Presentazione del progetto, a cosa serve e perché è stato sviluppato.

Presentazione del progetto, a cosa serve e perché è stato sviluppato.

Siamo due ragazzi del ITT Pacinotti di Fondi che devono iniziare il terzo superiore, che si sono conosciuti al corso di robotica ed hanno voluto ricreare da 0 il casco smart che è stato iniziato da ragazzi del quarto anno che durante una hackathon con Würth e Fondazione Mondo Digitale hanno conquistato il terzo posto per l'idea il 28 novembre 2018. L'idea è stata ripresa dal gruppo di robotica e poi portata avanti da noi due allievi: Oscar di Manno e Peppe Mathew

Il casco da lavoro Smart IoT è un elmetto per lavoratori edili "intelligente" (oppure un casco per moto e bicicletta o controllo bimbi).

Ogni anno avvengono numerosi incidenti sul cantiere per cause anche accidentali che possono essere letali a causa di un elevato tempo di attesa di un intervento di soccorso sommato legato ai ritardi nella scoperta dell'infortunio.

Con questo Casco Smart IoT è possibile conoscere la posizione in tempo reale, anche in caso di occultamento dell'operaio, con latitudine, longitudine ed elevazione, quindi la posizione precisa del "ferito".

Il casco con l'accelerometro riesce a determinare l'eventuale caduta improvvisa o a percepire lo stato di fermo del lavoratore in caso di inattività oltre un certo periodo, mentre con un GPS determina la posizione precisa e la visualizza su mappa, con un sensore di temperatura e umidità la condizione di lavoro.

L'elmetto viene munito anche di un pulsante di emergenza o S.O.S. nel caso in cui il lavoratore richieda un intervento urgente.

Sensoristica a bordo del casco:

Il GPS: Lettura di posizione ed elevazione

Il sensore Global Positioning System è un sensore dotato di una piccola antenna all'esterno del casco per la lettura di varie informazioni riguardo il posizionamento del casco in una mappa. Nello sviluppo del progetto viene effettuato il calcolo della posizione via latitudine e longitudine per la dimostrazione della posizione su una mappa 2D. In caso di soccorso viene in aiuto un terzo valore, ovvero, l'altitudine che viene, per ora, ricavata dal sensore GPS, ma non essendo molto accurata, in un futuro si prevede l'utilizzo di un barometro elettronico per rilevare l'altitudine dal livello del mare al variar della pressione rilevata. Viene utilizzato un tipo a basso consumo e con modalità ECO MODE e POWER SAVE MODE max 50mA

Accelerometro e giroscopio

L'accelerometro e il giroscopio sono due sensori che si trovano in un singolo circuito integrato. L'accelerometro permette di rilevare l'accelerazione lungo i vari assi. Il giroscopio invece permette di rilevare l'accelerazione angolare lungo i vari assi. Questo sensore viene utilizzato per eseguire la lettura dell'accelerazione che ha un lavoratore e combinando l'accelerazione di vari assi consente di capire se un lavoratore ha subito un'inciampata o incrociando i dati con il GPS ed il barometro, rilevare una precipitazione da un'alta quota (ad esempio una caduta dal 1° piano).

Funzionamento del sistema

Il sistema è composto da due "lati".

- Lato client, ovvero, il casco, all'interno contenente una board basata su ATmega328 che invia i dati ad un server.
- Lato server, il lato, che si occupa della ricezione e dell'elaborazione dei dati.

Lato Client – Il Fishino che invia i dati al server. (Si possono usare Arduino mini con accelerometro e altri sensori integrati completi di wifi e in tecnologia basso consumo)

Al "lato" del caschetto troviamo un Fishino UNO che esegue letture da un sensore GPS e da un accelerometro. Durante queste letture la board rimane in attesa del comando dal server cloud ad esempio per innescare un buzzer o altri attuatori. Le prime versioni software utilizzavano il protocollo http con autenticazione via header per l'invio dei dati. Poi ci fu il passaggio al protocollo MQTT, protocollo apposito per la telemetria e contenente meno dati inutili a differenza di http (vedasi ad esempio gli header).

