

---

# MONITORAGGIO PORTA CON SENSORE ULTRASONICO E DASHBOARD WEB

---

PROGETTO DI RETI DI CALCOLATORI

STILATO E PROGETTATO DA  
**STEFANO ZIZZI**

*Università degli Studi di Urbino  
Corso di Laurea in Informatica Applicata*



MATRICOLA: 312793



# Indice

<b>1</b>	<b>Introduzione</b>	<b>5</b>
<b>2</b>	<b>Specifiche Tecniche</b>	<b>7</b>
2.1	ESP32 e Sensore Ultrasonico HC-SR04 . . . . .	7
2.1.1	ESP32 . . . . .	7
2.1.2	Sensore Ultrasonico HC-SR04 . . . . .	8
2.1.3	Connessioni . . . . .	8
2.2	Web Server . . . . .	8
2.2.1	Configurazione del Server . . . . .	8
2.2.2	API REST . . . . .	9
2.2.3	Struttura e Funzionalità . . . . .	9
2.2.4	Implementazione dell'API con PHP . . . . .	9
2.2.5	Gestione degli Errori . . . . .	9
2.2.6	Sicurezza . . . . .	10
2.3	Database . . . . .	10
2.3.1	MySQL . . . . .	10
2.3.2	Interazione con il Database . . . . .	11
2.4	Dashboard Web . . . . .	11
2.4.1	Struttura e Tecnologie Utilizzate . . . . .	11
2.4.2	Funzionalità Principali . . . . .	11
2.4.3	Interfaccia Utente . . . . .	11
2.4.4	Implementazione Tecnica . . . . .	12
<b>3</b>	<b>Sfide e Difficoltà Incontrate</b>	<b>13</b>
3.1	Configurazione del Web Server e del Database MySQL . . . . .	13
3.2	Architettura Client-Server . . . . .	13
3.3	Interferenze del Sensore Ultrasonico . . . . .	13
3.4	Crittografia dei Messaggi con SSL . . . . .	14
3.5	Conclusioni sulle Sfide Incontrate . . . . .	14



# Capitolo 1

## Introduzione

In questa relazione verrà illustrato il progetto di un **sistema di monitoraggio della distanza basato sull'utilizzo di un sensore ultrasonico HC-SR04**, con visualizzazione dei dati in tempo reale tramite una dashboard web. Il progetto nasce dalla necessità di creare un sistema semplice ed efficace per monitorare la distanza di una porta, con l'obiettivo di visualizzare i dati raccolti in modo intuitivo e accessibile. Questa soluzione risponde a esigenze pratiche in contesti domestici e industriali, dove è fondamentale avere un controllo preciso e costante di aperture e chiusure, garantendo sicurezza e automazione.

La scelta di questo argomento è motivata dal mio profondo interesse per l'Internet of Things (IoT) e l'automazione, nonché dal desiderio di approfondire le mie competenze nella programmazione di microcontrollori e nello sviluppo web. Viviamo in un'epoca in cui l'interconnessione tra dispositivi e la capacità di raccogliere e analizzare dati in tempo reale sono cruciali per migliorare la qualità della vita e l'efficienza dei processi produttivi. Gli obiettivi principali del progetto sono i seguenti:

- **Realizzare un sistema di rilevamento della distanza:** Utilizzando il **sensore HC-SR04** e un **microcontrollore ESP32**, il sistema sarà in grado di misurare la distanza di un oggetto, come una porta, e rilevare eventuali cambiamenti.
- **Trasmettere i dati rilevati a un server remoto:** I dati raccolti dal sensore verranno inviati a un server remoto, dove saranno memorizzati per un'analisi successiva. Questa funzionalità permette di monitorare le attività in tempo reale e di mantenere uno storico dei dati per future analisi e report.
- **Visualizzare i dati in una dashboard web:** Una dashboard web permetterà di monitorare i dati in tempo reale, visualizzare lo storico delle misurazioni, applicare filtri per un'analisi dettagliata e attivare o disattivare il sensore. La visualizzazione grafica e l'interattività della dashboard sono fondamentali per rendere i dati facilmente comprensibili e utili per gli utenti finali.

Il progetto si avvale di diverse tecnologie e componenti software per garantire il corretto funzionamento del sistema. Al fine di realizzare il progetto, ho deciso di ospitare il server remoto sul mio laptop con sistema operativo Linux Ubuntu, utilizzando una rete locale e il **server web Apache**. Questa scelta è stata dettata dalla necessità di avere un ambiente di sviluppo controllato e accessibile, senza l'obbligo di rendere il dispositivo continuamente raggiungibile dall'esterno. Questa configurazione offre la flessibilità necessaria per testare e sviluppare il sistema in un ambiente sicuro e ben conosciuto.

Per quanto riguarda il database, la mia scelta è ricaduta su **MySQL**, in quanto rappresenta una soluzione semplice e intuitiva per un novizio. MySQL è un sistema di gestione di database relazionale molto diffuso, che offre una vasta documentazione e una comunità di supporto attiva, rendendolo ideale per chi si avvicina per la prima volta a questo tipo di tecnologia. La capacità di gestire grandi quantità di dati e di eseguire query complesse in modo efficiente lo rende particolarmente adatto per questo progetto.

Il processo di sviluppo del progetto ha comportato diverse fasi, ognuna delle quali ha richiesto l'applicazione di competenze tecniche specifiche e la risoluzione di problemi pratici. Nella fase iniziale, è

stato necessario configurare correttamente l'hardware, assicurando che il sensore HC-SR04 fosse collegato in modo appropriato al microcontrollore ESP32. Successivamente, è stata sviluppata la logica di programmazione per raccogliere e trasmettere i dati, utilizzando il **linguaggio Arduino** e le librerie specifiche per Arduino e ESP32.

La fase successiva ha coinvolto lo sviluppo del backend del server, utilizzando PHP per gestire le richieste di dati e interagire con il database MySQL. È stato implementato un sistema di API per permettere la comunicazione tra il microcontrollore e il server, garantendo la sicurezza e l'integrità dei dati trasmessi.

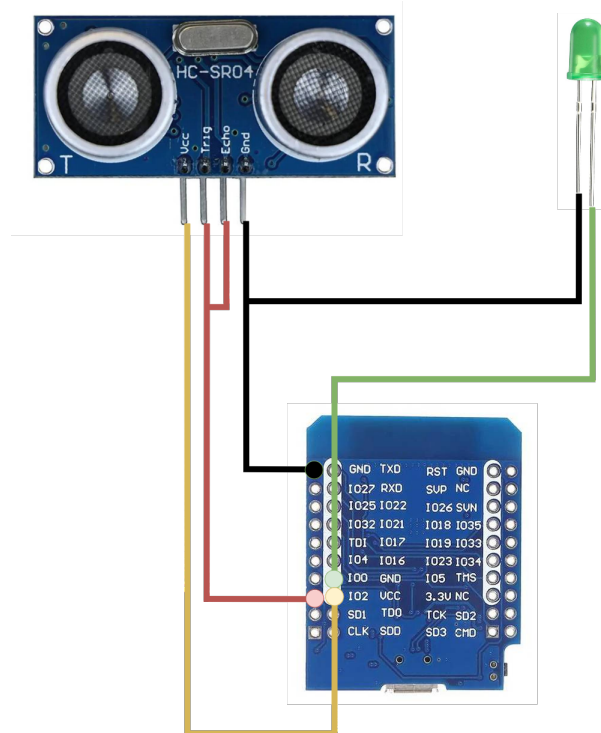
Infine, la creazione della dashboard web ha richiesto l'utilizzo di **HTML**, **CSS** e **JavaScript**, oltre a librerie come **Bootstrap** per il design responsive e **Chart.js** per la visualizzazione dei dati. La dashboard è stata progettata per essere *user-friendly* e offrire funzionalità avanzate, come la possibilità di applicare filtri ai dati e di visualizzare grafici interattivi.

## Capitolo 2

# Specifiche Tecniche

In questa sezione verranno descritte in dettaglio le specifiche tecniche del sistema di monitoraggio della distanza sviluppato. Il sistema è costituito da tre componenti principali: il microcontrollore ESP32 con il sensore ultrasonico HC-SR04, il server web e il database. Ognuno di questi componenti è stato progettato e configurato per garantire un funzionamento efficiente e affidabile del sistema complessivo.

