CapybaraSafe: Sicurezza e Tracciatura degli Operatori nella NO-IP Zone attraverso un Innovativo Sistema IoT

Alessio Tudisco, Miriana Russo

March 5, 2024

(a) CapybaraDangerous

(b) CapybaraSafe

Figure 1: Descrizione generale delle immagini.

Abstract

Il progetto propone un sistema all'avanguardia per il monitoraggio e la sicurezza degli operatori all'interno di una NO-IP Zone, attraverso l'implementazione di una soluzione IoT innovativa. Utilizzando dispositivi personali avanzati, come le Smartband dotate di sensori quali giroscopio, accelerometro e ossimetro, il sistema offre una tracciatura precisa della posizione degli operatori, insieme a una dettagliata analisi dei tempi di percorrenza e di stallo nel tunnel.

L'attenzione è focalizzata sul benessere degli operatori, con la possibilità di monitorare in tempo reale i parametri vitali attraverso tecnologie indossabili. Inoltre, il sistema integra soluzioni basate su machine learning per il riconoscimento delle cadute, il monitoraggio dello stato di salute e la stima della distanza tra gli operatori e le ancore nel tunnel.

Alimentato a batteria e con la capacità di funzionare in un ambiente senza connessione diretta, il dispositivo indossabile offre una maggiore autonomia. All'interno del tunnel, vengono posizionati dispositivi alimentati direttamente per ottimizzare la copertura e garantire una risoluzione efficiente del problema.

I dati raccolti vengono archiviati in un database interno, accessibile tramite una rete IP esterna alla NO-IP Zone. L'amministrazione dispone di un servizio completo per monitorare la posizione e lo stato degli operatori, gestire i dispositivi e la loro configurazione, oltre a consultare lo schedule degli operatori. La soluzione è valutata sulla base dell'operatività, dei protocolli progettati, dell'estendibilità e della semplicità di installazione. La presenza di AI all'interno della soluzione contribuisce a migliorare la qualità e l'usabilità del servizio.

1 Introduzione

Nel contesto delle crescenti esigenze di sicurezza e monitoraggio degli operatori all'interno della NO-IP Zone, la progettazione di un sistema IoT all'avanguardia diventa imperativa. La soluzione proposta si basa sull'impiego di dispositivi personali avanzati, in particolare la Smartband, al fine di affrontare in modo efficace le sfide legate alla sicurezza e alla tracciatura degli operatori in ambienti critici.

Questa introduzione si propone di esaminare approfonditamente le proposte innovative relative all'integrazione del machine learning, focalizzandosi sulla capacità di riconoscere cadute, monitorare parametri vitali e stimare distanze critiche. In parallelo, esploreremo le caratteristiche peculiari del dispositivo personale, alimentato a batteria e dotato di sensori avanzati come giroscopio, accelerometro e ossimetro, che ne migliorano la capacità di monitoraggio.

Un aspetto cruciale della soluzione riguarda il protocollo di comunicazione generale, che consente la trasmissione efficiente dei dati all'interno e all'esterno del tunnel. L'obiettivo è affrontare la sfida di garantire la continuità della connettività, consentendo l'accesso e la gestione dei dati anche al di fuori della NO-IP Zone attraverso una rete IP.

Durante il percorso di esplorazione di questa soluzione innovativa, approfondiremo i dettagli di ciascun elemento chiave, dalla progettazione del dispositivo indossabile alle implementazioni specifiche del machine learning, con l'obiettivo di fornire una panoramica completa e approfondita di questa proposta di tracciatura e sicurezza avanzata per gli operatori.

L'intero progetto è stato realizzato in linguaggio C++ utilizzando il servizio Platform.io e diverse librerie, le quali saranno discusse nel dettaglio quando affronteremo ciascun argomento nella nostra relazione.

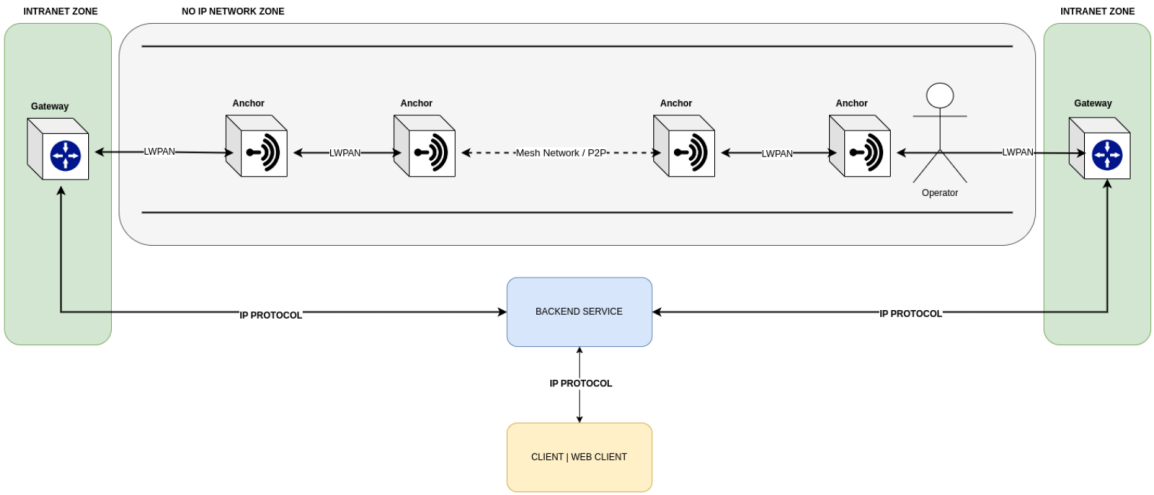

Figure 2: Idea Iniziale

2 Architettura del Sistema

La struttura architetturale del nostro sistema si basa su diversi componenti che interagiscono tra loro con diversi tipi di comunicazione, ciascuno finalizzato a garantire un monitoraggio efficiente e una rapida risposta a situazioni critiche. La nostra architettura include:

2.1 Comunicazione Periodica Utente-Gateway

Il dispositivo personale invia periodicamente dati all'utente attraverso un processo che coinvolge le ancore e il gateway. Questi dati includono l'identificatore (ID) dell'utente e il suo stato, con un segnale booleano che indica se l'utente ha segnalato un'emergenza. L'informazione viene trasmessa attraverso il gateway al server al fine di garantire aggiornamenti in tempo reale sul front-end. I dati vengono successivamente archiviati nel database per mantenere uno storico delle posizioni e delle condizioni degli utenti.

2.2 Comunicazione Avvisi da Gateway Esterni

In caso di avvisi improvvisi, ad esempio un comando di rientro alla base ("ALERT"), il front-end comunica con il server. Il server aggiorna la tabella delle segnalazioni nel database per registrare la situazione e invia l'avviso al gateway. Il gateway, utilizzando le ancore, trasmette l'avviso all'utente, consentendo una gestione efficace degli eventi imprevisti.

2.3 Gestione del Database

Il database è basato su MariaDB.

