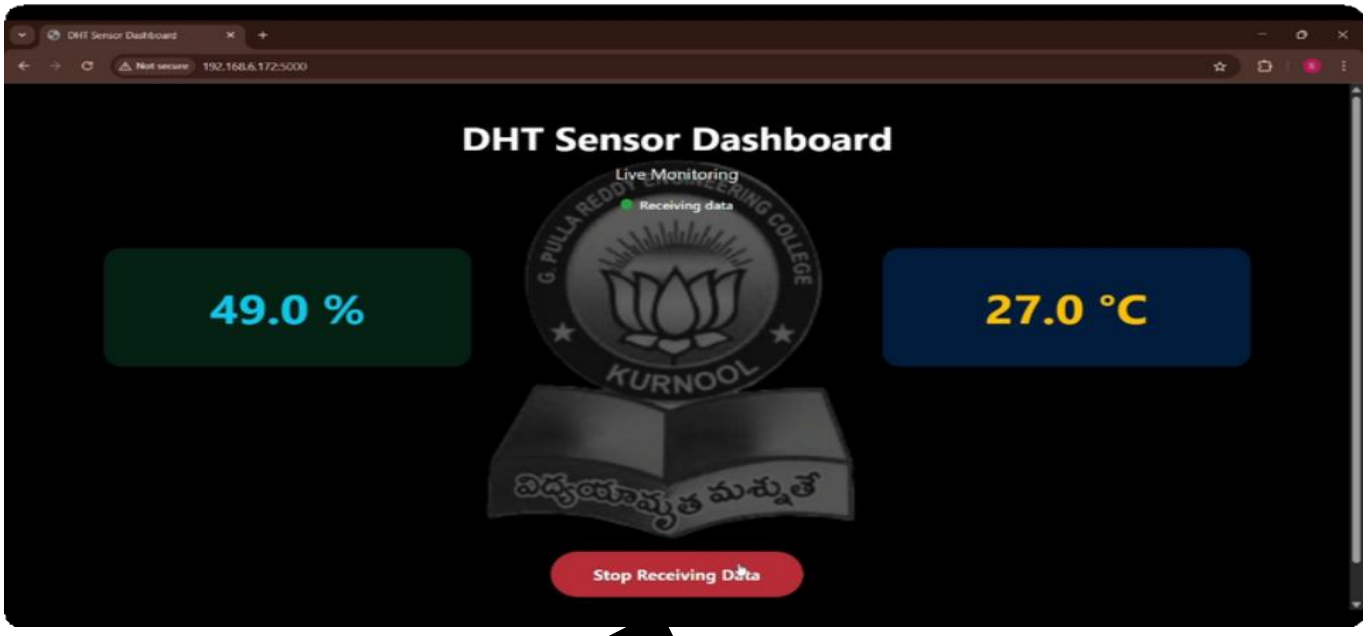
```
<meta charset="UTF-8">
<title>DHT Sensor Dashboard</title>
<meta name="viewport" content="width=device-width, initial-scale=1">
<!-- Bootstrap 5 -->
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css"
rel="stylesheet">
<style>
    body {
        background: url('{{ url_for('static', filename='logo_v2.png') }}') no-repeat center
center fixed;
        background-size: cover;
        font-family: 'Segoe UI', sans-serif;
        min-height: 100vh;
    }
    .dashboard-title {
        font-size: 2.8rem;
        font-weight: 700;
        color: #ffffff;
    }
    .glass-card {http
        background: rgba(255, 255, 255, 0.2);
        border-radius: 20px;
        backdrop-filter: blur(10px);
        box-shadow: 0 8px 30px rgba(0, 0, 0, 0.15);
        padding: 2.5rem;
        color: #fff;
        text-align: center;
        transition: transform 0.3s ease;
    }
    .glass-card:hover {
        transform: scale(1.03);
    }
    .value {
        font-size: 3.5rem;
        font-weight: bold;
    }
    .status-light {
        width: 14px;
        height: 14px;
        border-radius: 50%;
```

```
            display: inline-block;
            margin-right: 0.2rem;
        }
        .status-ok {
            background-color: #28a745;
        }
        .status-stop {
            background-color: #dc3545;
        }
        .control-btn {
            font-size: 1.25rem;
            padding: 0.8rem 2.5rem;
        }
    </style>
</head>
<body>
    <div class="container py-5">
        <div class="text-center mb-5">
            <h1 class="dashboard-title mb-2"> DHT Sensor Dashboard</h1>
            <p class="fs-5 text-light">Live Monitoring</p>
            <p>
                <span class="status-light {{ 'status-ok' if command == 'OK' else 'status-stop'
}}"></span>
                <span class="text-light fw-semibold">{{ 'Receiving data' if command == 'OK'
else 'Paused' }}</span>
            </p>
        </div>

        <div class="row justify-content-between mb-4">
    <!-- Humidity card - pushed to the left -->
    <div class="col-md-4 mb-3">
        <div class="glass-card bg-success bg-opacity-25">
            <div class="value text-info">{{ humidity }} %</div>
        </div>
    </div>

    <!-- Temperature card - pushed to the right -->
    <div class="col-md-4 mb-3">
        <div class="glass-card bg-primary bg-opacity-25">
            <div class="value text-warning">{{ temperature }} °C</div>
        </div>
    </div>
</div>

        <!-- Start/Stop Buttons -->
       <div class="text-center" style="margin-top: 240px;">
```

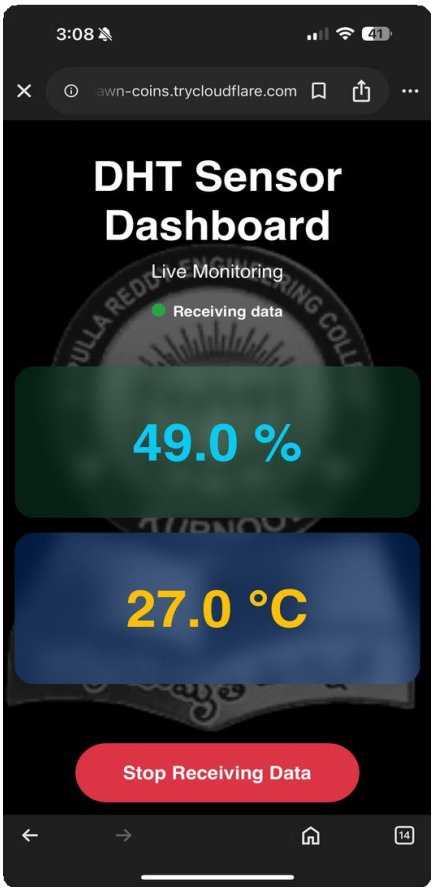
```
        <form action="/command" method="post">
            {% if command == 'OK' %}
                <button class="btn btn-danger btn-lg px-5 py-3 fw-bold shadow rounded-pill
control-btn"
                        name="cmd" value="STOP">
                    Stop Receiving Data
                </button>
            {% else %}
                <button class="btn btn-success btn-lg px-5 py-3 fw-bold shadow rounded-pill
control-btn"
                        name="cmd" value="OK">
                    ▶️ Start Receiving Data
                </button>
            {% endif %}
        </form>
    </div>
  </div>

  <!-- Bootstrap JS -->
  <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"></script>
</body>
</html>
```

Web Dashboard Screenshot:

This is an image of the real-time **web interface dashboard**, showing live readings of temperature and humidity as displayed to the user on a phone or PC.



DESKTOP VIEW



MOBILE VIEW

Android App Screenshot: