Proyecto tecnológico 4

Estación Meteorológica con IoT

Ing. Lukaszewicz, Cristian

Documento Versión 1.0

Índice

1. **Observación del Entorno**
    - 1.1. Disparador
    - 1.2. Recorte del Tema
    - 1.3. Objetivos
    - 1.3.1. Objetivo General
    - 1.3.2. Objetivos Específicos
    - 1.4. Materiales para Ejecutar el Proyecto
    - 1.5. Otros Actores Involucrados
    - 1.6. Roles y Funciones a Designar en Cada Grupo de Trabajo
    - 1.6.1. Equipos
    - 1.7. Tiempo Total del Proyecto
    - 1.8. Plan de Trabajo
    - 1.8.1. Planteamiento Guía
    - 1.8.2. Organización y Planificación
    - 1.8.3. Descripción Detallada de Actividades
2. **Diagnóstico**
    - 2.1. Conocimiento Técnico
    - 2.2. Aporte Curricular
    - 2.3. Aporte al Desarrollo de Competencias Genéricas
3. **Planteo de las Metas**
    - 3.1. Selección de Contenidos
    - 3.2. Producto Final
    - 3.3. Objetivos de Aprendizaje
    - 3.4. Resultados de Aprendizaje
4. **Acciones a Desarrollar**
    - 4.1. Semana 1: Introducción a los Sensores y Plataformas de IoT
    - 4.2. Semana 2: Montaje de la Estación Meteorológica con Sensores
    - 4.3. Semana 3: Conexión con la Plataforma en Línea para Monitoreo
    - 4.4. Semana 4: Pruebas y Ajuste del Sistema
5. **Evaluación**

- 5.1. Instrumento de Evaluación
- 5.2. Evaluación Formativa
- 5.3. Evaluación Sumativa
- 5.4. Entregables Esperables
- 5.5. Rúbricas

6. **Expansiones Opcionales**
   - 6.1. Agregar Más Sensores y Variables
   - 6.2. Implementar Alertas y Notificaciones
   - 6.3. Análisis de Datos y Visualización Avanzada
7. **Referencias**
8. **Anexos**
   - Anexo A: Diagramas de Conexión de Hardware
   - Anexo B: Scripts de Python Utilizados
   - Anexo C: Capturas de Pantalla de la Plataforma IoT
   - Anexo D: Manual de Usuario de la Estación Meteorológica

1. Observación del Entorno

1.1. Disparador

En un mundo donde el cambio climático y las condiciones meteorológicas impactan directamente en nuestras vidas, la capacidad de medir y monitorear el entorno se vuelve esencial. Nos preguntamos: ¿Cómo podemos medir las condiciones meteorológicas de nuestro entorno usando tecnología IoT?

1.2. Recorte del Tema

Crear una estación meteorológica que mida temperatura y humedad y envíe los datos a una plataforma en línea, permitiendo la visualización en tiempo real y el análisis de las condiciones ambientales locales.

1.3. Objetivos

1.3.1. Objetivo General

Desarrollar una estación meteorológica funcional que integre sensores de temperatura y humedad con una plataforma IoT, permitiendo la recolección y visualización de datos ambientales en tiempo real.

1.3.2. Objetivos Específicos

- Implementar sensores de temperatura y humedad para la recolección de datos ambientales.
- Configurar una plataforma IoT (ThingSpeak o Blynk) para el envío y visualización de datos.
- Desarrollar scripts en Python para la lectura de sensores y la transmisión de datos.
- Fomentar el trabajo en equipo y el desarrollo de competencias técnicas y blandas en los estudiantes.

1.4. Materiales para Ejecutar el Proyecto

- **Hardware:**
- Raspberry Pi o Arduino con módulos Wi-Fi
- Netbook con Windows (si no se dispone de Raspberry Pi)
- Sensores de temperatura y humedad (DHT11 o DHT22)
- Cables y componentes electrónicos adicionales (protoboard, resistencias, etc.)
- Fuentes de alimentación

- Caja de distribución para el montaje de la estación meteorológica
- **Software:**
- Plataforma IoT (ThingSpeak, Blynk)
- Python 3.x
- IDE de Arduino (si se usa Arduino)
- Bibliotecas de Python: `Adafruit_DHT`, `requests`, `paho-mqtt` (si se usa MQTT)
- **Otros:**
- Acceso a internet
- Herramientas básicas de electrónica (soldador, multímetro, etc.)