Il database contiene una tabella con i dati dei lavoratori, associando a ciascun lavoratore un ID del dispositivo al momento dell'ingresso nella NO-IP Zone. Il database

mantiene l'elenco degli utenti attualmente presenti nella NO-IP Zone, identificando gli esp attivi.

Memorizza l'ultima posizione nota di ciascun lavoratore, insieme alle informazioni sulle ancore, compresa la loro disposizione fisica (distanza e stato). Conserva gli avvisi generati dal front-end per informare gli utenti di eventuali situazioni di rilevanza.

2.4 Gestione del Backend

Il backend è in grado di gestire le mansioni assegnate a ciascun lavoratore.

Questa architettura integrata garantisce un sistema robusto e completo, consentendo un monitoraggio dettagliato degli operatori, la gestione efficiente degli avvisi e la registrazione accurata delle attività all'interno della NO-IP Zone.

- **Typescript**: Il linguaggio di programmazione TypeScript è impiegato per sviluppare il backend, offrendo tipizzazione statica e facilitando la manutenzione del codice.
- **MariaDB** come Datastore: MariaDB viene utilizzato come database per memorizzare in modo persistente i dati relativi alle mansioni e alle attività dei lavoratori.
- **Sequelize** come ORM: Sequelize funge da Object-Relational Mapping (ORM) e semplifica le operazioni di accesso e gestione dei dati nel database MariaDB.
- **Express** per API CRUD: Il framework web Express è impiegato per esporre API CRUD (Create, Read, Update, Delete), consentendo una comunicazione efficiente tra il frontend e il backend.
- **SocketIO** per Comunicazione Realtime: SocketIO è utilizzato per abilitare la comunicazione realtime fra il frontend e il backend, facilitando l'aggiornamento immediato delle informazioni.
- **MQTT** per Tecnologia di Messaggistica: La tecnologia MQTT viene adoperata per la messaggistica, sfruttando anche NanoMQ come tecnologia self-hosted MQTT-Broker per garantire una comunicazione stabile ed efficiente.

  In particolare, abbiamo scelto di adottare questa tipologia specifica di broker denominata NanoMQ, caratterizzata dalla sua leggerezza e facilità di portabilità. Questo broker consente il mirroring delle comunicazioni su broker appartenenti a reti differenti, ampliando così le possibilità di interconnessione.

  Tramite il software MQTTX correttamente installato abbiamo simulato l'invio di un messaggio tra un'ancora e il Personal Device. Il nostro progetto utilizza diveris topic tra cui:

  - broker/alert_system utilizzato per inviare gli Alert
  - broker/tracking_system utilizzato per inviare i messaggi.
  - broker/pairing_device verifichiamo che il nostro dispositivo sia effettivamente connesso. Se il dispositivo è connesso risponderà con un messaggio di "ack".

Figure 3: MQTT-Ack

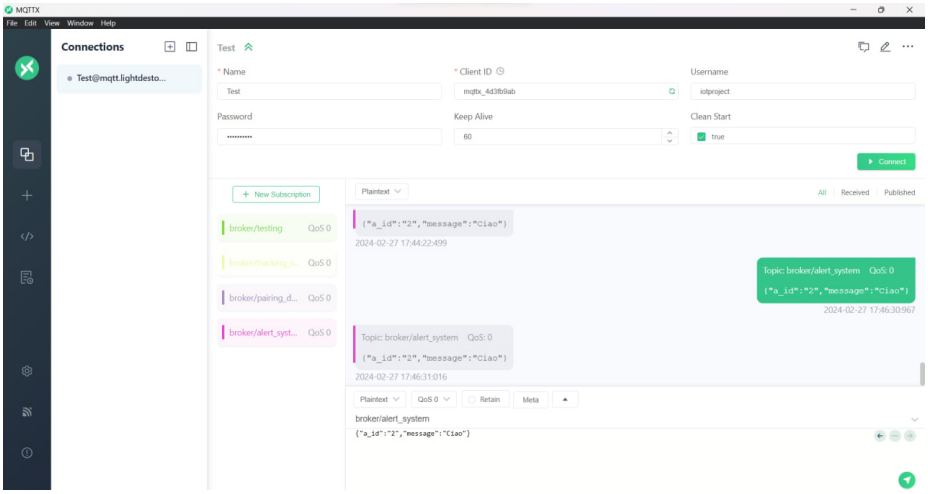

Figure 4: MQTTX

2.5 Gestione del Front-end

Tra le componenti usate nel Front-end abbiamo:

1. **HTML**: Linguaggio di markup utilizzato per strutturare il contenuto delle pagine web.

2. **CSS**: Linguaggio di stile che definisce l'aspetto visuale delle pagine web, garantendo una presentazione coerente e gradevole.

3. **JavaScript**: Linguaggio di programmazione utilizzato per implementare logica dinamica e interattività nel frontend, migliorando l'esperienza utente.

Il frontend è progettato con una pagina di login che facilita agli utenti l'accesso al nostro sistema e la visualizzazione delle informazioni. L'autenticazione avviene mediante l'utilizzo di token per garantire un elevato livello di sicurezza.

Dopo l'accesso, gli utenti si trovano di fronte a un'interfaccia che fornisce dettagli sulle attività attive in corso. Inoltre, è presente una schermata dedicata al profilo del lavoratore, che include informazioni quali la posizione attuale, l'attività in corso e le condizioni di salute (temperatura, saturazione e frequenza cardiaca).

La schermata offre anche un pannello laterale che consente di selezionare diverse sezioni del sistema, tra cui Real-Time Tracking, Attività, Workers, Alert System e Tracking Device.

