Sistema di Monitoraggio Presenze basato su Riconoscimento Biometrico

Luca Neve

Maggio 2025

1 Introduzione

L' idea di questo progetto nasce da un' esperienza che vivo in prima persona come studente universitario. Durante i tre anni trascorsi nei complessi residenziali offerti dalla regione per studenti con minori opportunità, ho osservato una situazione ricorrente: molti studenti ottengono l' assegnazione di una stanza, ma in seguito non vi risiedono effettivamente, lasciandola quindi formalmente occupata ma in pratica inutilizzata. Questa condizione porta a uno spreco di risorse destinate a supportare il percorso universitario di studenti che ne hanno bisogno e priva potenziali beneficiari di un' opportunità che potrebbero sfruttare appieno.

Per risolvere questa problematica, il progetto propone l' implementazione di un sistema di monitoraggio delle presenze che utilizzi un lettore di impronte digitali collegato a un database di registrazione. Grazie alla natura non replicabile dell' impronta digitale, gli studenti assegnatari dovranno registrare la propria presenza ogni `n` giorni attraverso l' impronta, confermando così l' effettiva occupazione della stanza. Questo sistema mira a garantire un uso più efficace delle risorse abitative, riservandole a chi ne beneficia attivamente.

2 Progettazione

2.1 Componenti Hardware

- **ESP32:** Questo microcontrollore rappresenterà il cuore del sistema e gestirà la lettura delle impronte digitali e la comunicazione con il database.

Figure 1: ESP32

Motivazione: l'ESP32 ha integrato le funzionalità Wi-Fi, questo rende più semplice e comoda la comunicazione con il database locale.

- **Sensore di Impronte Digitali:** Per l'integrazione con ESP32, i sensori **R307** e **GT511C3** rappresentano due delle migliori opzioni.

Figure 2: Sensore R307

Figure 3: Sensore GT511C3

Figure 4: Confronto visivo tra i sensori

Di seguito, la tabella 1 mette a confronto le caratteristiche principali di entrambi.

	GT511C3	**R307**
Capacità di Memoria	Fino a 200 impronte	Fino a 1000 impronte
Precisione (FAR)	0,001%	0,001%
Tempo di Identificazione	<1 secondo	<1 secondo
Interfaccia di Comunicazione	UART, USB	UART
Compatibilità	Arduino, Raspberry Pi	Arduino, ESP32, ESPHome
Costo	Medio-Alto	Basso-Medio
Facilità di Integrazione	Media, richiede più configurazione	Alta, facile da configurare
Affidabilità	Buona, meno resistente in ambienti intensivi	Molto buona, adatto a usi frequenti

Table 1: Confronto tra i sensori di impronte digitali GT511C3 e R307

Dalla comparazione risulta che il sensore **R307** è più adatto a questo progetto, grazie ai seguenti vantaggi:

1. **Capacità di memoria superiore**: Il R307 supporta fino a 1000 impronte digitali, ideale per una struttura che accoglie un gran numero di studenti, a differenza del GT511C3, che è limitato a 200 impronte e quindi meno adatto a gestire un ampio gruppo di utenti.
2. **Compatibilità nativa con ESP32**
3. **Affidabilità e robustezza per utilizzi intensivi**: Il sensore R307 è progettato per resistere all'uso frequente, garantendo performance costanti nel tempo, anche in contesti di accesso frequente come un collegio universitario.

- **Alimentazione:** Un alimentatore da 5V per il microcontrollore e per il sensore di impronte.
- **Computer per l'interfaccia di registrazione:** Un dispositivo accessibile dal personale all'ingresso della struttura per registrare i dati e l'impronta degli studenti.

2.2 Componenti Software

1. **Firmware ESP32:** Per l'implementazione è stato utilizzato `PlatformIO` come ambiente di sviluppo. Questo approccio ha permesso di scrivere il firmware in C++ e di integrare facilmente librerie come `AdafruitFingerprint` per la gestione del sensore.
2. **Interfaccia Grafica con Qt** Un'applicazione desktop, sviluppata con Qt, consente al personale di interagire con il sistema in modo semplice e intuitivo. La GUI funge da ponte tra l'operatore e l'ESP32, oltre a offrire funzionalità di consultazione e aggiornamento del database.