Lato Server – Il server cloud che elabora le richieste e genera gli allarmi

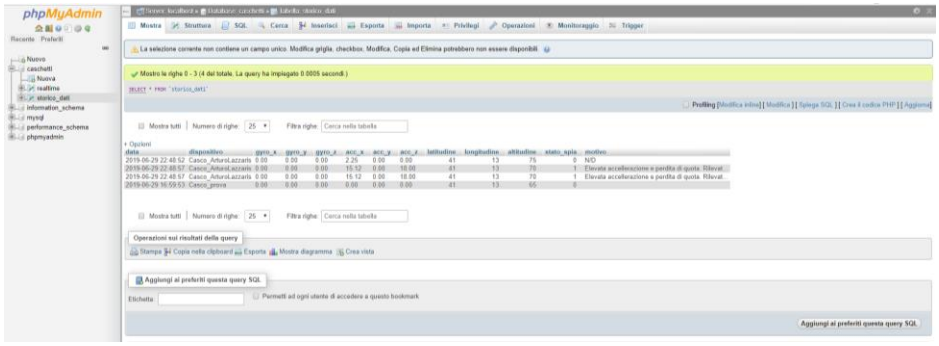
Nel lato cloud invece, troviamo un server web in node.js (apposito per il Real time) che riceve i dati leggendo un topic MQTT e gli incamera in un database MariaDB creando un grande archivio di storico dati per dare la possibilità di generare un grafico o da utilizzare come "Scatola nera". Il server oltre che archiviare i dati in un database li visualizza in tempo reale su un'interfaccia web.

Il Server che si occupa dell'elaborazione dati

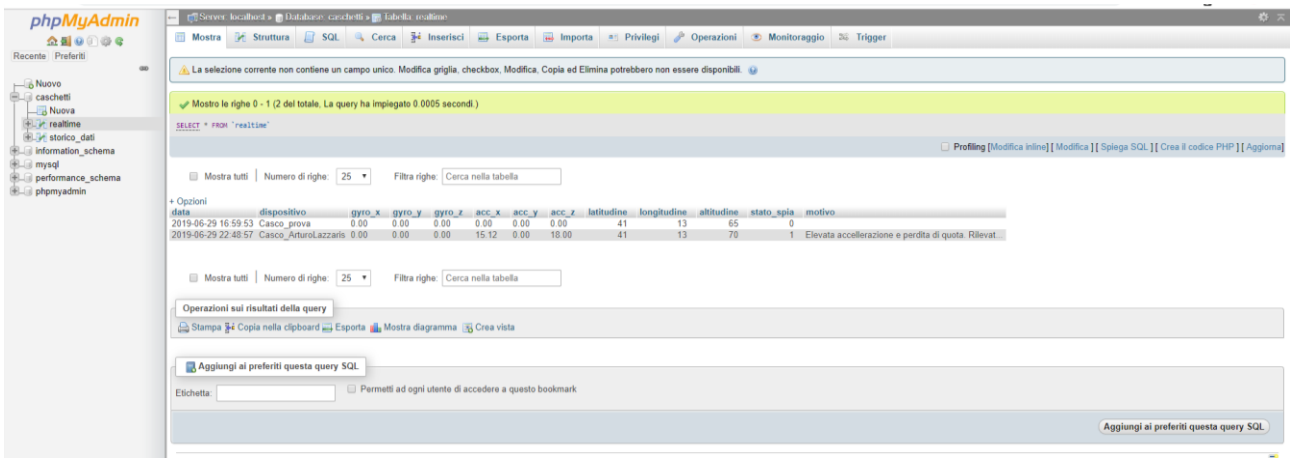
```
root@dimanno:~# neofetch
            .-/+oossssoo+/-.               root@dimanno.space
        `:+ssssssssssssssssss+:`           ------------------
      -+ssssssssssssssssssyyssss+-         OS: Ubuntu 18.04.1 LTS x86_64
    .ossssssssssssssssssdMMMNysssso.       Kernel: 4.15.0
   /ssssssssssshdmmNNmmyNMMMMhssssss/      Uptime: 3 days, 3 hours, 2 mins
  +ssssssssshmydMMMMMMMNddddyssssssss+     Packages: 649
 /sssssssshNMMMyhhyyyyhmNMMMNhssssssss/    Shell: bash 4.4.19
.ssssssssdMMMNhsssssssssshNMMMdssssssss.   Terminal: /dev/pts/0
+ssssshhhyNMMNyssssssssssssyNMMMysssssss+  CPU: Intel Xeon E5-2683 v3 (2) @ 2.499GHz
ossyNMMMNyMMhsssssssssssssshmmmhssssssso   Memory: 701MiB / 4096MiB
ossyNMMMNyMMhsssssssssssssshmmmhssssssso
+ssssshhhyNMMNyssssssssssssyNMMMysssssss+
.ssssssssdMMMNhsssssssssshNMMMdssssssss.
 /sssssssshNMMMyhhyyyyhdNMMMNhssssssss/
  +sssssssssdmydMMMMMMMMddddyssssssss+
   /ssssssssssshdmNNNNmyNMMMMhssssss/
    .ossssssssssssssssssdMMMNysssso.
      -+sssssssssssssssssyyyssss+-
        `:+ssssssssssssssssss+:`
            .-/+oossssoo+/-.

root@dimanno:~#
```