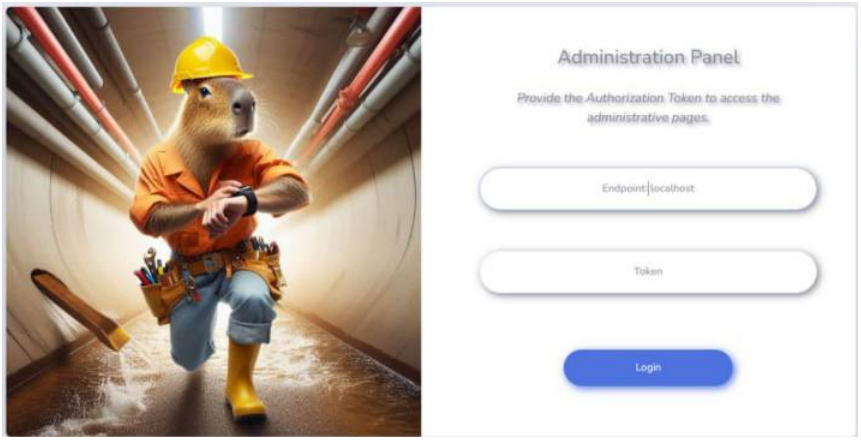

Figure 5: Login

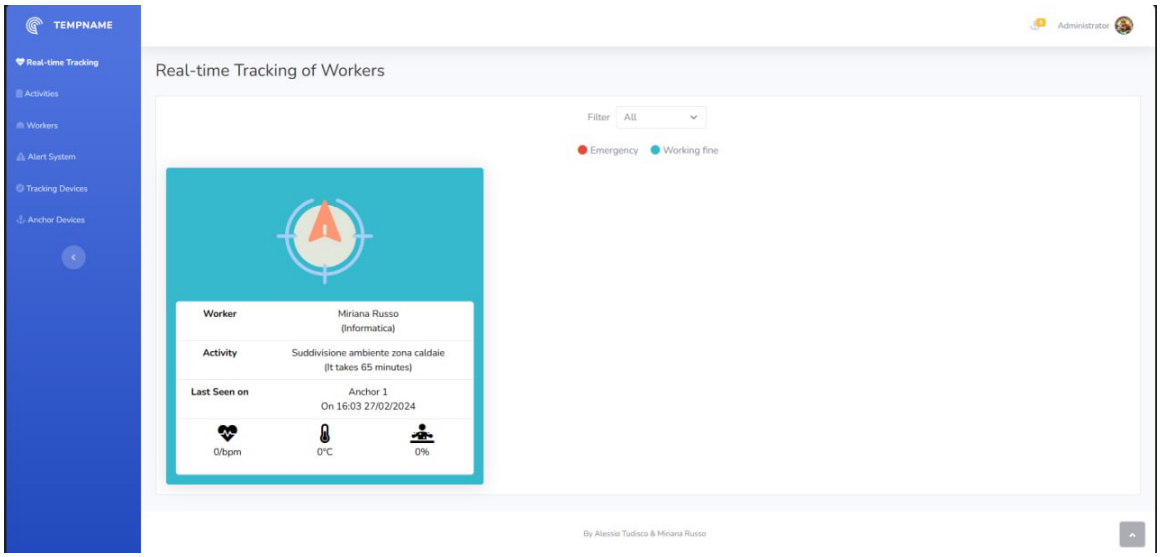

Figure 6: Real-time Tracking

Ciascuna sezione apre una pagina web dedicata con una tabella contenente informazioni dettagliate. Ad esempio, la sezione Tracking-Device mostra tutti i dispositivi, fornendo un pannello per filtrare le informazioni in base al loro utilizzo attuale.

Il frontend è progettato per fornire una visione completa del sistema, garantendo un'ottima usabilità per gli utenti incaricati di monitorare e verificare le informazioni. L'interfaccia intuitiva agevola la navigazione tra le diverse sezioni, permettendo agli utenti di accedere facilmente a dettagli specifici e garantendo un'esperienza efficiente nell'uso del sistema. La struttura chiara e ben organizzata del frontend in LaTeX riflette l'impegno nella progettazione di un'interfaccia utente user-friendly e altamente funzionale.

3 Database

Il sistema di gestione del database è fondamentale per monitorare e coordinare le attività degli operatori all'interno della NO-IP Zone. Al fine di mantenere un registro dettagliato

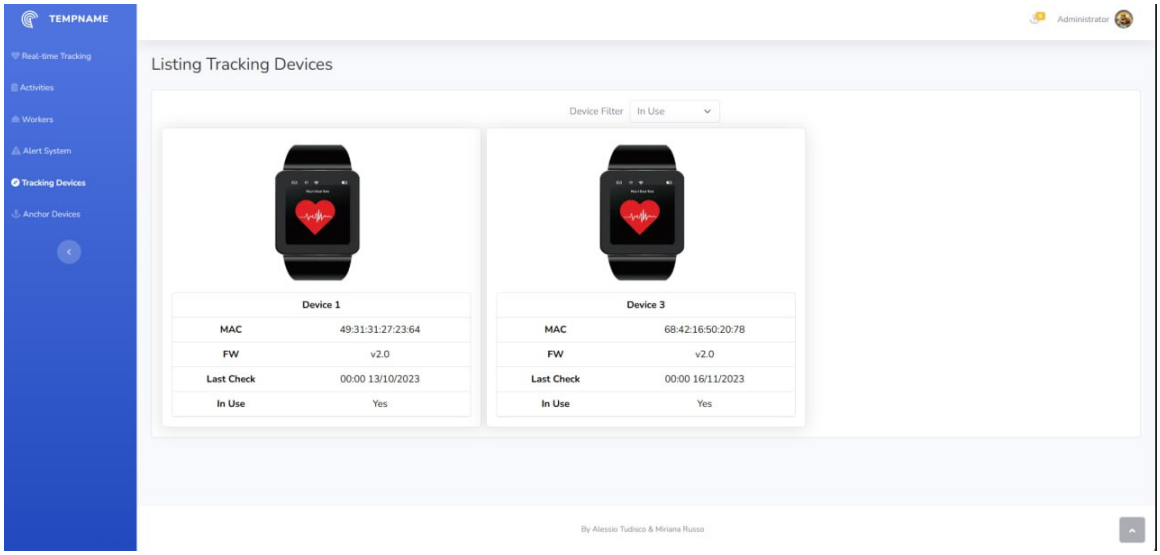

Figure 7: Tracking Device

dei lavoratori e delle attività, sono definite le seguenti entità:

1. **Lavoratori:**
   - Attributi: ID, Nome, Cognome, Professione.
2. **Attività:**
   - Attributi: ID, Descrizione, Durata, Data, Stato.
   - Le attività possono essere in corso, da ultimare, programmante o concluse.
   - Ogni attività ha una durata massima di 120 minuti, al termine della quale i lavoratori devono aver completato l'incarico o essere richiamati.

   Quest'ultimo punto è stato preso sotto considerazione in conformità con i contratti nazionali collettivi, i quali considerano la salute del lavoratore e stabiliscono che chi trascorre almeno 20 ore medie settimanali davanti a uno schermo del computer ha il diritto di interrompersi per 15 minuti ogni due ore.
3. **Incarichi:**
   - Attributi: ID, ID_lavoratore, ID_attività.
   - Le attività possono essere assegnate a uno o più lavoratori.
4. **Tracking Device:**
   - Attributi: ID, MAC, Ultima manutenzione, Versione firmware.
   - Ogni lavoratore è dotato di un dispositivo per il tracciamento all'interno della NO-IP Zone.
5. **Ancore:**
   - Attributi: ID, MacAddress, Stato.

- Le ancore sono dispositivi utilizzati per valutare la posizione dei lavoratori, basandosi su una distanza specifica e un punto di riferimento.

6. **Remote Tracking:**

   - Attributi: ID, ID_progressivo, ID_ancora, ID_lavoratore, ID_dispositivo, ID_attività, emergenza (bool), battiti cardiaci, saturazione, temperatura, forza_segnale.
   - Registra la posizione e lo stato di salute del lavoratore inviando report regolari durante l'attività. Il campo emergenza indica se il lavoratore ha bisogno di assistenza.

Questo sistema di gestione del database fornisce una struttura solida e organizzata per supportare l'efficace monitoraggio e coordinamento delle attività all'interno della NO-IP Zone, garantendo al contempo la sicurezza e il benessere degli operatori.

