Sistema de Monitoreo IoT con ESP32, Firebase y Control Remoto de LED.

Abril de 2025

Victor Anibal Murcia Calderon.

Escuela de Ingeniería de Sistemas y Computación, Maestría en Computación para el Desarrollo de Aplicaciones Inteligentes, Universidad del Valle. Cali, Colombia.

victor.murcia@correounivalle.edu.co

Resumen— Este trabajo presenta el desarrollo de un sistema IoT basado en un microcontrolador ESP32, capaz de registrar y visualizar en tiempo real variables ambientales simuladas mediante un potenciómetro, a través de una interfaz web. Además, se implementa el control remoto de un LED desde el dashboard del usuario. Se utiliza la plataforma Firebase Realtime Database para el almacenamiento de datos y la autenticación de usuarios mediante correo electrónico y contraseña. El sistema fue validado mediante pruebas funcionales, demostrando confiabilidad en la lectura de datos y respuesta inmediata al control del LED.

Palabras claves— ESP32, Firebase, IoT, LED, potenciómetro, base de datos en tiempo real, autenticación, interfaz web.

Abstract— This paper presents the development of an IoT system based on an ESP32 microcontroller, capable of real-time monitoring of simulated environmental variables using a potentiometer, through a web-based dashboard. In addition, remote control of an LED is implemented via the user interface. Firebase Realtime Database is used for data storage and user authentication via email and password. The system was validated through functional tests, showing reliability in data acquisition and immediate response to LED control.

I. Introducción

El Internet de las Cosas (IoT) ha revolucionado la forma en que los dispositivos interactúan con su entorno, permitiendo la recolección y análisis de datos en tiempo real. Este proyecto tiene como objetivo desarrollar un sistema de monitoreo de variables ambientales simulado basado en ESP32 y Firebase Realtime Database, incorporando además el control remoto de un LED.

II. METODOLOGIA

Se utilizó un ESP32 conectado a un potenciómetro para generar una señal analógica en el pin GPIO34. A partir de esta lectura se calcularon valores simulados de temperatura, humedad y presión. El conectado al pin GPIO14 LED fue microcontrolador. El sistema se conectó a Firebase Realtime Database utilizando la librería Firebase ESP Client, con autenticación mediante correo electrónico y contraseña.

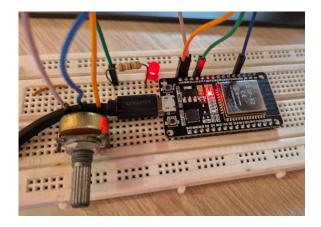


Fig. 1. Circuito del sistema con ESP32, LED y potenciómetro.

El frontend fue desarrollado en HTML, CSS y JavaScript, integrando Firebase SDK para leer y escribir datos en tiempo real. Se empleó la librería Highcharts para graficar la temperatura de forma continua. La API key del proyecto fue obtenida desde la consola de Firebase como se muestra en la Fig. 2.



Fig. 2. Configuración del proyecto Firebase y obtención de la API Key.

Se siguió un proceso detallado para configurar Firebase, incluyendo:

- → Creación del proyecto en Firebase Console.
- → Configuración de Firebase Hosting y despliegue del proyecto con firebase deploy (ver Fig. 3).
- → Creación de la base de datos Realtime Database.
- → Definición de reglas de seguridad:

```
PROBLEMS ① OUTPUT DEBUG CONSOLE TERMINAL PORTS

user@userPC:~/Desktop/firebaseyoutube$ firebase deploy

=== Deploying to 'esp32-d0b93'...

i deploying database, hosting
i database: rules syntax for database esp32-d0b93-default-rtdb is valid
i hosting[esp32-d0b93]: beginning deploy...
i hosting[esp32-d0b93]: found 7 files in public
hosting[esp32-d0b93]: file upload complete
i database: rules for database esp32-d0b93-default-rtdb released successfully
hosting[esp32-d0b93]: finalizing version...
/ database: rules for database esp32-d0b93-default-rtdb released successfully
hosting[esp32-d0b93]: releasing new version...
/ hosting[esp32-d0b93]: release complete

Peploy complete!

Project Console: https://console.firebase.google.com/project/esp32-d0b93/overview
Hosting URL: https://esp32-d0b93.web.app
user@userPC:~/Desktop/firebaseyoutube$ firebase deploy
```

Fig. 3. Proceso de despliegue del proyecto en Firebase Hosting y generación de la URL pública.