### 2.1 ESP32 e Sensore Ultrasonico HC-SR04



**Figura 2.1:** Connessioni Elettriche HC-SR04 e ESP32

#### 2.1.1 ESP32

L'**ESP32** è un microcontrollore potente e versatile, ampiamente utilizzato in applicazioni IoT grazie alle sue capacità di elaborazione e connettività wireless integrate. Per questo progetto, l'ESP32 è stato scelto per le seguenti caratteristiche:

- **Doppio core Tensilica LX6:** Garantisce alte prestazioni e multitasking.
- **Connettività Wi-Fi e Bluetooth integrata:** Permette di trasmettere dati in modalità wireless a un server remoto.

- **Ampia gamma di GPIO:** Supporta la connessione di vari sensori e attuatori.
- **Supporto per librerie Arduino:** Facilita lo sviluppo del firmware.

### 2.1.2 Sensore Ultrasonico HC-SR04

Il sensore **HC-SR04** è utilizzato per misurare la distanza tramite onde ultrasoniche. Le specifiche tecniche del sensore includono:

- **Tensione di funzionamento:** 5V DC.
- **Corrente operativa:** 15mA.
- **Frequenza ultrasonica:** 40kHz.
- **Distanza di rilevamento:** 2cm - 400cm.
- **Risoluzione:** 0.3cm.
- **Angolo di misurazione:** 15 gradi.

### 2.1.3 Connessioni

Le connessioni tra l'ESP32 e il sensore HC-SR04 sono state configurate come segue:

- **GND del sensore a GND dell'ESP32.**
- **VCC del sensore a VCC dell'ESP32 (5V).**
- **Trigger del sensore a GPIO 2 dell'ESP32.**
- **Echo del sensore a GPIO 2 dell'ESP32.**

Un LED è collegato per indicare lo stato di apertura della porta:

- **GND del LED a GND dell'ESP32.**
- **Anodo del LED a GPIO 0 dell'ESP32.**

## 2.2 Web Server

### 2.2.1 Configurazione del Server

Il server web è stato ospitato su un laptop con sistema operativo *Linux Ubuntu*, utilizzando il server web **Apache**. Apache è una scelta comune per applicazioni web grazie alla sua affidabilità, sicurezza e flessibilità. Le specifiche del server includono:

- **Sistema Operativo:** *Linux Ubuntu 20.04*.
- **Server Web:** *Apache 2.4*.
- **PHP:** Versione 7.4, utilizzato per la logica di backend e l'interazione con il database.
- **Configurazione di rete:** Il server è configurato per operare su una rete locale, garantendo l'accesso rapido e sicuro durante le fasi di sviluppo e test.



## 2.2.2 API REST

### 2.2.3 Struttura e Funzionalità

Per consentire la comunicazione tra l'ESP32 e il server web, è stata sviluppata un'API REST utilizzando *PHP*. L'API è progettata per gestire le operazioni di invio e recupero dei dati dal microcontrollore, oltre a fornire funzionalità per il controllo remoto del sensore. Le principali funzionalità dell'API includono:

- **Endpoint per l'invio dei dati:** L'ESP32 invia le misurazioni al server tramite una richiesta *HTTP POST*. L'API riceve questi dati e li salva nel database MySQL.
- **Endpoint per il recupero dei dati:** La dashboard web richiama i dati dal server tramite richieste *HTTP GET*, permettendo di visualizzare sia i dati in tempo reale che lo storico delle misurazioni.
- **Endpoint per il controllo dello stato del sensore:** Permette di abilitare o disabilitare il sensore in remoto. Questo è utile per gestire il sensore senza dover accedere fisicamente all'ESP32.