Di seguito una rappresentazione grafica del nostro Database :

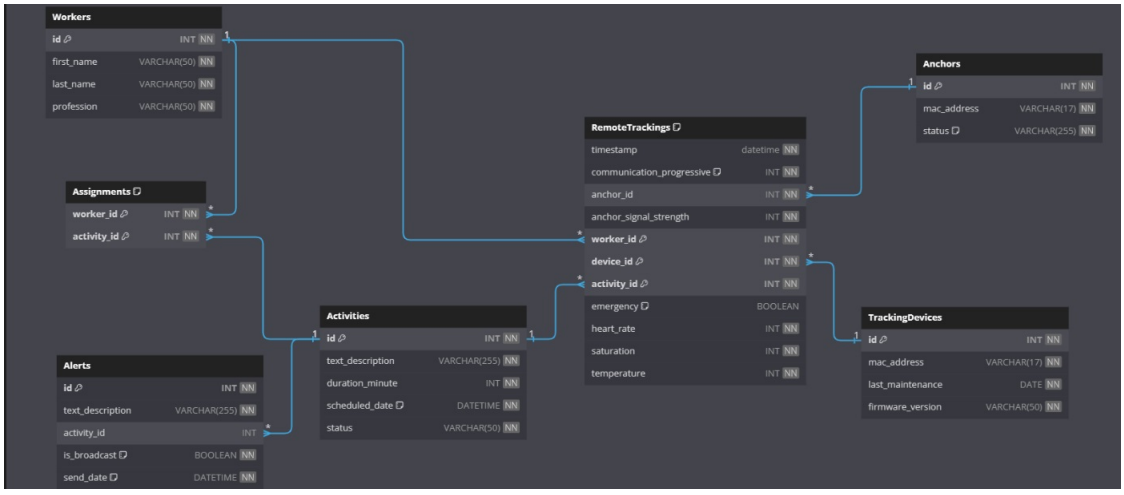

Figure 8: Database

4 Rete Neurale

All'interno del nostro progetto, abbiamo affrontato la sfida di sviluppare due reti neurali, denominate Heart Rate e Falling Edge, per la gestione della sicurezza personale. Una scelta strategica è stata quella di separare le due reti neurali, considerando le limitazioni di memoria del dispositivo hardware che stiamo utilizzando, in particolare un microcontrollore ESP32.

La rete Falling Edge è implementata all'interno del nostro dispositivo personale. Questa rete neurale è responsabile per la rilevazione delle cadute, indicando situazioni di svenimento o incidenti potenzialmente pericolosi per il lavoratore. Data la limitata capacità di memoria del dispositivo ESP32, abbiamo preso la decisione di eseguire il modello di inferenza della Falling Edge all'interno del dispositivo personale. Una volta che la rete ha effettuato l'inferenza, i risultati vengono trasmessi alle ancore, insieme ai parametri di temperatura, saturazione e battiti cardiaci.

Le ancore ricevono le informazioni dalla rete Falling Edge e al loro interno contengono un'altra rete neurale chiamata Heart Rate. Quest'ultima rete è dedicata alla monitorazione dei parametri cardiaci dell'utente, compiendo ulteriori passi per garantire la sicurezza.

Le ancore sono progettate per comprimere l'informazione ottenuta dalle reti neurali in un singolo bit. Questo bit diventa uno dei parametri dell'operatore logico OR, insieme al valore fornito dalla rete neurale del dispositivo personale. In caso uno dei due sia vero, si attiva la procedura di emergenza, segnalando un possibile pericolo e garantendo una risposta rapida.

La separazione delle reti è stata una scelta strategica dettata dalle limitazioni di memoria del nostro dispositivo ESP32, che non consente il processamento simultaneo di entrambe le reti neurali all'interno della propria memoria.

4.1 Heart Rate

La Rete Neurale Heart Rate è stata sviluppata con l'obiettivo di monitorare e prevenire potenziali attacchi cardiaci. L'approccio alla realizzazione di questa rete neurale si è basato sull'utilizzo di un datasheet che considera diversi parametri fisiologici fondamentali, tra cui temperatura corporea, saturazione dell'ossigeno e frequenza cardiaca. Questi parametri forniscono informazioni cruciali sulla salute cardiovascolare di un individuo.

Parametri Chiave

1. **Temperatura Corporea:** La temperatura corporea è un indicatore importante dello stato fisiologico di un individuo. Variazioni significative possono essere indicative di problemi cardiaci.

2. **Saturazione dell'Ossigeno:** La saturazione dell'ossigeno nel sangue è un parametro critico che riflette il livello di ossigeno trasportato nel corpo. Bassi livelli possono essere indicativi di problemi cardiovascolari.

3. **Frequenza Cardiaca:** La frequenza cardiaca è un elemento centrale nell'analisi della salute cardiaca. Variazioni anomale possono suggerire un potenziale rischio di attacco cardiaco.

Output della Rete Neurale

La rete neurale è stata progettata per produrre solamente due tipi di output: "malato" o "sano". Questa semplicità è intenzionale per facilitare l'interpretazione dei risultati e fornire una risposta rapida all'utente.

L'obiettivo principale della Rete Neurale Heart Rate è prevenire gli attacchi cardiaci. L'allenamento della rete coinvolge l'utilizzo di dataset ampi e diversificati che contengono casi di individui sani e malati, etichettati in base ai parametri sopra menzionati. Durante il processo di apprendimento, la rete neurale acquisisce la capacità di rilevare pattern e correlazioni che indicano la predisposizione a un attacco cardiaco.

4.2 Falling Edge

La Rete Neurale Falling Edge è stata progettata per la rilevazione delle cadute, indicando svenimenti o cadute che potrebbero mettere in serio rischio il lavoratore, come ad esempio incidenti sul luogo di lavoro. Questa rete neurale è configurata per inviare immediatamente un messaggio di emergenza quando viene rilevato un evento di caduta.

Parametri Chiave

1. **Velocità Lineare Massima:** La massima velocità lineare registrata fornisce informazioni sul movimento rapido dell'individuo, utile per identificare cadute o incidenti.

2. **Skewness dell'Accelerometro e del Giroscopio:** La skewness misura l'asimmetria della distribuzione dei dati. Nei dati provenienti da accelerometro e giroscopio, la skewness può indicare comportamenti non normali che potrebbero essere associati a una caduta.

3. **Velocità Lineare Precedente:** La velocità lineare precedente fornisce informazioni sul cambiamento di movimento e può essere cruciale per determinare se si è verificato un improvviso cambiamento di stato.

4. **Valore del Giroscopio Precedente:** Il valore precedente del giroscopio offre indicazioni sulla rotazione e sulle variazioni angolari. Cambiamenti improvvisi possono essere correlati a cadute o incidenti.

Output della Rete Neurale

L'output della Rete Neurale Falling Edge è binario, indicando se l'evento rilevato è una caduta ('FALL') o se l'individuo è in uno stato normale ('NOT_FALL').

Obiettivo Principale

Il principale obiettivo della Rete Neurale Falling Edge è rilevare e rispondere prontamente a situazioni di caduta o svenimento. Attraverso l'allenamento con dataset diversificati che includono dati relativi a incidenti di lavoro e situazioni di caduta, la rete neurale apprende a identificare pattern associati a tali eventi.