1.5. Otros Actores Involucrados

- Docentes de electrónica, IoT y ciencia de datos
- Asistentes técnicos para la configuración de plataformas IoT
- Coordinadores de proyectos tecnológicos
- Proveedores de hardware y software
- Tutores externos (opcional) para soporte adicional

1.6. Roles y Funciones a Designar en Cada Grupo de Trabajo

1.6.1. Equipos

Cada grupo se dividirá en:

- **Líder de Equipo:** Coordina actividades y mantiene el progreso.
- **Programadores:** Configuran los scripts en Python y programan el Arduino (si aplica).
- **Encargados del Hardware:** Montan sensores y dispositivos de control.
- **Testers:** Realizan pruebas del sistema.
- **Documentalistas:** Se encargan de la documentación y reportes del proyecto.

1.7. Tiempo Total del Proyecto

El proyecto se desarrollará en cuatro semanas, cada una enfocada en diferentes aspectos del desarrollo e integración del sistema de monitoreo meteorológico.

1.8. Plan de Trabajo

1.8.1. Planteamiento Guía

"¿Cómo podemos medir las condiciones meteorológicas de nuestro entorno usando tecnología IoT para recolectar, transmitir y visualizar datos en tiempo real?"

1.8.2. Organización y Planificación

El proyecto se divide en cuatro etapas semanales, cada una con tareas específicas de investigación, desarrollo e integración.

1.8.3. Descripción Detallada de Actividades

2. Diagnóstico

2.1. Conocimiento Técnico

Para llevar a cabo este proyecto, es esencial que los estudiantes tengan conocimientos básicos en:

- **Conceptos Básicos de Electrónica:** Entender cómo funcionan los sensores y cómo conectarlos a una placa de desarrollo.
- **Programación en Python:** Para el desarrollo de scripts que leen datos de los sensores y los envían a la plataforma IoT.
- **Uso de Plataformas IoT:** Familiarizarse con plataformas como ThingSpeak o Blynk para la visualización de datos en tiempo real.
- **Ciencia de Datos Básicos:** Entender cómo interpretar y analizar los datos recolectados.

2.2. Aporte Curricular

Este proyecto contribuye al currículo en las siguientes áreas:

- **Electrónica:** Integración y control de sensores electrónicos.
- **IoT (Internet de las Cosas):** Configuración y uso de plataformas IoT para el monitoreo y control de dispositivos.
- **Ciencia de Datos:** Recolección y análisis de datos ambientales.
- **Programación:** Aplicación práctica de conceptos de programación en Python.

2.3. Aporte al Desarrollo de Competencias Genéricas

- **Resolución de Problemas:** Identificación y solución de desafíos técnicos durante el desarrollo del proyecto.
- **Trabajo en Equipo:** Colaboración efectiva entre los miembros del grupo.
- **Organización y Planificación:** Gestión del tiempo y recursos para cumplir con los objetivos del proyecto.
- **Motivación y Creatividad:** Fomentar el interés por la tecnología y su aplicación en problemas reales.

3. Planteo de las Metas

3.1. Selección de Contenidos

- **Sensores de Temperatura y Humedad:** Funcionamiento y aplicaciones de los sensores DHT11 y DHT22.
- **Introducción a IoT:** Conceptos básicos y uso de plataformas de monitoreo en línea.
- **Programación en Python:** Desarrollo de scripts para la lectura de sensores y envío de datos.
- **Visualización de Datos:** Configuración de dashboards en plataformas IoT para la visualización en tiempo real.

3.2. Producto Final

Una estación meteorológica funcional que mide temperatura y humedad, enviando los datos a una plataforma en línea para su visualización en tiempo real, permitiendo el monitoreo continuo de las condiciones ambientales.

3.3. Objetivos de Aprendizaje

- Comprender el funcionamiento de los sensores de temperatura y humedad.
- Aplicar conceptos de electrónica para la conexión y montaje de sensores.
- Desarrollar scripts en Python para la recolección y transmisión de datos.
- Configurar y utilizar una plataforma IoT para la visualización de datos en tiempo real.
- Analizar y presentar datos ambientales de manera efectiva.

3.4. Resultados de Aprendizaje

- **Implementación de Sensores:** Montaje correcto de sensores de temperatura y humedad en la placa de desarrollo.
- **Recolección de Datos:** Desarrollo de scripts en Python que leen y almacenan datos de los sensores.
- **Transmisión de Datos:** Configuración exitosa de la transmisión de datos a una plataforma IoT.

