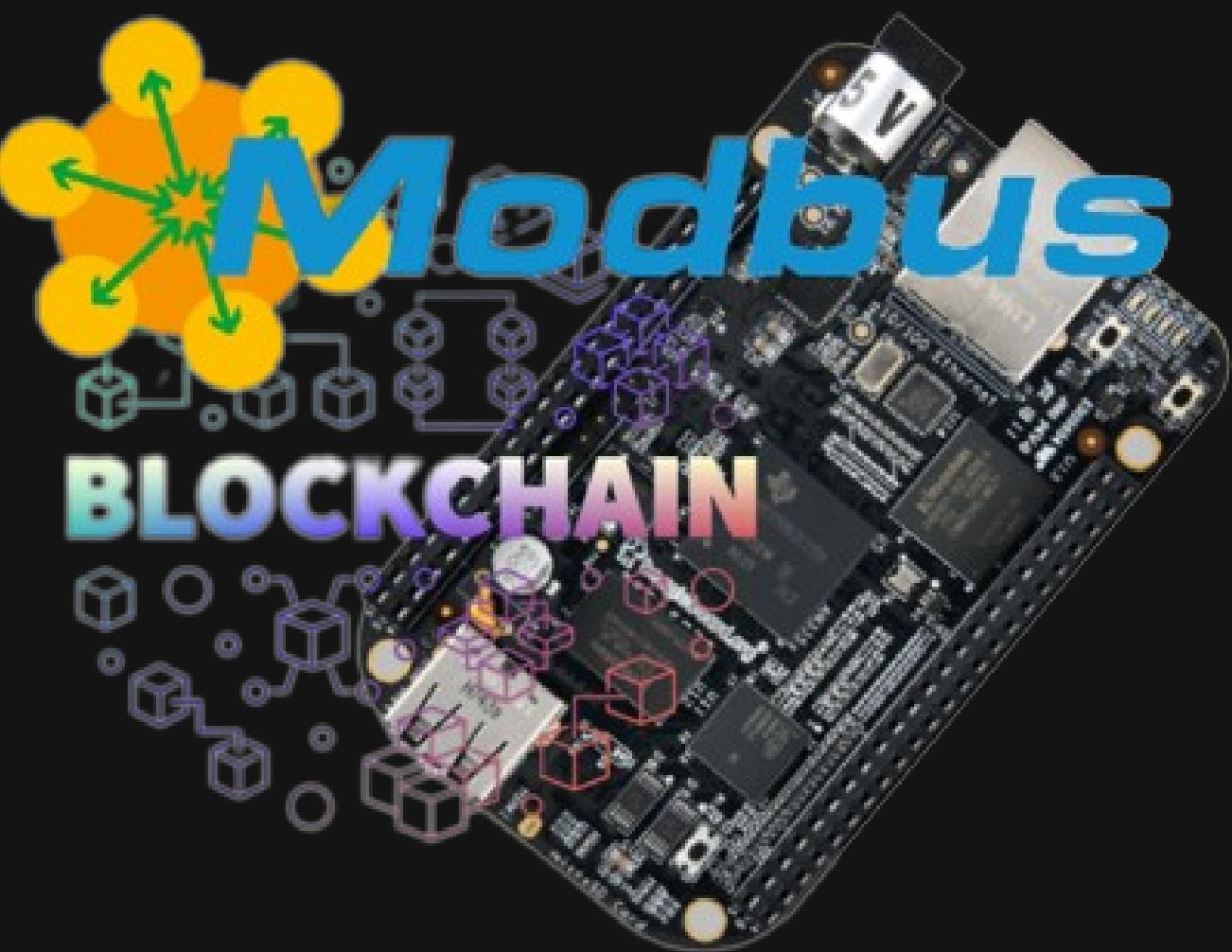


# MODBUS<sup>2</sup>CHAIN



**Studenti:**  
Alessandro Cavaliere  
Carmine Citro

# Panoramica del progetto

## Goals

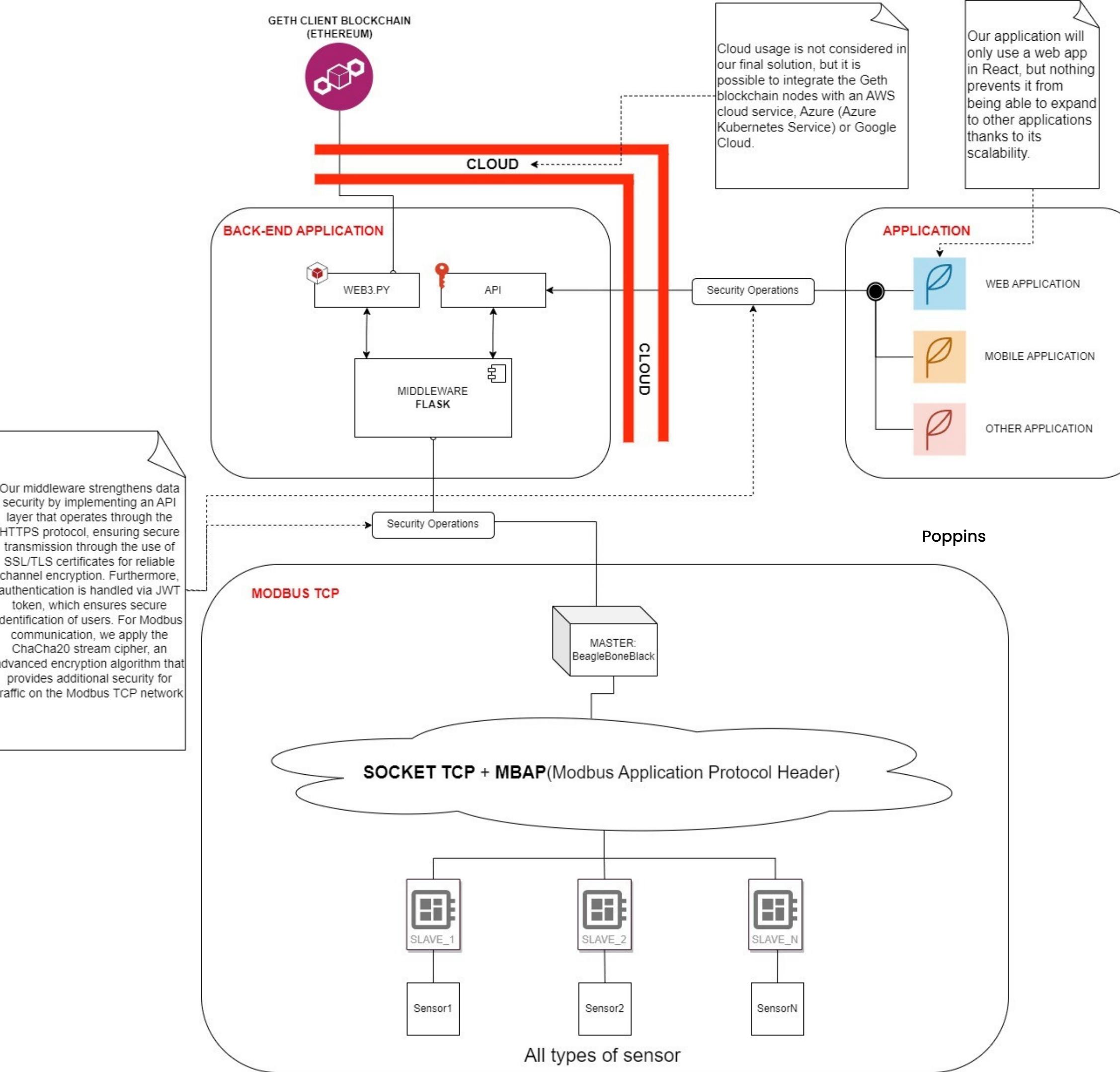
Creare un sistema efficiente per la raccolta e condivisione di dati ambientali focalizzato sulla simulazione di una casa domotica.

Utilizzo di sensori DHT11, sensori di movimento e cicalini.

Integrazione con la blockchain per la sicurezza dei dati



Diagramma dell'architettura di sistema di ModBus2chain



# Comunicazione IoT

## ModBus2Chain

Nel nostro progetto abbiamo utilizzato BeagleBone Black come master e Raspberry Pi Pico come slave, comunicando tramite protocollo Modbus TCP. La sicurezza dei dati scambiati è stata garantita dalla cifratura ChaCha20.