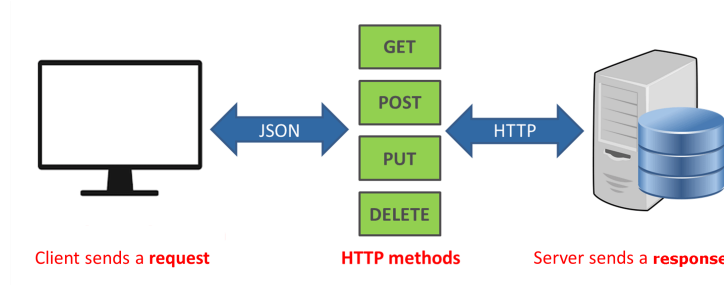


Figura 2.2: API RESTful

### 2.2.4 Implementazione dell'API con PHP

L'implementazione dell'API REST in *PHP* coinvolge diversi script che gestiscono le richieste *HTTP*, interagiscono con il database e restituiscono le risposte appropriate. Di seguito viene fornita una descrizione dettagliata delle principali operazioni:

- **Ricezione e Salvataggio dei Dati:** Quando l'ESP32 invia una misurazione tramite una richiesta *POST*, l'API riceve i dati in formato *JSON*. Lo script *PHP* decodifica il *JSON*, estrae le informazioni rilevanti (come la distanza misurata) e le inserisce nel database MySQL. Inoltre, lo script può aggiornare lo stato del LED basandosi sulla distanza rilevata.
- **Recupero dei Dati:** Quando la dashboard web invia una richiesta *GET* per ottenere i dati delle misurazioni, lo script *PHP* esegue una query al database per recuperare le informazioni richieste. I dati vengono quindi codificati in formato *JSON* e inviati come risposta alla richiesta. Questo permette alla dashboard di visualizzare i dati aggiornati senza necessità di ricaricare l'intera pagina.
- **Controllo dello Stato del Sensore:** Per gestire lo stato del sensore (abilitato/disabilitato), l'API include un endpoint che riceve una richiesta *POST* con il nuovo stato del sensore. Lo script *PHP* aggiorna un file di stato sul server per riflettere il cambiamento. Questo meccanismo consente di controllare il sensore in remoto, modificando il comportamento dell'ESP32 in base al nuovo stato.

### 2.2.5 Gestione degli Errori

L'API è progettata per gestire in modo robusto eventuali errori che possono verificarsi durante le operazioni. Ad esempio:

- Se una richiesta POST non include tutti i dati necessari, lo script PHP restituisce un messaggio di errore dettagliato e uno stato *HTTP 400 (Bad Request)*.
- Se si verifica un errore durante l’inserimento dei dati nel database, viene restituito uno stato *HTTP 500 (Internal Server Error)* con un messaggio di errore.
- Le risposte dell’API includono sempre un campo di stato che indica il successo o il fallimento dell’operazione richiesta, permettendo al client di gestire adeguatamente le diverse situazioni.

2.2.6 Sicurezza

Per garantire la sicurezza della comunicazione tra l’ESP32 e il server, sono state implementate le seguenti misure:

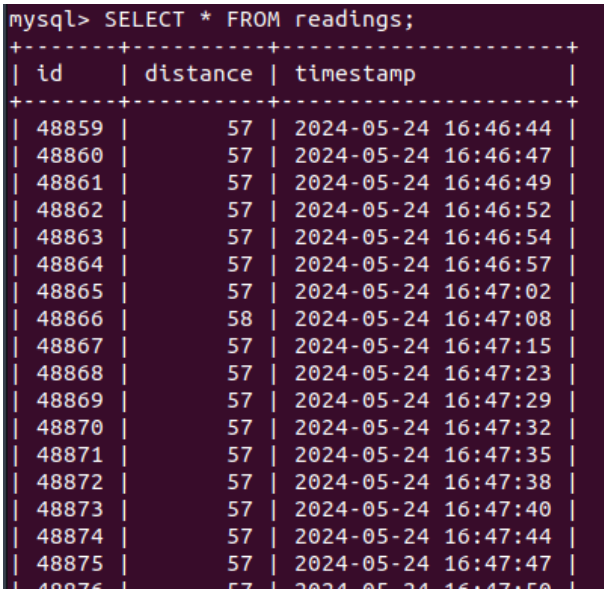
- **Crittografia SSL:** Le comunicazioni sono protette utilizzando HTTPS (con certificato auto-firmato) per prevenire intercettazioni e manomissioni dei dati. La configurazione HTTPS è stata realizzata utilizzando **OpenSSL**, che ha permesso di generare e gestire i certificati necessari per la crittografia.