- **Visualización de Datos:** Creación de dashboards en la plataforma IoT que muestran los datos en tiempo real.
- **Análisis de Datos:** Interpretación y presentación de los datos recolectados para identificar patrones o anomalías.

4. Acciones a Desarrollar

4.1. Semana 1: Introducción a los Sensores y Plataformas de IoT

Objetivo de la Etapa:
Introducir a los estudiantes en los conceptos básicos de sensores de temperatura y humedad, así como en las plataformas IoT que se utilizarán para el monitoreo y visualización de datos.

Actividades:

- **Conceptos Básicos de Sensores:**
- **Definición y Funcionamiento:** Explicación de cómo funcionan los sensores DHT11 y DHT22.
- **Aplicaciones:** Uso de sensores en proyectos de monitoreo ambiental.
- **Introducción a IoT:**
- **¿Qué es IoT?** Definición y ejemplos de aplicaciones en la vida real.
- **Plataformas IoT:** Presentación de plataformas como ThingSpeak y Blynk, sus características y usos.
- **Instalación de Software y Bibliotecas:**
- **Python y Bibliotecas Necesarias:**

```
sudo apt update
sudo apt install python3-pip -y
pip3 install Adafruit_DHT requests paho-mqtt
```

Explicación: Este comando actualiza el sistema e instala Python3 y las bibliotecas necesarias para leer sensores y enviar datos a la plataforma IoT.

- **Configuración de la Plataforma IoT:**
- **ThingSpeak:**
- Crear una cuenta en ThingSpeak.
- Crear un nuevo canal y configurar campos para temperatura y humedad.
- Obtener la API Key del canal.

- **Blynk:**
- Descargar la aplicación Blynk en un dispositivo móvil.
- Crear un nuevo proyecto y obtener el Auth Token.
- Configurar widgets para recibir datos de temperatura y humedad.
- **Dinámica de Introducción:**
- **Preguntas de Guía:**
- ¿Qué tipos de datos podemos recolectar del entorno?
- ¿Cómo puede la tecnología IoT ayudarnos a monitorear el clima local?
- **Discusión en Grupo:**
- Intercambio de ideas sobre la importancia del monitoreo ambiental.
- Identificación de posibles ubicaciones para instalar la estación meteorológica.

4.2. Semana 2: Montaje de la Estación Meteorológica con Sensores

Objetivo de la Etapa:
Montar físicamente la estación meteorológica integrando los sensores de temperatura y humedad con la placa de desarrollo (Raspberry Pi o Arduino).

Actividades:

- **Conexión de Sensores:**
- **Raspberry Pi:**
- **Esquema de Conexión del DHT11/DHT22:**
- VCC del sensor a 5V de Raspberry Pi.
- GND del sensor a GND de Raspberry Pi.
- DATA del sensor a GPIO 4.
- Resistencia pull-up de 10kΩ entre VCC y DATA.
- **Código de Prueba para Lectura de Sensores:**

```
import Adafruit_DHT


SENSOR = Adafruit_DHT.DHT22

PIN = 4
```

```
humedad, temperatura = Adafruit_DHT.read_retry(SENSOR, PIN)
if humedad is not None and temperatura is not None:
    print(f'Temperatura: {temperatura:.1f}°C')
    print(f'Humedad: {humedad:.1f}%')
else:
    print('Fallo al leer el sensor')
```

Explicación: Este script lee los valores de temperatura y humedad del sensor y los imprime en la consola.

- **Arduino:**
- **Esquema de Conexión del DHT11/DHT22:**
- VCC del sensor a 5V de Arduino.
- GND del sensor a GND de Arduino.
- DATA del sensor a pin digital 2.
- Resistencia pull-up de 10kΩ entre VCC y DATA.
- **Código de Prueba para Lectura de Sensores:**

```
#include "DHT.h"

#define DHTPIN 2
#define DHTTYPE DHT22

DHT dht(DHTPIN, DHTTYPE);

void setup() {
  Serial.begin(9600);
```

```
  dht.begin();
}

void loop() {
  float humedad = dht.readHumidity();
  float temperatura = dht.readTemperature();

  if (isnan(humedad) || isnan(temperatura)) {
    Serial.println("Fallo al leer el sensor DHT!");
    return;
  }

  Serial.print("Temperatura: ");
  Serial.print(temperatura);
  Serial.print("°C ");
  Serial.print("Humedad: ");
  Serial.print(humedad);
  Serial.println("%");
  delay(2000);
}
```

Explicación: Este código lee los valores de temperatura y humedad del sensor DHT22 y los envía al monitor serial.