La macchina da noi utilizzata è una macchina virtuale situata su un server. La macchina gira su server Supermicro equipaggiati con doppio Xeon E5 2683 V3, SSD PCIe NVMe Intel, doppia alimentazione e doppia scheda di rete. È equipaggiato anche di protezione contro attacchi DDoS fino a 2Mpps (Milioni di pacchetti al secondo). La nostra macchina utilizza Linux per una miglior performance, esattamente gira Ubuntu Server 18.04 che ospita un server Node.js incaricato della visualizzazione a schermo dei dati e dell'inserimento dei dati all'interno di un server MariaDB che è ospitato anche lui all'interno della macchina e di un Broker MQTT per la ricezione dei dati dal caschetto e per l'invio di comandi da attuare sul caschetto. Dopo aver ricevuto i dati via MQTT il software da noi scritto li inserisce all'interno di un database MariaDB, in due tabelle.

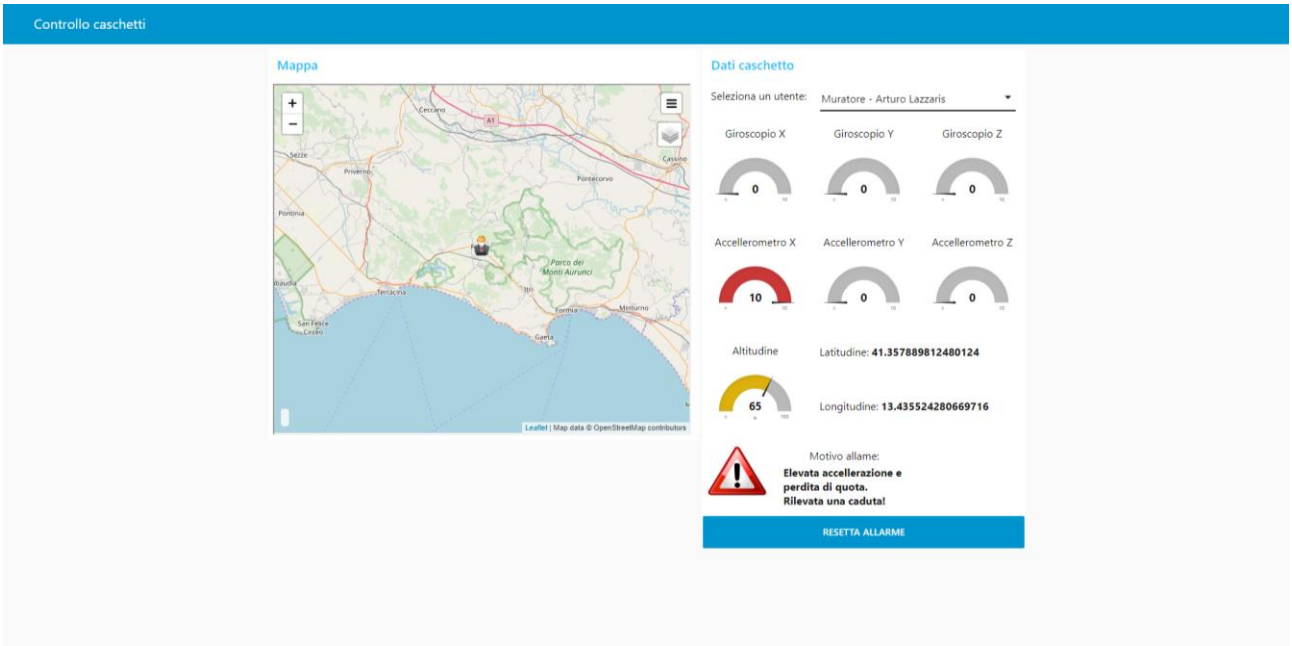


Una tabella rinominata “storico_dati” contiene tutti i vari dati ricevuti dai vari caschi. Questa tabella viene utilizzata per creare grafici in relazione al tempo.



L’altra tabella esistente è la tabella “realtime” che contiene gli ultimi valori ricevuti dai rispettivi caschetti. Codesta tabella viene utilizzata per la visualizzazione in tempo reale dei valori ricevuti dai vari sensori dei caschetti.

L’interfaccia è una pagina web in qui vengono mostrati i dati in tempo reale. L’interfaccia contiene la mappa per la visualizzazione a schermo della posizione dei vari caschetti, vari “manometri/indicatori” che visualizzano i valori di altitudine, accelerazione lineare ed angolare