2.3 Database

2.3.1 MySQL

Il database utilizzato per questo progetto è **MySQL**, un sistema di gestione di database relazionale open-source, noto per la sua affidabilità e facilità d’uso. Le specifiche del database includono:

- **Sistema di Gestione:** *MySQL 8.0*.
- **Struttura del database:** Una tabella principale denominata *readings* per memorizzare le misurazioni del sensore.
- **Campi della tabella:**
  - **id:** Identificativo univoco (INT, AUTO\_INCREMENT, PRIMARY KEY).
  - **timestamp:** Data e ora della misurazione (TIMESTAMP).
  - **distance:** Distanza rilevata dal sensore (FLOAT).



```
mysql> SELECT * FROM readings;
```

id	distance	timestamp
48859	57	2024-05-24 16:46:44
48860	57	2024-05-24 16:46:47
48861	57	2024-05-24 16:46:49
48862	57	2024-05-24 16:46:52
48863	57	2024-05-24 16:46:54
48864	57	2024-05-24 16:46:57
48865	57	2024-05-24 16:47:02
48866	58	2024-05-24 16:47:08
48867	57	2024-05-24 16:47:15
48868	57	2024-05-24 16:47:23
48869	57	2024-05-24 16:47:29
48870	57	2024-05-24 16:47:32
48871	57	2024-05-24 16:47:35
48872	57	2024-05-24 16:47:38
48873	57	2024-05-24 16:47:40
48874	57	2024-05-24 16:47:44
48875	57	2024-05-24 16:47:47
48876	57	2024-05-24 16:47:50

Figura 2.3: Database MySQL

### 2.3.2 Interazione con il Database

L'interazione con il database avviene tramite script *PHP* che eseguono operazioni di lettura e scrittura:

- **Inserimento dei dati:** Quando l'ESP32 invia una nuova misurazione, il server la salva nel database.
- **Recupero dei dati:** La dashboard web richiede i dati dal database per visualizzare le misurazioni in tempo reale e lo storico.

## 2.4 Dashboard Web

### 2.4.1 Struttura e Tecnologie Utilizzate

La dashboard web è stata sviluppata per fornire una visualizzazione intuitiva e interattiva dei dati raccolti dal sensore ultrasonico. Le tecnologie principali utilizzate per la realizzazione della dashboard sono:

- **HTML5 e CSS3:** Per la struttura e lo stile della pagina web.
- **Bootstrap:** Per il design responsivo e l'interfaccia utente moderna.
- **JavaScript:** Per la logica di interazione e la gestione dinamica dei contenuti.
- **Chart.js:** Per la visualizzazione grafica dei dati.
- **AJAX:** Per l'aggiornamento in tempo reale dei dati senza ricaricare la pagina.

### 2.4.2 Funzionalità Principali

La dashboard offre diverse funzionalità per il monitoraggio e l'analisi dei dati:

- **Visualizzazione in tempo reale:** I dati raccolti dal sensore vengono aggiornati in tempo reale, permettendo di monitorare costantemente lo stato della porta.
- **Grafici interattivi:** Utilizzando *Chart.js*, i dati vengono presentati in grafici lineari che permettono di visualizzare facilmente le variazioni di distanza nel tempo.
- **Storico delle misurazioni:** È possibile consultare lo storico delle misurazioni effettuate, con la possibilità di filtrare i dati per data e intervallo di distanza.
- **Filtri avanzati:** La dashboard consente di applicare filtri sui dati visualizzati, permettendo di analizzare specifici intervalli di tempo o di distanza.
- **Modalità scura:** Un'opzione per attivare la modalità scura, migliorando la leggibilità in ambienti con poca luce.