3. **Backend PHP + Database MySQL** Un server locale gestisce il database degli utenti e riceve le richieste inviate sia dall'ESP32 che dall'interfaccia Qt. Il backend è responsabile della persistenza e coerenza dei dati. **N.B.** Non viene memorizzata alcuna immagine o dato biometrico delle impronte nel database: il sensore R307 gestisce internamente l'archiviazione e il matching delle impronte.

2.3 Passaggi del Flusso

A. Inizializzazione

1. L'ESP32 si avvia, inizializza il sensore di impronte digitali R307 e si connette al WiFi.
2. Il programma entra in modalità di attesa comandi da PC (via porta seriale).

B. Interazione tramite GUI

1. L'interfaccia grafica realizzata in Qt permette al personale di:
   - Registrare un nuovo utente
   - Eseguire un accesso
   - Eliminare un singolo utente
   - Eliminare tutti gli utenti
2. Quando si preme un pulsante nella GUI, viene inviato un comando seriale all'ESP32 nel formato:

Operazione	**Comando inviato**
Registrazione utente	`REGISTRA;Nome;Cognome;Stanza\n`
Accesso	`ACCEDI\n`
Eliminazione utente specifico	`ELIMINA;ID\n`
Eliminazione di tutte le impronte	`PULISCI\n`
Navigazione (torna indietro)	`INDIETRO\n`

Table 2: Comandi inviati all'ESP32 tramite la GUI Qt

C. Risposta ESP32

1. L'ESP32 riceve il comando e lo interpreta, eseguendo:
   - Operazioni locali (registrazione, verifica, cancellazione).
   - Invio dei dati via HTTP al server PHP per aggiornare il database.
2. La risposta dell'ESP32 viene letta dalla GUI, che permette all'utente di visualizzarla tramite etichette o log nella finestra.

2.4 Collegamenti hardware

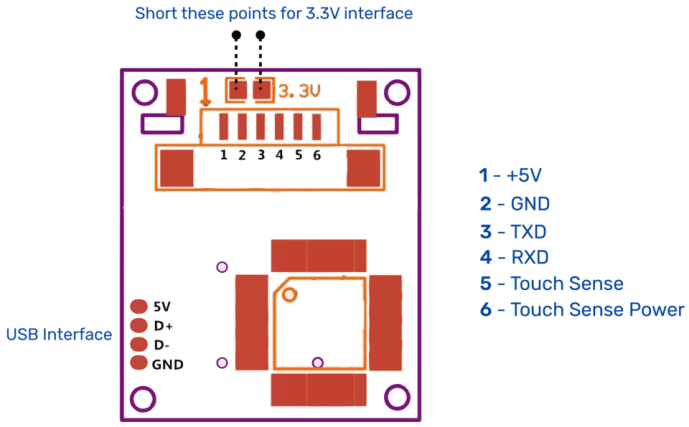

Fig. R307 Fingerprint Scanner Pinout

www.vishnumaiea.in

Pin Number	Name	Type	Description
1	+5V	IN	Positive Supply (DC 4.2V-6V)
2	GND	GND	Supply Ground
3	TXD	OUT	Data output (TTL)
4	RXD	IN	Data input (TTL)
5	Touch	OUT	Finger detection signal (max output current: 50mA)
6	3.3V	IN	Finger detection power (DC 3.3V-5V ~5uA)

Figure 5: R307 fingerprint scanner pinout

Pin del Sensore R307	Collegamento ESP32
VCC	Pin **5V**
GND	Pin **GND**
TXD	Pin **RX (GPIO16)**
RXD	Pin **TX (GPIO17)**
TOUCH	Pin **GPIO18**
3.3V	Pin **3.3V**

Table 3: Collegamenti tra il sensore R307 e l'ESP32

3 Implementazione

3.1 Configurazione dell'ESP32 e del Lettore di Impronte

Il firmware dell'ESP32 è stato sviluppato in C++ tramite l'ambiente PlatformIO. Il microcontrollore riceve comandi testuali inviati da un'interfaccia grafica realizzata in Qt.