4.3 Implementazione

L'implementazione della rete neurale utilizza la libreria Eloquent e TensorFlow per Esp32, e il modulo TENSORFLOW_RUNNER agisce come interfaccia per i modelli di

machine learning TensorFlow Lite, focalizzandosi sulla rilevazione del battito cardiaco e sulla segnalazione di cadute.

All'interno del codice, la classe Eloquent::TinyML::TfLite gestisce il modello, con parametri template derivati dalle costanti definite in HEARTH_FAILTURE_MODEL.

L'inizializzazione del modello avviene attraverso la funzione initModel, che tenta il caricamento e registra eventuali errori tramite il modulo SERIAL_LOGGER.

La funzione requestEvaluation accetta dati simulati dal sensore cardiaco e utilizza il modello per effettuare previsioni, memorizzando i risultati in un array. La funzione heartDetected restituisce true se il modello predice la presenza di battito cardiaco o di caduta. Complessivamente, il codice dimostra una chiara modularità nella gestione del modello TensorFlow Lite, offrendo un'interfaccia ben definita per le operazioni di inizializzazione e valutazione.

4.4 Strumenti e Protocolli utilizzati

Nel corso dello sviluppo del nostro progetto, abbiamo sfruttato diversi strumenti e protocolli per garantire una comunicazione efficiente tra i vari dispositivi, compresi le Ancore, il Gateway e il Personal Device.

Tra gli strumenti utilizzati, ci siamo avvalsi del linguaggio di programmazione C++ e della piattaforma di sviluppo PlatformIO, che ci hanno fornito un ambiente di programmazione robusto e flessibile.

Per quanto riguarda i protocolli, abbiamo implementato una serie di soluzioni per facilitare la comunicazione:

4.5 Mesh con Tecnologia LoRa

Abbiamo adottato il protocollo Mesh basato su LoRa per orchestrare la comunicazione tra le Ancore e il Gateway, una decisione che ha dimostrato di essere cruciale per ottimizzare la connessione all'interno del nostro progetto. La topologia di rete mesh si è rivelata particolarmente vantaggiosa, garantendo una flessibilità maggiore nella gestione delle comunicazioni. La scelta del protocollo Mesh è stata motivata dalla necessità di superare le sfide legate alla complessità della struttura del nostro ambiente. La configurazione mesh ha consentito una distribuzione uniforme delle Ancore, creando una rete resiliente in grado di adattarsi a eventuali ostacoli presenti nel percorso delle comunicazioni. La tecnologia LoRa, utilizzata come base per il protocollo Mesh, ha fornito un vantaggio significativo. Grazie alle sue caratteristiche di trasmissione a lungo raggio e basso consumo energetico, si è rivelata ideale per l'ambiente specifico indoor del nostro progetto . Questa tecnologia ha consentito di estendere la copertura delle comunicazioni, riducendo al contempo il consumo energetico dei dispositivi coinvolti.

Implementazione Vediamo alcuni punti chiave della sua struttura e funzionalità:

Il codice inizia con la dichiarazione e l'inizializzazione di variabili e oggetti necessari per gestire la comunicazione mesh. Viene utilizzato un oggetto RH_RF95 per la comunicazione radio basata su LoRa, mentre l'oggetto RHMesh funge da gestore della rete mesh. Viene poi richiamata la funzione init() che si occupa dell'inizializzazione di questi componenti.

Successivamente viene configurato il modulo RF95 con parametri come la potenza di trasmissione. Queste configurazioni sono fondamentali per ottimizzare la trasmissione dei dati all'interno dell'ambiente di lavoro.

Come in ogni implementazione interna ai micocontrollori abbiamo implementato una funzione mesh_loop() la quale rappresenta il cuore del sistema di comunicazione. Controlla periodicamente se è giunto il momento di inviare un nuovo messaggio, popolando e inviando i dati presenti in una coda implementata nel nostro sistema modulare e definita come "GLOBALS". Inoltre, gestisce la ricezione dei messaggi provenienti dalla rete mesh e intraprende azioni specifiche in base al tipo di messaggio ricevuto.

Il codice gestisce due tipi di messaggi principali, identificati come "A" (ALERT) e "T" (tracciamento).

In caso di Alert, il messaggio viene inviato sia tramite modulo Bluetooth e in Broadcast. Il gateway comunica con le nostre ancore e che trasmettono il messaggio agli altri nodi della rete mesh fino a comunicare tramite il BLE con il nostro personal Personal Device che farà scattare un allarme indicando al lavoratore di ritornare immediata-

mente. Nel nostro caso scatterà un allarme sonoro e sullo schermo del dispositivo apparirà il messaggio inviato tramite il broker MQTT.