### 2.4.3 Interfaccia Utente

L'interfaccia utente della dashboard è progettata per essere user-friendly e accessibile. Le componenti principali dell'interfaccia includono:

- **Barra di navigazione:** Contiene i controlli per attivare/disattivare il monitoraggio in tempo reale e la modalità scura.
- **Tasto di Accensione e Spegnimento:** Utilizzato per attivare/disattivare il sensore da remoto.
- **Grafico della distanza:** Visualizza i dati in tempo reale e storici utilizzando un grafico lineare.
- **Tabella dello storico:** Elenca tutte le misurazioni raccolte, con opzioni per filtrare e ordinare i dati.
- **Controlli di filtro:** Permettono di applicare filtri sui dati visualizzati nel grafico e nella tabella.

- **Paginazione:** Gestisce la visualizzazione dei dati storici in pagine, migliorando la navigazione e la consultazione dei dati.

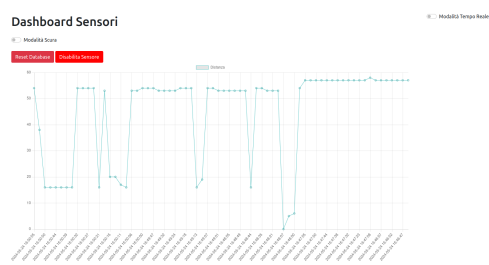


Figura 2.4: Grafico e Controlli

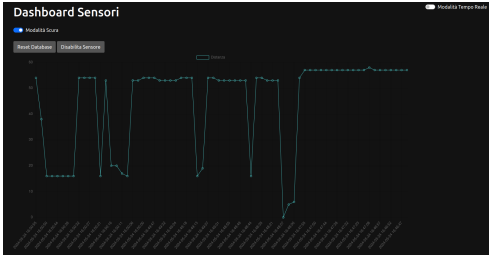


Figura 2.5: Modalità Scura

Rilevamento Apertura Porta	
Timestamp	Distanza
2024-05-24 16:50:53	38
2024-05-24 16:50:50	16
2024-05-24 16:50:48	16
2024-05-24 16:50:44	16
2024-05-24 16:50:42	16
2024-05-24 16:50:39	16
2024-05-24 16:50:36	16
2024-05-24 16:50:21	16
2024-05-24 16:50:16	20
2024-05-24 16:50:14	20
First Previous 1 2 Next Last	

Figura 2.6: Tabella Apertura Porta

Storico delle Misurazioni	
Timestamp	Distanza
2024-05-24 16:50:55	54
2024-05-24 16:50:53	38
2024-05-24 16:50:50	16
2024-05-24 16:50:48	16
2024-05-24 16:50:44	16
2024-05-24 16:50:42	16
2024-05-24 16:50:39	16
2024-05-24 16:50:36	16
2024-05-24 16:50:32	54
2024-05-24 16:50:30	54
First Previous 1 2 3 4 5 Next Last	
mm/dd/yyyy	
Distanza minima	
Distanza massima	
Applica Filtri	

Figura 2.7: Storico Misurazioni con Filtri

### 2.4.4 Implementazione Tecnica

La dashboard è stata implementata con un approccio modulare, separando chiaramente la logica di presentazione, la logica di interazione e la gestione dei dati. Alcuni dettagli tecnici dell’implementazione includono:

- **Utilizzo di AJAX per l’aggiornamento dei dati:** Le richieste *AJAX* vengono inviate periodicamente al server per ottenere i dati più recenti senza ricaricare la pagina.
- **Integrazione con Chart.js:** I dati ottenuti vengono elaborati e visualizzati utilizzando i grafici di *Chart.js*, che offrono interattività e personalizzazione.
- **Gestione dei filtri con JavaScript:** I filtri applicati dall’utente vengono gestiti tramite *JavaScript*, aggiornando dinamicamente la visualizzazione dei dati nel grafico e nella tabella.

## Capitolo 3

# Sfide e Difficoltà Incontrate

Lo sviluppo di questo progetto ha presentato diverse sfide che hanno richiesto soluzioni creative e un'approfondita comprensione delle tecnologie coinvolte. Di seguito sono descritte le principali difficoltà incontrate durante il processo di realizzazione del sistema di monitoraggio della distanza con sensore ultrasonico, server web, database e dashboard.