- **Montaje Físico:**
- **Caja de Distribución:** Ensamblar los componentes en una caja para protegerlos y organizar el cableado.

- **Prototoboard:** Utilizar una protoboard para facilitar las conexiones y evitar soldaduras.
- **Pruebas Iniciales:**
- Verificar que los sensores están correctamente conectados y que los scripts leen los valores de temperatura y humedad sin errores.
- **Dinámica de Trabajo:**
- **Trabajo en Equipo:** Distribuir tareas entre los miembros del grupo para el montaje físico y la programación.
- **Resolución de Problemas:** Identificar y solucionar problemas de conexión o lectura de sensores.

4.3. Semana 3: Conexión con la Plataforma en Línea para Monitoreo

Objetivo de la Etapa:
Configurar la transmisión de datos desde la estación meteorológica a una plataforma IoT para la visualización y análisis en tiempo real.

Actividades:

- **Configuración de Scripts para Envío de Datos:**
- **Raspberry Pi:**
- **Código para Enviar Datos a ThingSpeak:**

```
import Adafruit_DHT
import requests
import time

SENSOR = Adafruit_DHT.DHT22
PIN = 4
API_KEY = "TU_API_KEY_THINGSPEAK"

def leer_sensor():
    humedad, temperatura = Adafruit_DHT.read_retry(SENSOR, PIN)
```

```
    return humedad, temperatura


def enviar_a_thingspeak(humedad, temperatura):
    url = "https://api.thingspeak.com/update"
    params = {
        "api_key": API_KEY,
        "field1": temperatura,
        "field2": humedad
    }
    response = requests.get(url, params=params)
    if response.status_code == 200:
        print("Datos enviados correctamente")
    else:
        print("Error al enviar datos")


while True:
    humedad, temperatura = leer_sensor()
    if humedad is not None and temperatura is not None:
        enviar_a_thingspeak(humedad, temperatura)
    else:
        print("Fallo al leer el sensor")
    time.sleep(60)  # Enviar datos cada 60 segundos
```

Explicación: Este script lee los datos del sensor y los envía a ThingSpeak cada 60 segundos.

- **Arduino:**
- **Uso de MQTT para Envío de Datos a Blynk:**

- **Configuración de MQTT:**

Explicación: Este código conecta el Arduino a una red Wi-Fi, lee los datos del sensor DHT22 y los publica en un broker MQTT público cada 60 segundos.

- **Configuración de la Plataforma IoT:**
- **ThingSpeak:**
- Verificar que el canal está configurado correctamente y que los campos de temperatura y humedad reciben los datos.
- **Blynk:**
- Configurar widgets para recibir y visualizar los datos publicados en el broker MQTT.
- **Pruebas de Conexión:**
- Verificar que los datos se envían correctamente desde la estación meteorológica a la plataforma IoT.
- Corroborar la actualización en tiempo real de los dashboards.
- **Dinámica de Trabajo:**
- **Programación en Equipo:** Distribuir tareas de desarrollo de scripts y configuración de la plataforma IoT.
- **Resolución de Problemas:** Identificar y solucionar problemas de conexión o transmisión de datos.

4.4. Semana 4: Pruebas y Ajuste del Sistema

Objetivo de la Etapa:
Realizar pruebas exhaustivas del sistema de monitoreo meteorológico, identificar y corregir errores, y optimizar la transmisión y visualización de datos.

Actividades:

- **Pruebas de Funcionamiento:**
- **Lectura de Sensores:** Verificar que los sensores de temperatura y humedad están proporcionando datos precisos.
- **Transmisión de Datos:** Asegurar que los datos se envían correctamente a la plataforma IoT sin interrupciones.