dei caschetti, delle stampe a schermo di testo riguardante latitudine, longitudine e nel caso in cui l’allerta sia accesa, la motivazione dell’allerta, contiene anche un’icona che sia attiva all’attivarsi dell’allerta ed un pulsante che consente di rimuovere lo stato di allerta su quel determinato caschetto.



Il caschetto connesso ad Internet

Il sistema è composto da un Arduino MKR1000 connesso ai vari sensori disposti nelle varie zone del casco. Arduino si occupa di raccogliere i dati vari sensori ed inviarsi al server tramite il protocollo di telemetria MQTT. Fishino riceve i vari dati dai sensori in diverso modo. Dal sensore GPS riceve i dati tramite comunicazione seriale secondo lo standard di comunicazione seriale NMEA 0183, effettua il parsing in locale ed invia latitudine, longitudine ed altitudine al server. Dall'accelerometro invece, riceve i dati secondo il protocollo i2c. Dopo aver fatto la lettura dei dati, invia i dati tramite una stringa JSON via protocollo MQTT.

La sicurezza dell'Internet Of Things

Per una miglior sicurezza nello scambio dei dati, abbiamo preso tutte le precauzioni necessarie. Il sito web in cui vengono letti i vari dati è crittografato con un certificato SSL per prevenire una lettura dei dati a malintenzionati tramite tecnica del Man In The Middle e richiede un login per entrare nel portale. Per inviare i dati al Broker MQTT è richiesta autenticazione e per prevenire il MITM è applicata una crittografia tramite certificato SSL, applicabile anche su Arduino per prevenire un eventuale invio "falsificato" dei dati al server.