Struttura delle funzioni

1. **Funzioni operative**:
   - **setup**: inizializza WiFi, seriale e sensore.
   - **loop**: attende e interpreta comandi provenienti dalla seriale.
   - **registraImpronta**: gestisce la registrazione di una nuova impronta nel lettore.
   - **eseguiAccesso**: verifica che l'impronta letta corrisponda ad un impronta già salvata all'interno del lettore e in caso positivo restituisce l'id dell'impronta.
   - **cancellaImpronta**, **cancellaTutte**: rimuovono impronte all'interno del lettore.
2. **Funzioni di comunicazione con il backend (PHP)** dopo aver eseguito l'operazione all'interno del lettore entrano in gioco queste funzioni, che eseguono l'operazione anche sul database.
   - **registraUtente**
   - **aggiornaUltimoIngresso**
   - **eliminaUtente**

   Il microcontrollore comunica con un backend PHP locale per registrare, aggiornare ed eliminare utenti nel database. Queste operazioni avvengono tramite richieste HTTP POST ai seguenti file:

Operazione	**File PHP**
Registrazione utente	`registra_utente.php`
Aggiornamento data ultimo accesso	`aggiorna_accesso.php`
Eliminazione utente	`elimina_utente.php`

Table 4: Connessione tra funzioni ESP32 e file PHP

L'ESP32 invia direttamente i dati al server locale, il quale aggiorna il database MySQL. Questo approccio garantisce che l'interazione con il database avvenga in modo centralizzato e sicuro, lasciando all'ESP32 solo il compito di raccogliere i dati biometrici e trasmetterli.

3.2 Interfaccia grafica sviluppata con Qt Creator

Per facilitare l'interazione con il sistema, è stata sviluppata un'interfaccia grafica utilizzando Qt Creator, un ambiente di sviluppo integrato per la creazione di GUI multipiattaforma in C++.

A differenza dell'ESP32, che si occupa della registrazione e dell'elaborazione delle impronte digitali, l'interfaccia Qt ha lo scopo principale di visualizzare e modificare i dati presenti nel database, fornendo una panoramica chiara e interattiva degli utenti registrati.

Essa comunica con:

Operazione	**File PHP**
Lettura utenti dal database	`get_users.php`
modifica di campi	`update_user.php`

Table 5: Connessione tra funzioni Qt e file PHP

La comunicazione avviene via HTTP in formato JSON, e i dati vengono visualizzati in una tabella interattiva che supporta la ricerca e la modifica diretta dei record.

3.2.1 Distribuzione del programma

Per facilitare l'utilizzo, la GUI viene fornita come **eseguibile precompilato per Windows**, accompagnato da tutte le libreria Qt necessarie (DLL, plugin, ecc..). In questo modo l'utente può utilizzare il programma senza bisogno di compilare nulla o di installare Qt Creator.

`N.B.` Non è necessario accedere al codice sorgente per utilizzare l'interfaccia. Tutto il necessario è incluso nella cartella "GUI_QT" presente nel progetto

3.3 Schema del flusso del funzionamento

AZIONE DELL'UTENTE SULLA GUI	COMANDO INVIATO ALL'ESP32	FUNZIONE CHIAMATA SU ESP32	AZIONE FINALE SUL DATABASE
REGISTRA UTENTE	REGISTRA;nome; cognome;stanza\n	registraImpronta() → registraUtente()	Chiamata HTTP a registra_utente.php
ACCEDI	ACCEDI\n	eseguiAccesso() → aggiornaUltimoIngresso()	Chiamata HTTP a aggiorna_accesso.php
ELIMINA UTENTE	ELIMINA;id\n	cancellaImpronte() → eliminaUtente()	Chiamata HTTP a elimina_utente.php
SVUOTA DATABASE	PULISCI\n	clearAll() → eliminaTuttiUtenti()	Chiamata HTTP a elimina_tutti_utenti.php

Figure 6: flusso di funzionamento

4 Procedura di installazione e avvio del sistema

Per poter utilizzare il sistema di monitoraggio presenze, è necessario eseguire una serie di operazioni preliminari per configurare l'ambiente software e la comunicazione tra i vari componenti. Di seguito viene descritta la procedura completa per installare e avviare il sistema in ambiente Windows.