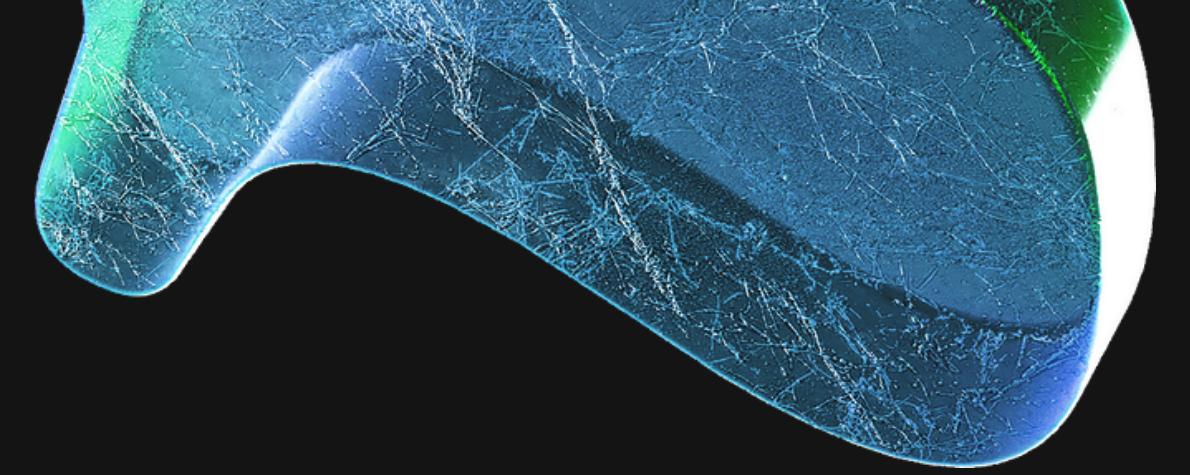
### Cos'è Modbus?

Modbus è un protocollo semplice e compatibile per l'automazione industriale, che collega controller, sensori e attuatori, e si adatta a reti più ampie nella sua variante Modbus TCP, mantenendo l'affidabilità originale.

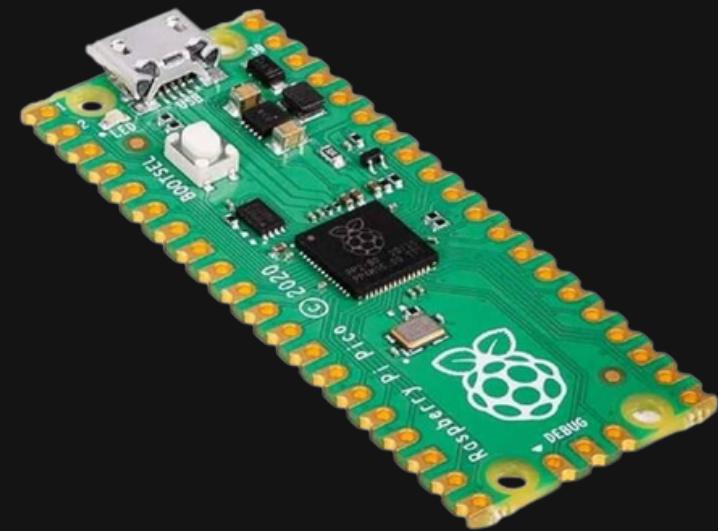


Beaglebone  
black

[Link specifiche tecniche.](#)