- **Visualización en Tiempo Real:** Confirmar que los dashboards reflejan los datos en tiempo real.
- **Optimización del Sistema:**
- **Mejorar la Frecuencia de Envío:** Ajustar los intervalos de tiempo para enviar datos según las necesidades del monitoreo.
- **Reducir Latencia:** Optimizar el código para minimizar la demora en la transmisión de datos.
- **Implementar Manejo de Errores:** Añadir manejo de excepciones en los scripts para gestionar fallos de conexión o lecturas erróneas de sensores.

```
import Adafruit_DHT
import requests
import time

SENSOR = Adafruit_DHT.DHT22
PIN = 4
API_KEY = "TU_API_KEY_THINGSPEAK"

def leer_sensor():
    try:
        humedad, temperatura = Adafruit_DHT.read_retry(SENSOR,
PIN)
        if humedad is not None and temperatura is not None:
            return humedad, temperatura
        else:
            raise ValueError("Fallo al leer el sensor")
    except Exception as e:
```

```
        print(f"Error al leer el sensor: {e}")
        return None, None

def enviar_a_thingspeak(humedad, temperatura):
    try:
        url = "https://api.thingspeak.com/update"
        params = {
            "api_key": API_KEY,
            "field1": temperatura,
            "field2": humedad
        }
        response = requests.get(url, params=params)
        if response.status_code == 200:
            print("Datos enviados correctamente")
        else:
            print("Error al enviar datos")
    except Exception as e:
        print(f"Error al enviar datos: {e}")

while True:
    humedad, temperatura = leer_sensor()
    if humedad is not None and temperatura is not None:
        enviar_a_thingspeak(humedad, temperatura)
    time.sleep(60)
```

- **Documentación de Resultados:**

- Registrar los resultados de las pruebas, incluyendo cualquier problema encontrado y las soluciones implementadas.
- Crear gráficos que muestren la precisión de los datos y la estabilidad del sistema.
- **Feedback y Mejoras:**
- Recopilar retroalimentación de los estudiantes sobre la facilidad de uso y la precisión del sistema.
- Implementar mejoras basadas en el feedback recibido.
- **Preparación para la Presentación Final:**
- Organizar los datos y resultados obtenidos para su presentación.
- Preparar demostraciones en vivo de la estación meteorológica y su integración con la plataforma IoT.

5. Evaluación

5.1. Instrumento de Evaluación

- **Evaluación Continua:** Revisión semanal del progreso del proyecto y cumplimiento de objetivos.
- **Evaluación Final:** Verificación de la funcionalidad completa de la estación meteorológica, incluyendo la lectura de sensores y la transmisión de datos a la plataforma IoT.
- **Autoevaluación y Coevaluación:** Reflexión individual y evaluación del trabajo en equipo.

5.2. Evaluación Formativa

- **Observación Directa:** Monitoreo de la participación y compromiso de los estudiantes durante las actividades.
- **Bitácoras de Proyecto:** Registro de actividades, problemas y soluciones propuestas por los estudiantes.

5.3. Evaluación Sumativa

- **Funcionamiento del Sistema:** Evaluación de la precisión y eficiencia de la lectura de sensores y la transmisión de datos.
- **Presentación del Proyecto:** Exposición oral y demostración de la estación meteorológica funcionando en tiempo real.
- **Informe Final:** Documento escrito detallando el desarrollo del proyecto, decisiones técnicas y resultados obtenidos.

5.4. Entregables Esperables

- **Bitácoras de Proyecto:** Documentación diaria de actividades y avances.
- **Programas en Python:** Scripts desarrollados para la lectura de sensores y envío de datos.
- **Documentación Técnica:** Especificaciones de hardware y software utilizados.
- **Informe Final:** Compilación de todo el trabajo realizado con análisis y conclusiones.
- **Presentación del Proyecto:** Diapositivas y demostración de la estación meteorológica funcionando.

5.5. Rúbricas

Rúbrica para la Evaluación del Sistema de Monitoreo Meteorológico

Criterio	Nivel 1 (Insuficiente)	Nivel 2 (Aceptable)	Nivel 3 (Bueno)	Nivel 4 (Excelente)
Lectura de Sensores	No funciona o es inconsistente.	Funciona con dificultades.	Funciona correctamente con algunos ajustes.	Funciona de manera fluida y precisa.
Transmisión de Datos	No logra enviar datos a la plataforma IoT.	Envío básico con errores o retrasos.	Envío adecuado con mínima intervención.	Envío completo y eficiente sin errores.
Visualización de Datos	No hay visualización o es difícil de interpretar.	Visualización básica pero funcional.	Visualización clara y con algunos ajustes.	Visualización completa, clara y detallada en tiempo real.
Integración del Sistema	No hay integración entre sensores y plataforma IoT.	Integración parcial con múltiples errores.	Integración adecuada con algunos aspectos por mejorar.	Integración completa y sin errores entre todos los sistemas.
Documentación	Documentación incompleta o ausente.	Documentación básica pero faltan	Documentación clara y detallada con	Documentación completa, clara y detallada.

		detalles importantes.	mínimos ajustes.	
Presentación	Presentación desorganizada y difícil de entender.	Presentación comprensible pero falta de claridad en algunos puntos.	Presentación clara y bien organizada con buena comunicación.	Presentación excepcionalmente clara, organizada y profesional.