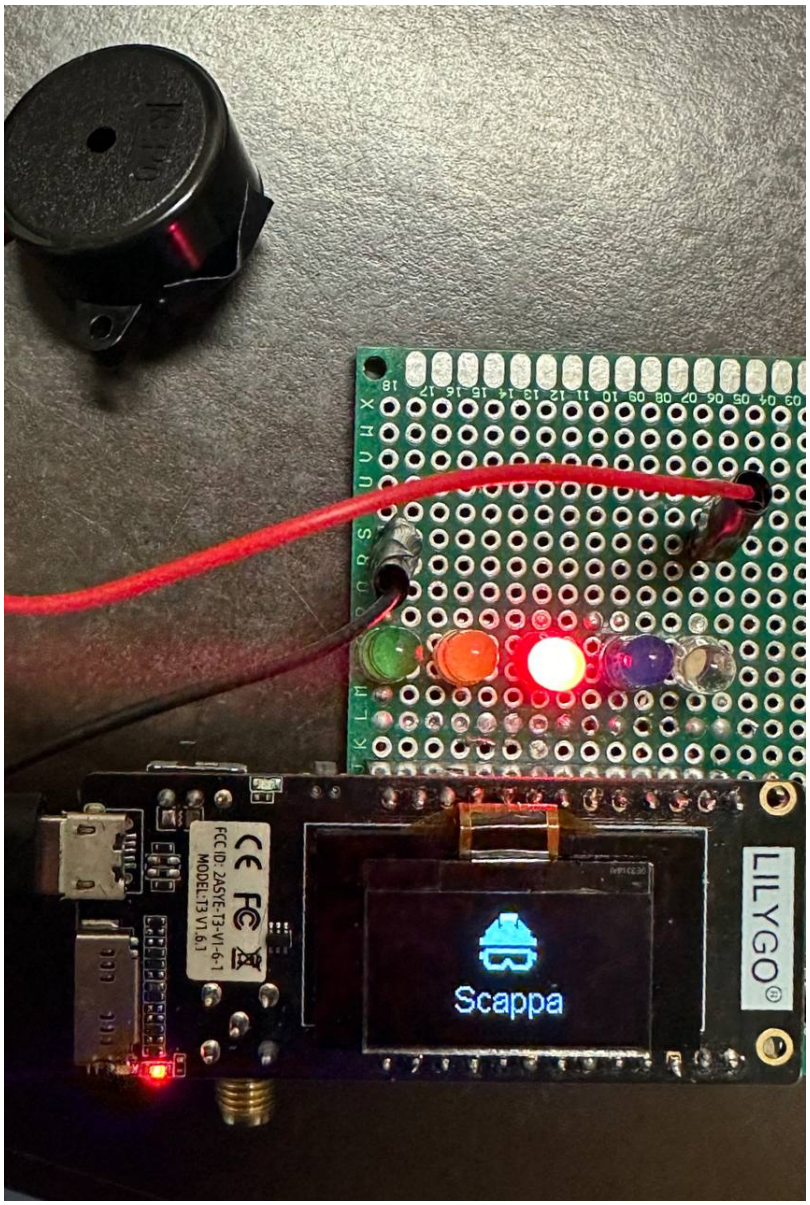

Figure 9: Messaggio Alert

Nel caso di messaggi di tracciamento, se il nodo corrente è un gateway, il messaggio viene inoltrato a un server MQTT tramite WiFi.

La generazione del payload segue un protocollo da noi definito, e per questo motivo sono implementate due funzioni distinte.

La prima funzione elabora i dati provenienti dal personal device e crea una stringa formattata. Successivamente, attraverso un'altra funzione, questa stringa viene ulteriormente processata in modo da essere inoltrata già elaborata. Questo processo consente al messaggio di percorrere la rete mesh, raggiungere il gateway e, infine, essere trasmesso al backend in una forma già pronta per l'elaborazione.

4.6 Bluethoot Low Energy or BLE

Abbiamo implementato il protocollo Bluetooth Low Energy (BLE) per gestire la comunicazione tra le Ancore e il Personal Device all'interno del nostro sistema. Il nostro modulo BLE si articola in un BLE Server e un BLE Client, consentendoci di stabilire una comunicazione bidirezionale efficiente con il dispositivo personale.

Il ruolo del BLE Server è fondamentale nella nostra implementazione. Esso si occupa di gestire la disponibilità di servizi e caratteristiche, consentendo al Personal Device di accedere a dati specifici e interagire con le Ancore. D'altra parte, il BLE Client agisce sul Personal Device, permettendo di inizializzare la connessione, inviare richieste e ricevere i dati forniti dal BLE Server.

Questa architettura client-server offre un meccanismo robusto e scalabile per la comunicazione Bluetooth all'interno del nostro ecosistema. La flessibilità di BLE consente una gestione efficiente dell'energia, essenziale per garantire un funzionamento ottimale dei dispositivi a lungo termine.

L'implementazione del protocollo BLE arricchisce notevolmente la nostra piattaforma, consentendo una connettività senza fili affidabile e a basso consumo energetico tra le Ancore e il Personal Device. Questo approccio avanzato contribuisce alla versatilità e alla praticità del nostro sistema di comunicazione, garantendo una user experience fluida e efficiente.

Implementazione Per la comunicazione BLE tra le Ancore e il Personal Device, abbiamo sfruttato la libreria NimBLE. Questa libreria offre un'implementazione efficiente del protocollo BLE e ci ha permesso di creare un ambiente di comunicazione bidirezionale tra i dispositivi.

La configurazione del servizio BLE, la scansione dei dispositivi circostanti e la gestione delle connessioni sono gestite attraverso questa libreria. Inoltre, il codice si occupa di notificare eventuali allerte ai dispositivi BLE trovati, sfruttando le opportunità di comunicazione offerte dalla tecnologia BLE.

4.7 Persistenza e Serializzazione

Per gestire la persistenza dei dati, come l'ID del dispositivo, abbiamo utilizzato la libreria Preferences di ESP32. Questa libreria ci ha permesso di memorizzare in modo permanente le informazioni chiave necessarie per il funzionamento del sistema, come l'ID delle Ancore.

4.8 Librerie Aggiuntive

Al fine di implementare funzionalità specifiche, abbiamo utilizzato altre librerie di supporto. Ad esempio, abbiamo impiegato la libreria JSON Arduino per la serializzazione e la deserializzazione dei dati JSON, fondamentali per la comunicazione e l'interpretazione dei messaggi.

Abbiamo optato per l'impiego della libreria RegEX per filtrare i dispositivi. Ciò è dovuto al fatto che, durante la scansione Bluetooth del nostro ESP, vengono rilevati numerosi dispositivi. Tuttavia, grazie a questa libreria, siamo in grado di effettuare un filtraggio che ci consente di comunicare esclusivamente con i nostri dispositivi.

Complessivamente, l'utilizzo di queste librerie ha semplificato lo sviluppo del nostro sistema, offrendo moduli pronti all'uso per la gestione della comunicazione e la persistenza dei dati.

4.9 Struttura del codice

La decisione di adottare una struttura a moduli anziché una struttura modulare è stata guidata dalla necessità di organizzare il codice in modo gerarchico, facilitando la comprensione, la manutenzione e la collaborazione tra i membri del team di sviluppo. La distinzione tra una struttura a moduli e una modulare può essere argomentata considerando diversi aspetti:

1. **Comprensibilità del Codice**: Una struttura a moduli consente di organizzare il codice in sezioni logiche, ognuna dedicata a un aspetto specifico del sistema. Ciò rende più agevole individuare le funzionalità correlate e capire il funzionamento generale di ciascun modulo. Ad esempio, avendo moduli separati per Bluetooth, Mesh, Persistenza, Sensor Fake, WiFi e MQTT, i membri del team possono concentrarsi su specifiche aree di competenza senza dover esplorare l'intera struttura modulare.

2. **Manutenzione Semplificata:** La suddivisione in moduli facilita la manutenzione, poiché ogni modulo può essere trattato come un'entità autonoma. Eventuali modifiche o miglioramenti possono essere apportati in modo isolato, riducendo il rischio di impatti indesiderati su altre parti del sistema. Ciò semplifica anche la gestione delle versioni, in quanto le modifiche possono essere controllate più efficacemente.

3. **Flessibilità e Aggiornamenti Incrementali**: La struttura a moduli consente una maggiore flessibilità nell'aggiornare o estendere il sistema. I nuovi moduli possono essere aggiunti o esistenti possono essere migliorati senza dover riscrivere l'intera struttura. Questo approccio favorisce gli aggiornamenti incrementali e consente l'integrazione agevole di nuove funzionalità.

5 Inizializzazione e Configurazione dei Dispositivi

La soluzione proposta si caratterizza per la sua semplicità di installazione, sfruttando un'applicazione Bluetooth di terze parti facilmente scaricabile su dispositivi mobili. Dopo l'attivazione, i dispositivi personali, il gateway e le ancore diventano visibili attraverso la scansione eseguita dall'applicazione. Questo processo agevola la connessione ai dispositivi, consentendo inoltre di modificare l'ID mediante un'operazione di scrittura.

Attraverso la nostra applicazione è possibile connettersi al dispositivo e da una semplice interfaccia scrivere o leggere le sue caratteristiche.