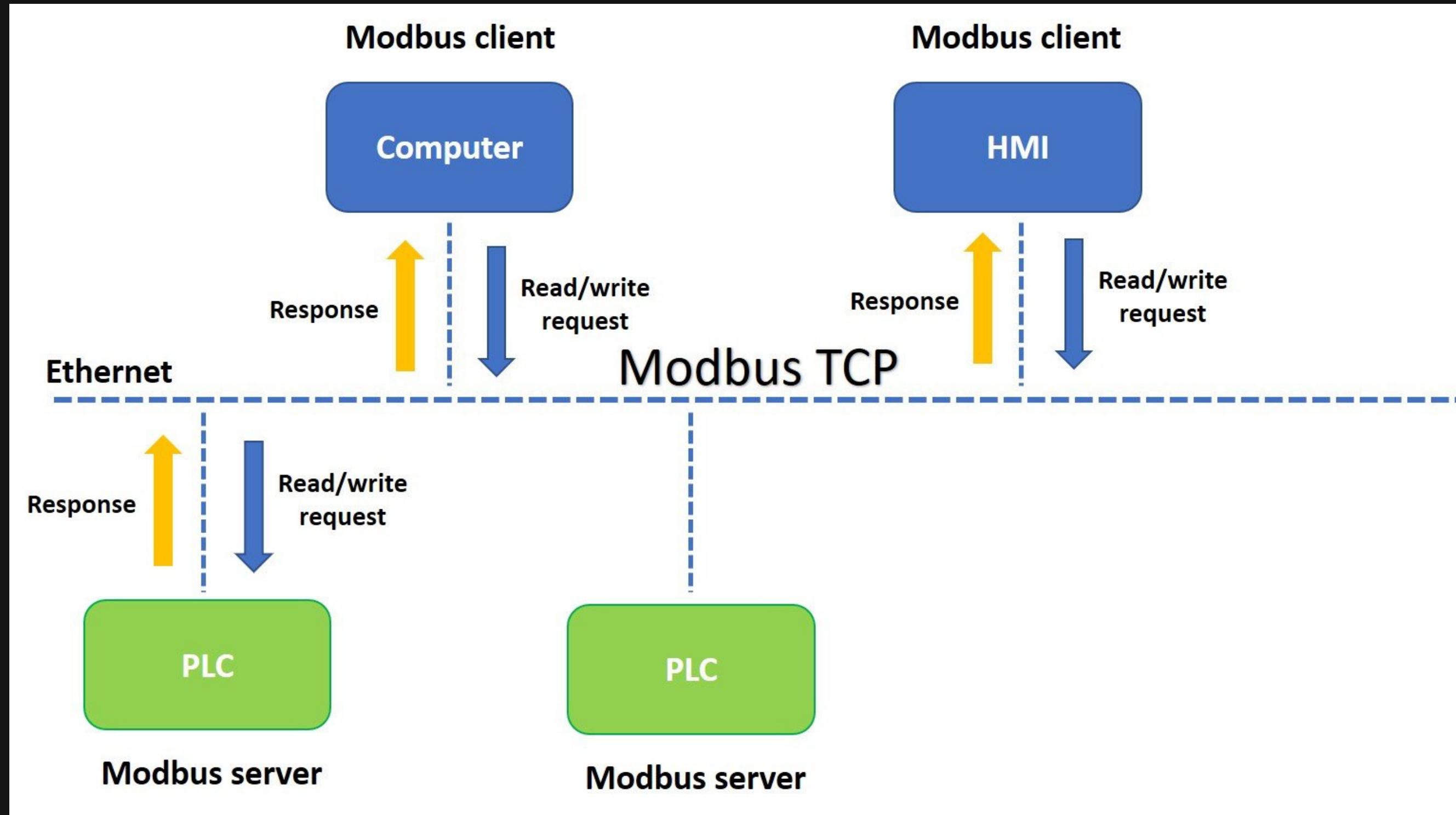
Raspberry Pi Pico W



[Link specifiche tecniche.](#)

# Il Protocollo Modbus TCP nel dettaglio

Grafico illustrativo  
del protocollo  
ModBus TCP



# Il Protocollo Modbus TCP nel dettaglio

## Vantaggi:

---

- **Standardizzazione:** Modbus è uno standard industriale de facto per la comunicazione tra dispositivi di automazione. Questo significa che produttori diversi possono produrre dispositivi che sono compatibili tra loro;
- **Affidabilità e Semplicità di Integrazione:** Modbus è un protocollo relativamente semplice; i dispositivi possono essere facilmente integrati in una rete Modbus TCP esistente senza la necessità di componenti aggiuntivi o di configurazioni complesse.
- **Basso Costo:** Molti dispositivi industriali includono il supporto Modbus di base senza licenze software aggiuntive o costi nascosti, rendendo il protocollo economicamente vantaggioso.

## Svantaggio:

---

Modbus TCP di per sé non include meccanismi di sicurezza integrati; non ha crittografia nativa, autenticazione o altre funzionalità di sicurezza che si trovano nei moderni protocolli di rete (come HTTPS). Questo lo rende vulnerabile ad attacchi man-in-the-middle e altri.