6. Expansiones Opcionales

6.1. Agregar Más Sensores y Variables

- **Sensores Adicionales:**
  Implementar sensores para medir otras variables ambientales como presión atmosférica, luz, nivel de CO_2, etc.
- **Hardware:** Añadir sensores como BMP280 (presión), BH1750 (luz), MQ-135 (CO_2).
- **Software:** Modificar los scripts para leer y enviar datos de los nuevos sensores.
- **Ejemplo de Código para BMP280:**

```
import Adafruit_BMP.BMP085 as BMP085


bmp = BMP085.BMP085()


def leer_presion():
    presion = bmp.read_pressure()
    temperatura = bmp.read_temperature()
    return presion, temperatura


presion, temperatura = leer_presion()
print(f'Presión: {presion} Pa, Temperatura: {temperatura:.1f}°C')
```

- **Análisis de Variables Múltiples:**
  Utilizar los datos de múltiples sensores para realizar análisis más completos del entorno.

6.2. Implementar Alertas y Notificaciones

- **Alertas en la Plataforma IoT:**
  Configurar alertas para condiciones meteorológicas específicas (por ejemplo, temperatura muy alta o baja).
- **ThingSpeak:** Utilizar los canales de alerta integrados para enviar notificaciones por correo electrónico o SMS.
- **Blynk:** Configurar widgets de alerta para recibir notificaciones en dispositivos móviles.
- **Notificaciones Push:**
- **Herramientas:** Uso de servicios gratuitos como IFTTT para enviar notificaciones a dispositivos móviles.
- **Implementación en Python:**

```
import requests


IFTTT_WEBHOOK_URL =
"https://maker.ifttt.com/trigger/{event}/with/key/{your_key}"

EVENT_NAME = "alerta_meteorologica"


def enviar_notificacion(mensaje):

    url = IFTTT_WEBHOOK_URL.format(event=EVENT_NAME,
your_key="TU_WEBHOOK_KEY")

    data = {"value1": mensaje}

    response = requests.post(url, json=data)

    if response.status_code == 200:

        print("Notificación enviada correctamente")

    else:

        print("Error al enviar la notificación")


# Ejemplo de uso
```

```
temperatura = 35

humedad = 10

if temperatura > 30:

    enviar_notificacion("Alerta: La temperatura es demasiado
alta.")

if humedad < 20:

    enviar_notificacion("Alerta: La humedad es demasiado baja.")
```

6.3. Análisis de Datos y Visualización Avanzada

- **Análisis de Tendencias:**
  Utilizar herramientas de ciencia de datos para analizar tendencias y patrones en los datos recolectados.
- **Herramientas:** Uso de Python con bibliotecas como `pandas` y `matplotlib` para análisis y visualización.
- **Ejemplo de Código para Análisis:**

```
import pandas as pd

import matplotlib.pyplot as plt


# Descargar datos desde ThingSpeak

url =
"https://api.thingspeak.com/channels/TU_CHANNEL_ID/feeds.csv?api_
key=TU_READ_API_KEY&results=8000"

datos = pd.read_csv(url)


# Convertir timestamp a datetime

datos['created_at'] = pd.to_datetime(datos['created_at'])
```

```
# Plotear temperatura y humedad

plt.figure(figsize=(10,5))

plt.plot(datos['created_at'], datos['field1'], label='Temperatura
(°C)')

plt.plot(datos['created_at'], datos['field2'], label='Humedad
(%)')

plt.xlabel('Fecha y Hora')

plt.ylabel('Valores')

plt.title('Datos Meteorológicos en Tiempo Real')

plt.legend()

plt.show()
```

- **Dashboards Personalizados:**
  Crear dashboards personalizados que muestren gráficos interactivos y análisis detallados.
- **Herramientas:** Uso de plataformas como Grafana para dashboards avanzados.
- **Integración con ThingSpeak:** Configurar ThingSpeak para enviar datos a Grafana a través de APIs o plugins.