### 3.1 Configurazione del Web Server e del Database MySQL

Una delle prime sfide affrontate è stata la configurazione del web server e del database MySQL. Non avendo mai lavorato prima con un web server e con MySQL, ho dovuto dedicare molto tempo alla ricerca e alla comprensione dei concetti fondamentali. La creazione e configurazione di questi componenti non è stata semplice, in particolare:

- **Installazione e Configurazione di Apache:** Ho dovuto imparare a installare Apache su un sistema Linux Ubuntu e configurarlo correttamente per ospitare il server web. Questo ha comportato la modifica dei file di configurazione e la gestione dei permessi di accesso.
- **Creazione e Gestione del Database MySQL:** La creazione di un database MySQL, la definizione delle tabelle e la scrittura delle query SQL per interagire con i dati sono state attività complesse. Ho dovuto apprendere come gestire i backup e le operazioni di ripristino.

### 3.2 Architettura Client-Server

Un'altra sfida significativa è stata la progettazione e implementazione di un'architettura client-server. Non avendo mai creato un'architettura di questo tipo, ho dovuto comprendere come strutturare la comunicazione tra il client (ESP32) e il server. Le principali difficoltà incontrate sono state:

- **Definizione dei Protocolli di Comunicazione:** Ho dovuto decidere quali protocolli utilizzare per la comunicazione tra il client e il server, optando per HTTP/HTTPS per la semplicità di implementazione e la compatibilità con le librerie disponibili.
- **Gestione delle Richieste e delle Risposte:** La scrittura del codice per inviare richieste HTTP dal client e gestire le risposte del server ha richiesto una comprensione approfondita delle tecniche di serializzazione/deserializzazione dei dati e delle modalità di gestione degli errori.

### 3.3 Interferenze del Sensore Ultrasonico

Il sensore HC-SR04 ha mostrato problemi di interferenza a causa degli sbalzi di tensione dovuti all'utilizzo del modulo Wi-Fi dell'ESP32. Questi sbalzi di tensione causavano misurazioni instabili e inaccurate. Per risolvere questi problemi, ho utilizzato la libreria `NewPing.h`, che ha migliorato la stabilità e l'affidabilità delle misurazioni:

- **Utilizzo della Libreria NewPing:** La libreria `NewPing.h` è stata scelta per gestire le operazioni del sensore HC-SR04. Questa libreria offre funzionalità avanzate per il controllo del sensore ultrasonico, riducendo significativamente le interferenze e migliorando la precisione delle misurazioni.

### 3.4 Crittografia dei Messaggi con SSL

Implementare la crittografia dei messaggi utilizzando SSL è stata una sfida significativa. Non avendo esperienza precedente con SSL, ho dovuto apprendere come generare certificati, configurare il server per utilizzare HTTPS e assicurarmi che tutte le comunicazioni fossero crittografate. Le principali difficoltà incontrate sono state:

- **Generazione dei Certificati SSL:** Utilizzando OpenSSL, ho dovuto generare certificati autofirmati e configurare Apache per utilizzarli. Questo processo ha richiesto una buona comprensione dei concetti di crittografia e della configurazione di Apache.
- **Configurazione del Server:** Configurare Apache per utilizzare HTTPS ha comportato la modifica dei file di configurazione e la risoluzione di vari errori di configurazione che impedivano il corretto funzionamento del server.
- **Integrazione con l'ESP32:** Assicurarmi che l'ESP32 fosse in grado di comunicare correttamente con il server HTTPS ha richiesto la gestione dei certificati SSL sul dispositivo e la risoluzione di problemi legati alla compatibilità delle librerie.

### 3.5 Conclusioni sulle Sfide Incontrate

Le sfide affrontate durante lo sviluppo di questo progetto hanno fornito un'opportunità preziosa per apprendere e crescere come sviluppatore. Ogni difficoltà superata ha contribuito a migliorare le mie competenze tecniche e la mia capacità di risolvere problemi complessi. La configurazione del web server e del database, la progettazione dell'architettura client-server, la gestione delle interferenze del sensore e l'implementazione della crittografia SSL sono state tutte esperienze che hanno arricchito il mio percorso di apprendimento e mi hanno permesso di realizzare un sistema di monitoraggio che ampliato ad una rete non locale, potrei tranquillamente utilizzare nella mia casa.