# Il Protocollo Modbus TCP nel dettaglio

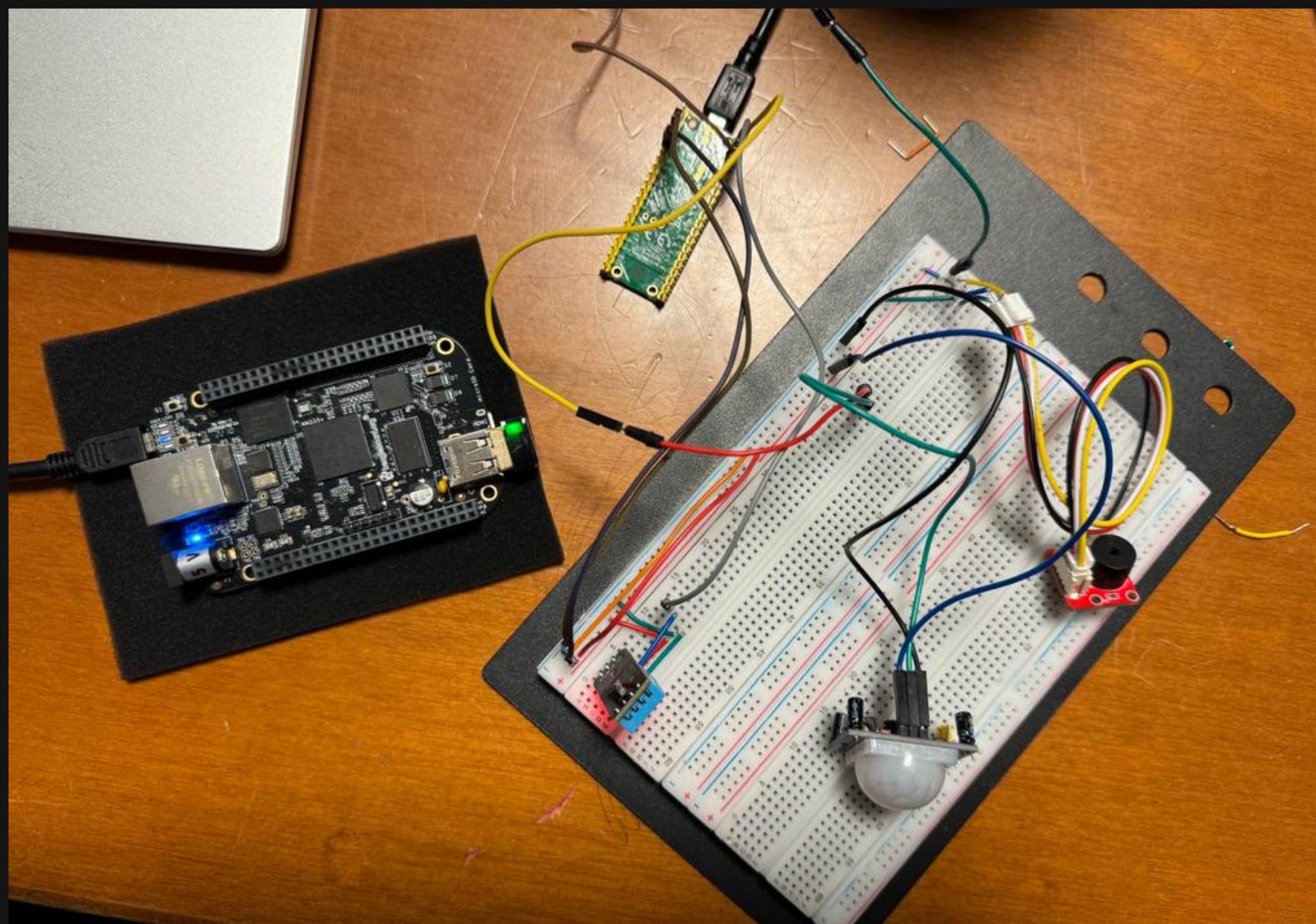
## Funzionalità Chiave:

- Incapsula dati Modbus in pacchetti TCP/IP per reti moderne; modello **TCP/IP**: Affidabile su reti IP, compatibile con infrastrutture esistenti.
- **Modello Client-Server**: Comunicazione tra sistemi di controllo/computer (client) e dispositivi di campo(server).
- **Unit Identifier nel MBAP Header**: Identifica più dispositivi in una rete.

## Benefici per L' IoT:

- **Connettività**: Perfetto per IoT; facilita connessione remota e monitoraggio.
- **Scalabilità**: Gestisce numerosi dispositivi, semplificando la rete.
- **Interoperabilità**: Compatibilità tra dispositivi di produttori diversi.