7. Referencias

1. **Adafruit.** *DHT Sensor Library*. 2023. https://learn.adafruit.com/dht.
2. **ThingSpeak.** *IoT Analytics Platform*. 2023. https://thingspeak.com/.
3. **Blynk.** *IoT Platform for Mobile Applications*. 2023. https://blynk.io/.
4. **OpenAI.** *Whisper: An Automatic Speech Recognition System*. 2023. https://openai.com/research/whisper.
5. **Python.org.** *Python Documentation*. 2023. https://docs.python.org/.
6. **Paho MQTT.** *MQTT Client Documentation*. 2023. https://www.eclipse.org/paho/index.php?page=clients/python/index.php.
7. **Transformers by Hugging Face.** *Transformers Documentation*. 2023. https://huggingface.co/docs/transformers/index.
8. **IFTTT.** *IFTTT Webhooks Documentation*. 2023. https://ifttt.com/maker_webhooks.
9. **OpenCV.** *Open Source Computer Vision Library*. 2023. https://opencv.org/.

Anexos

Anexo A: Diagramas de Conexión de Hardware

- **Diagrama 1:** Conexión del Sensor DHT22 a Raspberry Pi.
- **Diagrama 2:** Conexión del Sensor DHT22 a Arduino.
- **Diagrama 3:** Configuración de la Estación Meteorológica en una Protoboard.
- **Diagrama 4:** Integración de Múltiples Sensores (Opcional).

Anexo B: Scripts de Python Utilizados

- **Script 1:** Lectura de Sensores y Envío de Datos a ThingSpeak (Unificado para Raspberry Pi y Windows).

```
import Adafruit_DHT
import requests
import time
import platform


if platform.system() == "Linux":
    # Configuración para Raspberry Pi
    SENSOR = Adafruit_DHT.DHT22
    PIN = 4
    API_KEY = "TU_API_KEY_THINGSPEAK"


    def leer_sensor():
        humedad, temperatura = Adafruit_DHT.read_retry(SENSOR,
PIN)
        return humedad, temperatura
```

```
    def enviar_a_thingspeak(humedad, temperatura):
        try:
            url = "https://api.thingspeak.com/update"
            params = {
                "api_key": API_KEY,
                "field1": temperatura,
                "field2": humedad
            }
            response = requests.get(url, params=params)
            if response.status_code == 200:
                print("Datos enviados correctamente")
            else:
                print("Error al enviar datos")
        except Exception as e:
            print(f"Error al enviar datos: {e}")

elif platform.system() == "Windows":
    # Configuración para Arduino con MQTT (Ejemplo)
    import paho.mqtt.client as mqtt

    MQTT_BROKER = "broker.hivemq.com"
    MQTT_PORT = 1883
    MQTT_TOPIC = "home/estacion_meteorologica"

    client = mqtt.Client()
```

```
    client.connect(MQTT_BROKER, MQTT_PORT, 60)


    def leer_sensor():
        # Este ejemplo asume que los datos vienen del Arduino vía
MQTT
        # Aquí se podrían recibir datos desde una cola o similar
        return None, None  # Placeholder


    def enviar_a_thingspeak(humedad, temperatura):
        # Alternativamente, enviar a otra plataforma o procesar
localmente
        pass  # Placeholder


def main():
    while True:
        humedad, temperatura = leer_sensor()
        if humedad is not None and temperatura is not None:
            enviar_a_thingspeak(humedad, temperatura)
        time.sleep(60)


if __name__ == "__main__":
    main()
```

- **Script 2:** Notificaciones Visuales y Sonoras (Opcional).

```
import tkinter as tk
```

```
from playsound import playsound


def notificacion_visual(mensaje):
    root = tk.Tk()
    root.title("Notificación")
    tk.Label(root, text=mensaje, font=("Helvetica",
16)).pack(pady=20)
    root.after(3000, root.destroy)  # Cierra la ventana después
de 3 segundos
    root.mainloop()


def notificacion_sonora(ruta_sonido):
    playsound(ruta_sonido)


# Ejemplo de uso
if __name__ == "__main__":
    notificacion_visual("Datos enviados correctamente")
    notificacion_sonora("alerta.mp3")
```

- **Script 3:** Control de Dispositivos IoT (Opcional).

```
import paho.mqtt.client as mqtt


MQTT_BROKER = "broker.hivemq.com"
MQTT_PORT = 1883
MQTT_TOPIC = "home/estacion_meteorologica/control"
```

```
client = mqtt.Client()
client.connect(MQTT_BROKER, MQTT_PORT, 60)

def encender_luces():
    client.publish(MQTT_TOPIC, "ENCENDER_LUCES")
    print("Luces encendidas")

def apagar_luces():
    client.publish(MQTT_TOPIC, "APAGAR_LUCES")
    print("Luces apagadas")

# Ejemplo de uso
if __name__ == "__main__":
    encender_luces()
    time.sleep(5)
    apagar_luces()
```