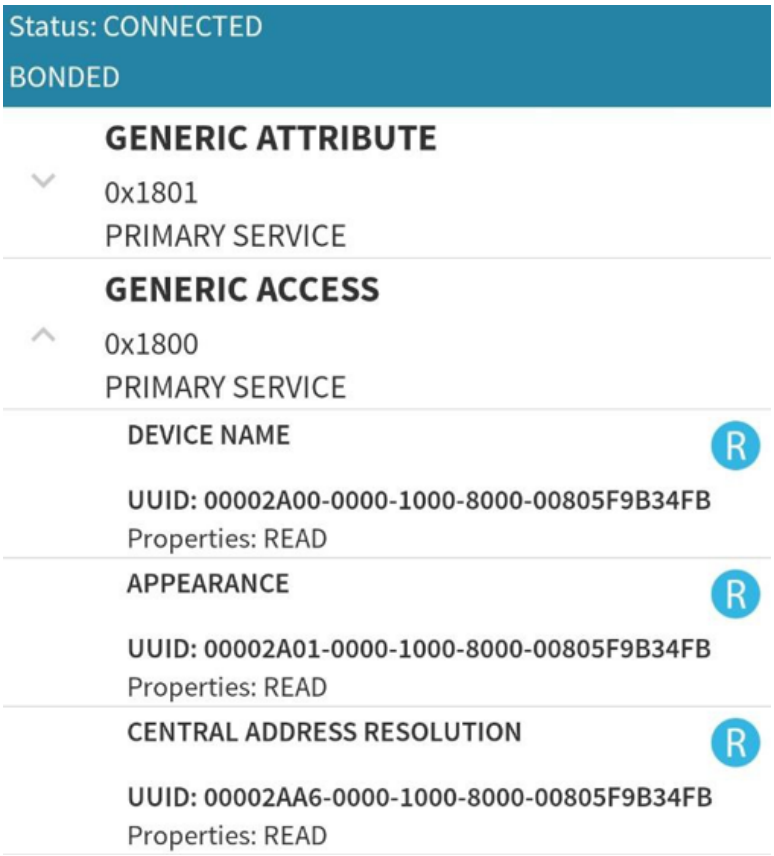

Figure 10: Bluetooth Application

Per garantire uniformità, abbiamo standardizzato gli ID del gateway e per convenzione i loro valori saranno 1 e 254. Tale scelta comporta che, in caso di sostituzione di un'ancora difettosa, l'addetto alla manutenzione può agevolmente riconfigurare la nuova ancora, assegnandole lo stesso ID della precedente. Collocando quindi la nuova ancora nella stessa posizione all'interno del tunnel, si affronta in modo rapido e lineare la problematica connessa alla sostituzione.

Questa metodologia garantisce un'agevole gestione delle ancore e semplifica le operazioni di manutenzione, offrendo una soluzione pratica e immediata per eventuali sostituzioni o aggiornamenti degli elementi del sistema. La flessibilità della procedura consente di mantenere la continuità operativa del sistema, riducendo i tempi di fermo e ottimizzando l'efficienza del processo di manutenzione.

In aggiunta, il Personal Device è dotato di due pulsanti che consentono di eseguire un hard reset del dispositivo.

6 Test e Problematiche

Il nostro percorso non è stato privo di sfide, in quanto ci siamo imbattuti in alcune problematiche, come sottolineato da questa citazione:

> "La vita è ciò che succede mentre sei occupato a fare altri progetti." - Allen Sanders
>
> Questa citazione suggerisce che, nonostante i nostri piani e desideri, la vita spesso ci presenta sfide impreviste. È un invito a essere flessibili, adattarsi alle circostanze e imparare dalle esperienze, anche quando le cose non vanno come previsto ed è esattamente quello che è successo a noi.

Abbiamo dovuto confrontarci con aspetti che, inizialmente, ci erano sconosciuti, ma che hanno inevitabilmente messo alla prova la solidità dei nostri progetti.

Problemi riscontrati:

- Abbiamo tentato di implementare due reti neurali all'interno dell'ESP, ma il nostro sforzo è risultato infruttuoso nonostante un attento tuning per individuare la dimensione minima del tensore. Nel contesto del personal device, attualmente, è presente una rete neurale, ampiamente discussa nei paragrafi precedenti, focalizzata sul Falling Edge. Abbiamo altresì cercato di integrare nello stesso dispositivo un'altra rete neurale dedicata al rilevamento di temperatura, battiti cardiaci, e altri parametri, ma ciò è stato impedito dalla limitata capacità di memoria dell'ESP, portando al fallimento del test.

- Come precedentemente menzionato nel paragrafo intitolato *"Strumenti Utilizzati"*, abbiamo adottato il protocollo BLE per la comunicazione tra le Ancore e il nostro Personal Device. Per la comunicazione tra le Ancore, invece, abbiamo optato per il protocollo LoRa, attraverso il quale è stata implementata la Mesh. Inizialmente, la nostra idea prevedeva l'uso del protocollo BLE-MESH. Tuttavia, ciò non è stato possibile a causa della limitata documentazione disponibile per il nostro microcontrollore ESP32. Inoltre, l'implementazione di BLE-MESH richiedeva l'utilizzo di una configurazione device per il provisioning, rendendo l'intero sistema ancora più complesso da realizzare.

- Non è possibile attivare contemporaneamente il modulo Stack e il Bluetooth sul nostro ESP32. Questa limitazione deriva da restrizioni hardware o conflitti di risorse che richiedono una gestione o un bilanciamento attento delle funzionalità del dispositivo.

- La presenza di un codice monolitico, ovvero un'unica entità di software senza una struttura modulare, costituisce una sfida nell'ambito della realizzazione del progetto. Tale approccio può rendere più complessa la manutenzione, la risoluzione di bug e l'aggiunta di nuove funzionalità. Si potrebbe valutare una suddivisione in moduli o una struttura più scalabile per migliorare la gestibilità del codice.

- Memory Leak il che significa che stiamo riscontrando una perdita progressiva di memoria nel corso del tempo, portando a potenziali malfunzionamenti o arresti anomali del sistema. Questo ha richiesto un'analisi approfondita del codice per individuare e correggere le aree responsabili delle perdite di memoria.

- Problematiche legate alla trasmissione LoRa, con un limite di 50 byte per ogni trasmissione. La sfida principale in questa situazione è rappresentata dalla necessità di gestire efficacemente la dimensione dei dati trasmessi, poiché superare il limite prestabilito potrebbe comportare la perdita di informazioni o il malfunzionamento del sistema. È cruciale esaminare attentamente il flusso di dati e considerare strategie come la compressione dei dati o la suddivisione in pacchetti più piccoli per affrontare questa limitazione e garantire una trasmissione affidabile attraverso la rete LoRa.