Referencia al código Dashboard: [1] Referencia al código Arduino: [2]

III. PRUEBAS Y RESULTADOS

El sistema fue probado exitosamente. Desde el puerto serie del Arduino se observó el envío correcto de datos simulados (ver Fig. 4).

```
Output Serial Monitor X

Message [Enter to send message to 'DOIT ESP32 DEVKIT VI' on 'dev/tbyUSBO')

UA1A: UN

Writing Value: 265.50 on the following path: /UsersData/OvzbFZY3P6Y67LM6LqBzz9imCCa2/temperature
PASSED

PATH: /UsersData/OvzbFZY3P6Y67LM6LqBzz9imCCa2/temperature

TYPE: float

Writing Value: 531.00 on the following path: /UsersData/OvzbFZY3P6Y67LM6LqBzz9imCCa2/humidity
PASSED

PATH: /UsersData/OvzbFZY3P6Y67LM6LqBzz9imCCa2/humidity

TYPE: int

Writing Value: 796.50 on the following path: /UsersData/OvzbFZY3P6Y67LM6LqBzz9imCCa2/pressure
PASSED

PATH: /UsersData/OvzbFZY3P6Y67LM6LqBzz9imCCa2/pressure

TYPE: float

PATH: /UsersData/OvzbFZY3P6Y67LM6LqBzz9imCCa2/pressure

TYPE: float

TYPE: string

DATA: ON
```

Fig. 4. Puerto serial mostrando datos simulados enviados exitosamente a Firebase

La interfaz web muestra lecturas en tiempo real de temperatura, humedad y presión simuladas, además de una gráfica actualizada dinámicamente.

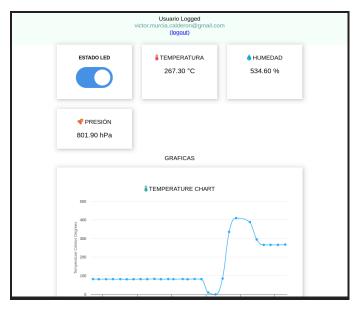


Fig. 5. Dashboard web con lecturas y control del LED en tiempo real.

El control del LED desde el dashboard web funcionó, con un retardo de un segundo aproximadamente y la gráfica de presión respondió correctamente a las variaciones generadas por el potenciómetro. El LED encendía o apagaba según el estado del botón web.

IV. CONCLUSIONES

- → Se logró implementar exitosamente un sistema IoT con capacidad de monitoreo en tiempo real y control remoto, utilizando herramientas accesibles y modernas como Firebase y ESP32. A pesar de no contar con un sensor ambiental real, el uso de un potenciómetro permite simular de manera efectiva la variación de variables ambientales.
- → Firebase proporciona una plataforma confiable para la implementación rápida de aplicaciones IoT pequeñas y medianas, facilitando tanto el almacenamiento como la autenticación de usuarios. La posibilidad de definir reglas de seguridad detalladas garantiza el acceso

- controlado y seguro a los datos, lo que lo convierte en una herramienta escalable para proyectos más complejos en el futuro.
- → La modularidad del sistema permite su ampliación futura a sensores reales u otros actuadores, y su integración en aplicaciones más robustas gracias a la flexibilidad de la arquitectura y la escalabilidad de Firebase.

REFERENCIAS

- [1] Google Drive Códigos Dashboard (HTML/JS): https://drive.google.com/drive/folders/13nQheIoEmYOW93f_TRNF 9BDLT1nBL4Sh?usp=sharing
- [2] Google Drive Código Arduino (ESP32): https://drive.google.com/file/d/1L9ZHQ-QqlWu4Mqbw7MSNYv85-l-Q3Nzv/view?usp=sharing
- [3] Firebase Realtime Database: https://firebase.google.com/products/realtime-database
- [4] Firebase ESP Client Library for Arduino: https://github.com/mobizt/Firebase-ESP-Client