# Schema architettura del sistema



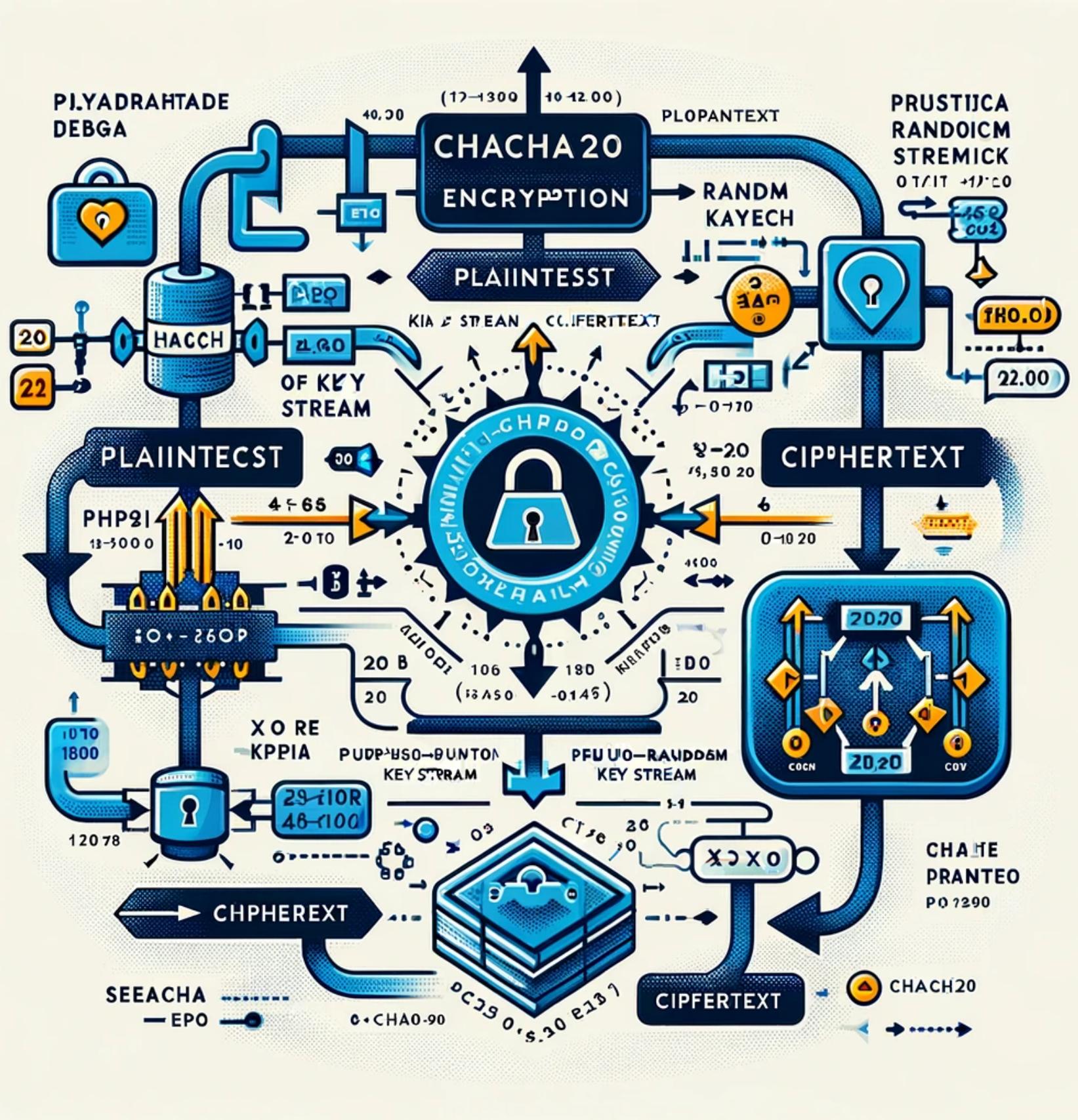
# Sensori e rilevamento Dati

In questo progetto utilizziamo il sensore di temperatura [DHT11](#), il sensore di movimento [HC-SR501 PIR](#), e il [Crowtail Buzzer 2.0](#). Il [BeagleBone Black](#), agendo come dispositivo master, richiede al [Raspberry Pi Pico W](#), nostro dispositivo slave, di misurare la temperatura. Utilizzando il DHT11 e il protocollo Modbus, il Raspberry Pi Pico W rileva temperatura e umidità. Inoltre, abbiamo implementato una soluzione domotica: se il DHT11 registra una temperatura oltre una certa soglia, si attiva il Crowtail Buzzer 2.0 come allarme. Similmente, l'allarme si attiva con la rilevazione di movimento da parte del sensore HC-SR501.