7 Architettura Finale

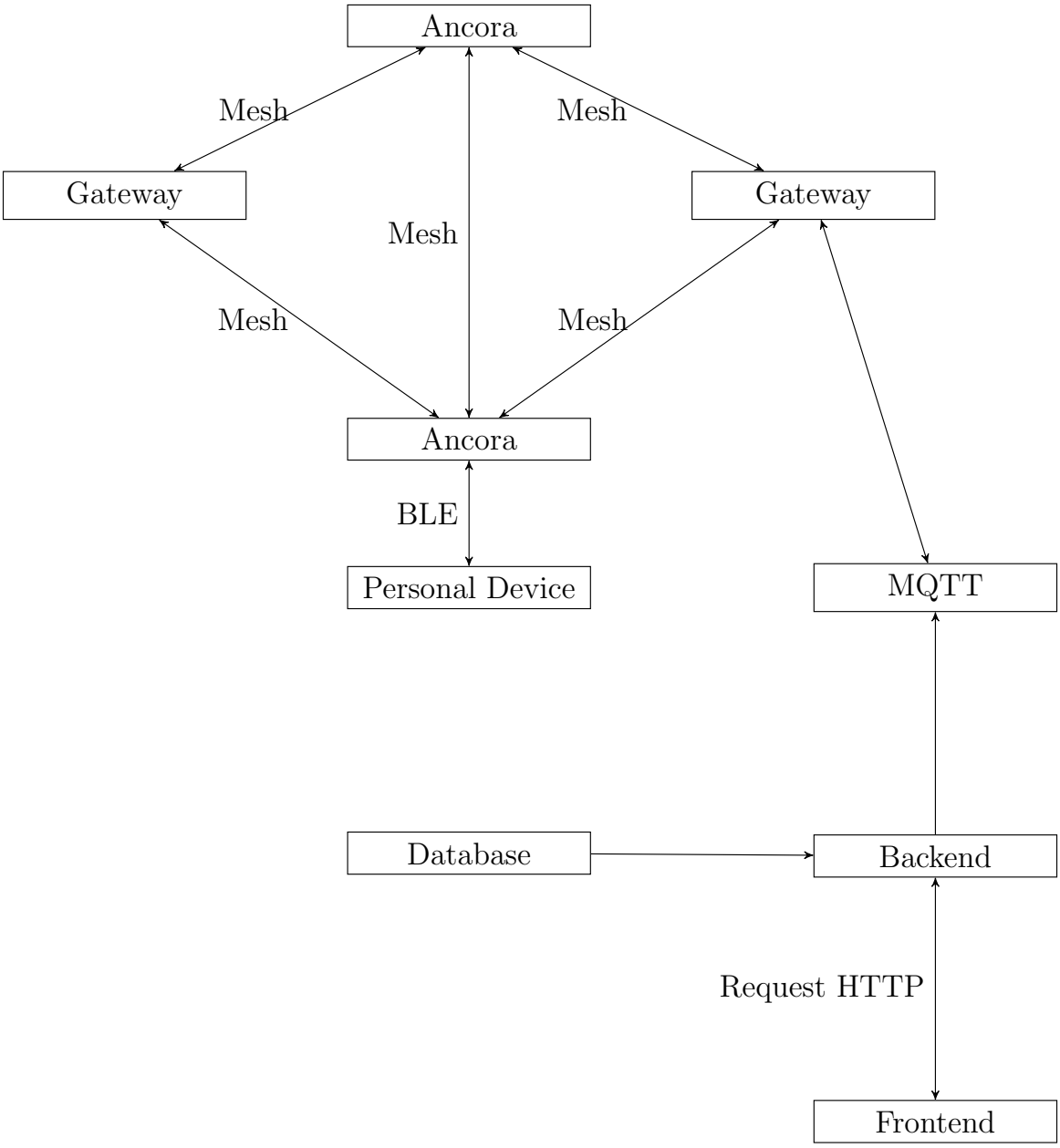

Figure 11: Architettura del Sistema

8 Sviluppi Futuri

Il nostro progetto rappresenta solo un primo passo in una direzione che può essere ulteriormente sviluppata e migliorata nel tempo. Purtroppo, durante la fase attuale, non siamo riusciti ad affrontare alcuni aspetti che avremmo voluto esplorare a causa delle tempistiche ridotte.

Tra gli sviluppi futuri, consideriamo:

1. Sistema di Allarme per Emergenze: Intendiamo implementare un sistema di allarme che si attivi nel caso in cui la durata massima di permanenza di un lavoratore venga superata senza che questi esca dal luogo di NO-IP zone. Questo meccanismo è pensato per segnalare eventuali situazioni di emergenza o pericolo, garantendo una risposta tempestiva alle necessità del lavoratore.

2. Ottimizzazione dell'Interfaccia Utente: Miglioreremo l'esperienza utente attraverso un'interfaccia più intuitiva e dettagliata. L'obiettivo è fornire informazioni chiare e facilmente comprensibili riguardo alle attività in corso e magari anche alle condizioni ambientali.

3. Integrazione con Altre Tecnologie: Esploreremo l'integrazione del nostro sistema con altre tecnologie emergenti, potenzialmente sfruttando l'intelligenza artificiale per migliorare la precisione delle previsioni o implementare funzionalità avanzate di analisi dei dati.

4. Espansione della Rete di Ancore e Miglioramento della Mesh: Prevediamo di espandere la rete di ancore e di ottimizzare ulteriormente la Mesh per garantire una copertura più ampia e una comunicazione più robusta.

5. Dato che operiamo con una rete neurale, esiste la possibilità di incappare in errori di probabilità. Pertanto, uno dei prossimi sviluppi potrebbe consistere nell'avvisare l'utente quando scatta l'allarme per la falling edge (implementata all'interno del Personal Device). Nel caso in cui non si verifichi effettivamente una caduta, sarà data all'utente la possibilità di annullare l'allarme entro un periodo di 10 secondi, qualora si tratti di un falso positivo.

Questi sviluppi futuri riflettono il nostro impegno continuo nel rendere il nostro sistema sempre più efficace e adattato alle esigenze reali degli utenti.

9 Conclusione

In conclusione, il nostro progetto è stato un'entusiasmante percorso di apprendimento che ci ha permesso di esplorare nuove tecnologie, studiare nuove librerie e ampliare le nostre competenze nel campo della comunicazione e del lavoro di gruppo.

La sfida di sviluppare un sistema complesso come il nostro ci ha offerto l'opportunità di mettere in pratica le conoscenze teoriche acquisite durante il corso, superare ostacoli e implementare soluzioni innovative. L'esperienza di lavorare insieme come team ha contribuito significativamente al nostro sviluppo professionale, migliorando le nostre capacità di collaborazione e risoluzione dei problemi.

Desideriamo esprimere la nostra gratitudine al nostro docente, che ci ha guidato e sostenuto lungo questo percorso di apprendimento. La sua competenza e il suo incoraggiamento hanno reso possibile la realizzazione di questo progetto.

Concludiamo questo capitolo del nostro percorso formativo con soddisfazione e gratitudine, pronti ad applicare le competenze acquisite in future sfide.

Figure 12: Gli sviluppatori

Un ringraziamento speciale va a Google, da sempre e per sempre il nostro piu grande alleato da Alessio Tudisco e Miriana Russo