4.1 Posizionamento del progetto

Copiare l'intera cartella del progetto, denominata `ProgettoBiometrico`, direttamente all'interno del disco locale `C:\`, in modo da ottenere il seguente percorso: `C:\ProgettoBiometrico\`. La struttura della cartella sarà già organizzata con tre sottodirectory principali:

- `ESP32`: contiene il progetto PlatformIO con il firmware per il microcontrollore.
- `GUI_QT`: contiene l'eseguibile Qt e tutte le librerie necessarie.
- `Server\htdocs\backend`:contiene i file PHP per la gestione del database.

4.2 installazione di XAMPP

1. Scaricare ed installare l'ambiente XAMPP dal sito ufficiale.
2. Avviare il pannello di controllo CAMPP e avviare i servizi Apache e MySQL

4.3 Configurazione del database

1. Accedere a phpMyAdmin tramite `http://localhost`.
2. Creare un nuovo database con nome: `gestione_impronte`.
3. All'interno di tale database, creare la tabella **studenti** con la seguente struttura:

Campo	**Tipo**
ID_Template	INT(11)
nome	VARCHAR(20)
cognome	VARCHAR(20)
stanza	VARCHAR(20)
DataUltimoAccesso	TIMESTAMP

Table 6: Campi database

N.B. Il campo DataUltimoAccesso potrà essere impostato con valore di default `CURRENT_TIMESTAMP` e aggiornato ad ogni accesso.

4.4 Configurazione dell'ESP32

- All'interno del codice del microcontrollore (file `main.cpp` del progetto PlatformIO), è necessario modificare le seguenti righe con le credenziali corrette del WiFi a cui si desidera collegarsi:

```
// Configurazione WiFi
const char* ssid = "NOME_RETE_WIFI";
const char* password = "PASSWORD_WIFI";
```

- Nel file `main.cpp` del progetto PlatformIO, oltre alle credenziali WiFi, è presente anche la seguente riga:

```
String serverBase = "http://192.168.171.245/backend/";
```

  Questa riga definisce l'indirizzo del server a cui l'ESP32 invia le richieste HTTP. È **fondamentale sostituire l'indirizzo IP 192.168.171.245 con l'indirizzo IP effettivo del computer su cui è in esecuzione il server Apache/XAMPP.** Può essere utile assegnare un IP statico a quel computer per evitare problemi di connettività nel tempo.

Successivamente, caricare il firmware sull'ESP32 utilizzando l'ambiente PlatformIO oppure un qualsiasi IDE compatibile.

4.5 Configurazione di Apache (XAMPP)

Per fare in modo che Apache punti alla cartella corretta del progetto, è necessario modificare il file di configurazione principale.

1. Aprire il file: `C:\xampp\apache\conf\httpd.conf`
2. Cercare le seguenti righe:

```
DocumentRoot "C:/xampp/htdocs"
<Directory "C:/xampp/htdocs">
```

3. Sostituirle con:

```
DocumentRoot "C:/ProgettoBiometrico/Server/htdocs"
<Directory "C:/ProgettoBiometrico/Server/htdocs">
```

4. Salvare il file e riavviare Apache dal pannello XAMPP.

4.6 Avvio del sistema

Una volta completati i passaggi sopra, è possibile avviare il sistema:

(a) Collegare l'ESP32 al computer tramite cavo USB.

(b) Eseguire il programma Qt (contenuto nella cartella GUI_QT), facendo doppio click sull'eseguibile.

(c) Assicurarsi che l'ESP32 sia sulla porta COM corretta e che sia stato riconosciuto.

(d) Utilizzare l'interfaccia grafica per registrare nuovi utenti, verificare presenze o modificare dati nel database.