In Seguito tutti i link dei dispositivi per le specifiche tecniche

- [DHT11](#)
- [Sensore di movimento HC-SR501 PIR](#)
- [Crowtail Buzzer 2.0.](#)

# Cifratura con ChaCha20



## Cos'è Chacha20?

- ChaCha20 è un algoritmo di cifratura a flusso.
- Funziona mediante la generazione di un flusso di chiavi pseudo-casuali che viene poi combinato con i dati originali (**plaintext**) tramite l'operazione di XOR.
- Nel nostro progetto, per assicurare una trasmissione sicura dei dati tra master e slave, abbiamo scelto ChaCha20, superando la sfida di adattarlo alle limitazioni dei registri ModBus, che possono gestire solo 16 bit per messaggio. Grazie alla flessibilità del cifrario ChaCha20, siamo riusciti a crittografare i dati efficacemente, rispettando la dimensione dei registri ModBus.

# Integrazione con la Blockchain Ethereum (GETH Client)

Nel nostro progetto, abbiamo creato una blockchain privata con il client Geth di Ethereum e un algoritmo di consenso PoA. Abbiamo deployato uno smart contract in Solidity per notarizzare i dati IoT, garantendo autenticità e integrità in aree come monitoraggio ambientale e sicurezza. La blockchain privata offre più privacy e controllo sui dati rispetto alle blockchain pubbliche. Abbiamo anche implementato meccanismi personalizzati di autorizzazione e validazione, rafforzando la sicurezza e l'efficienza e rendendo il nostro ecosistema IoT più sicuro e affidabile.

## Vantaggi della Blockchain per l'Integrità dei Dati

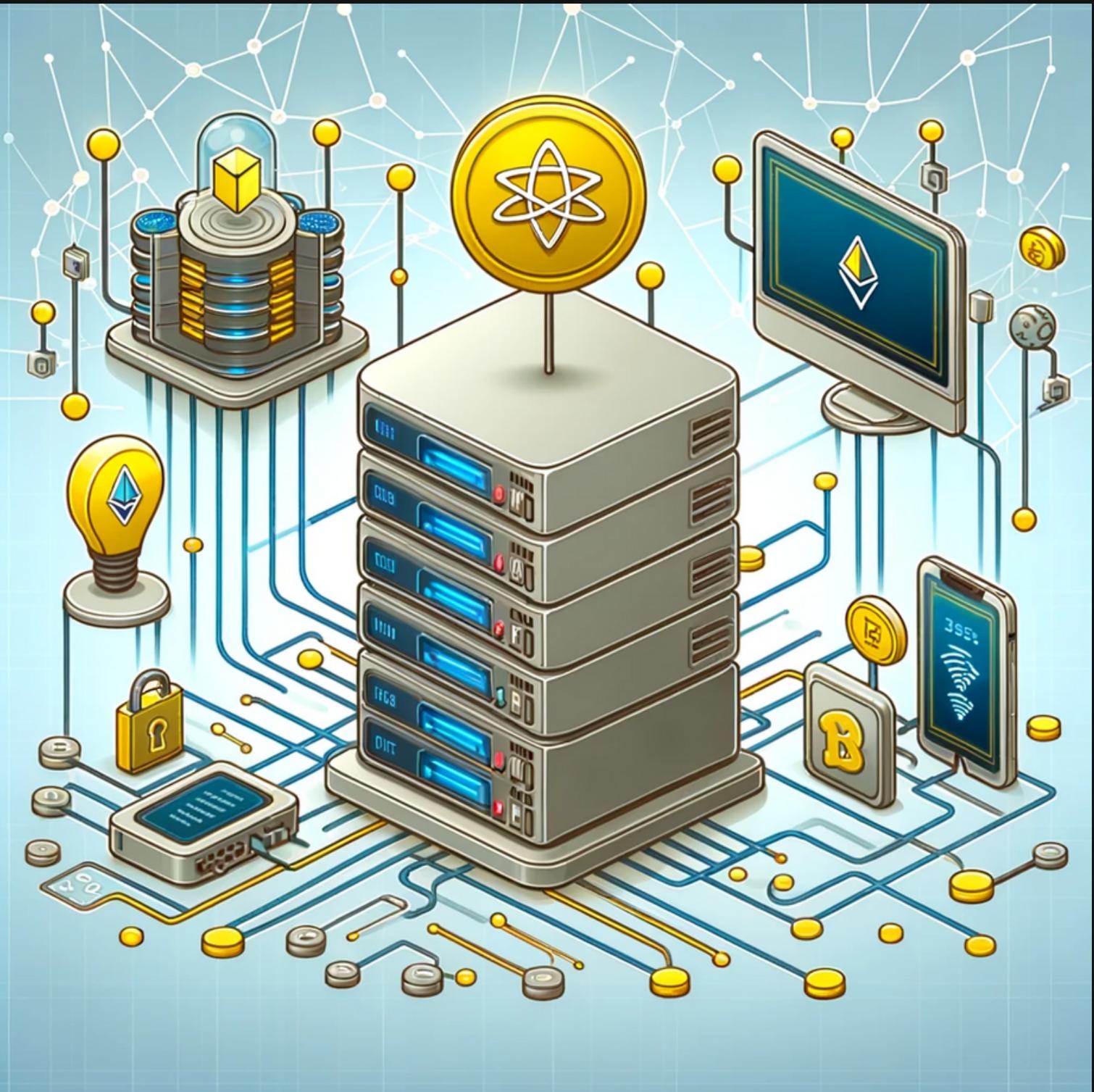
- **Immutabilità Rafforzata**
  - Dati inalterabili per integrità a lungo termine.
  - Fondamentale per compliance e sicurezza.
- **Sicurezza Affidabile**
  - Basata su account validatori fidati.
  - Resistenza ad attacchi specifici.
- **Prestazioni e Scalabilità Elevate**
  - Superiori a reti pubbliche.
  - Ideali per esigenze IoT.
- **Controllo Personalizzato**
  - Configurazione flessibile per specifiche IoT.
- **Privacy dei Dati**
  - Accesso limitato a partecipanti autorizzati..