Anexo C: Capturas de Pantalla de la Plataforma IoT

- **Captura 1:** Dashboard de ThingSpeak con Gráficos de Temperatura y Humedad.
- **Captura 2:** Dashboard de Blynk con Widgets de Monitoreo.
- **Captura 3:** Visualización en Tiempo Real de Datos en ThingSpeak.
- **Captura 4:** Indicadores de Estado en Blynk para Dispositivos Controlados.

Anexo D: Manual de Usuario de la Estación Meteorológica

- **Instrucciones de Inicio:**

- **Instalación de Dependencias:** Cómo instalar las bibliotecas necesarias.

```
sudo apt update
sudo apt install python3-pip -y
pip3 install Adafruit_DHT requests paho-mqtt
```

- **Configuración del Sistema:**
- **Raspberry Pi:**
- Conectar los sensores correctamente.
- Configurar la plataforma IoT (ThingSpeak) con la API Key.
- **Arduino:**
- Configurar la conexión Wi-Fi en el script de Arduino.
- Asegurar que el broker MQTT está activo y accesible.
- **Uso de la Estación Meteorológica:**
- **Lectura de Datos:** Ejecutar el script de Python para iniciar la recolección y transmisión de datos.

```
python3 enviar_datos.py
```

- **Visualización de Datos:** Acceder a la plataforma IoT para ver los gráficos en tiempo real.
- **Solución de Problemas:**
- **Problema:** Sensores no leen datos correctamente.
- **Solución:** Verificar las conexiones físicas y asegurar que los pines están correctamente configurados.
- **Problema:** Datos no se envían a la plataforma IoT.
- **Solución:** Verificar la conexión a internet y que la API Key está correcta.
- **Problema:** Error en la ejecución del script de Python.
- **Solución:** Revisar el código en busca de errores de sintaxis o librerías faltantes.
- **Mantenimiento del Sistema:**

- **Actualización de Software:** Cómo actualizar los scripts y bibliotecas.

```
pip3 install --upgrade Adafruit_DHT requests paho-mqtt
```

- **Revisión de Hardware:** Inspección periódica de conexiones y componentes electrónicos para asegurar su buen funcionamiento.

Mejoras y Recomendaciones Adicionales (Futuras)

1. **Optimización del Reconocimiento de Voz:**
    - **Filtros de Audio:** Implementar filtros de ruido en el preprocesamiento del audio para mejorar la precisión del reconocimiento.
    - **Comandos Personalizados:** Crear un diccionario de comandos personalizados para adaptarse mejor al entorno específico de uso.

2. **Seguridad del Sistema:**
    - **Protección de Datos:** Asegurar que los datos transmitidos a la plataforma IoT estén protegidos.
    - **Acceso Controlado:** Establecer mecanismos de autenticación para acceder a la plataforma IoT y al sistema de monitoreo.

3. **Interfaz de Usuario Mejorada:**
    - **Aplicación Gráfica:** Desarrollar una interfaz gráfica con `tkinter` para facilitar el uso del sistema.
    - **Feedback Visual y Sonoro:** Integrar señales visuales (como luces LED) o sonoras para confirmar la recepción y transmisión de datos.

4. **Expansión de Funcionalidades:**
    - **Control de Otros Dispositivos:** Añadir control de electrodomésticos adicionales como termostatos inteligentes, sistemas de riego, etc.
    - **Automatización Basada en Tiempo:** Programar acciones automáticas según horarios predefinidos (por ejemplo, activar ventiladores al superar cierta temperatura).

5. **Documentación y Formación Continua:**
    - **Guías de Usuario Detalladas:** Crear manuales más extensos y detallados para facilitar el uso y mantenimiento del sistema.
    - **Capacitaciones Adicionales:** Ofrecer sesiones de formación continuas para los estudiantes sobre nuevas tecnologías y actualizaciones del sistema.

Links de actualización pequeños

Link pequeño

https://sor.bz/eMFCw

QR pequeño

Links de actualización completos

Link completo

https://github.com/guideahon/Pro_tec_4_IOT

QR completo