# Integrazione IoT-Blockchain con Server Flask

Il server Flask è il fulcro del progetto, facilita la raccolta, il trasferimento e la gestione dei dati ambientali.

- **Middleware Avanzato in Flask**
  - Core del sistema in Python con Flask.
- **Flusso Dati IoT e Blockchain**
  - Avvio e gestione del prelievo dati ambientali.
  - Trasmissione sicura e immutabile alla blockchain.
- **Comunicazione Sicura via API HTTPS**
  - API criptate con SSL/TLS.
  - Comunicazione affidabile a interfaccia React.
- **Autenticazione JWT**
  - Accesso controllato e sicuro per utenti.
- **Blockchain Geth con PoA**
  - Ambiente privato, controllato e sicuro.
  - Validazione rapida e affidabile delle transazioni.



# RECAP

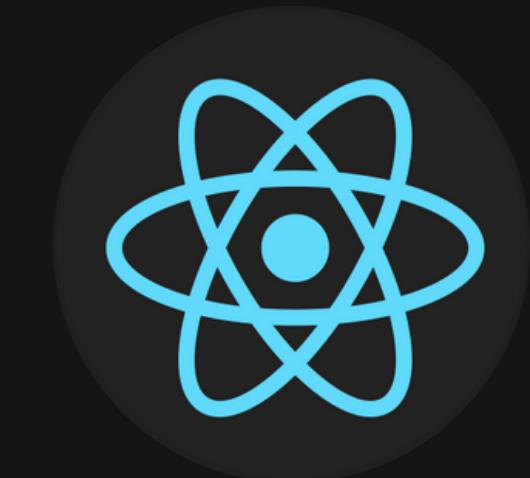
## Tecnologie Utilizzate



Go Ethereum



mongoDB®



# Casi D'uso ModBus2Chain

## Casi d'Uso Versatili

- Sicurezza Domestica e Industriale: Monitoraggio e controllo sicuro degli accessi.
- Monitoraggio Ambientale: Raccolta e gestione sicura dei dati ambientali nel tempo.
- Applicabile in vari settori per l'automazione e la gestione dei processi.



# Conclusioni e Miglioramenti

- **Sicurezza dei Dati Rafforzata:**  
Aggiungere livelli di sicurezza, come un architettura Multi-Factor Authentication (MFA).
- **Ottimizzazione esperienza utente:**
- Migliorare l'interfaccia utente in maniera da rendere possibile la visualizzazione dei dati in maniera più chiara.
- **Ampliamento della Gamma di Sensori.**
- **Interoperabilità con Tecnologie IoT:**  
Aumentare flessibilità e applicabilità.
- **Sostituzione Server Flask:**
- Per il monitoraggio in tempo reale dei sensori, si potrebbe sostituire quest'ultimo con **Quart**, il quale supporta in maniera nativa le operazioni asincrone.



Grazie per l'attenzione!