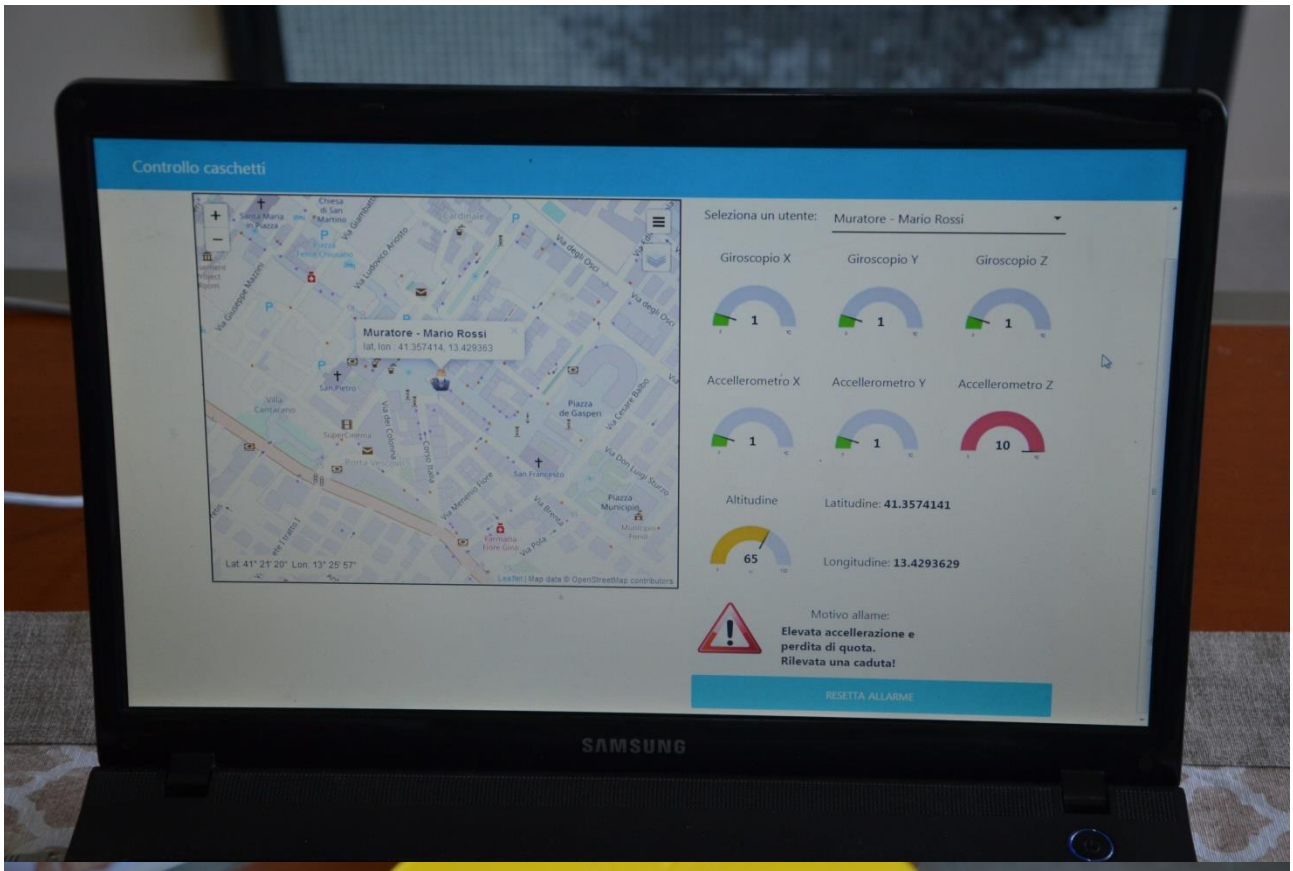
ALLARMI : Gli allarmi avvengono tramite segnalazione sul sinottico e allarme acustico da casse collegate alla scheda audio del PC, inoltre il segnale audio attiva una lampada Rossa girante.

Un messagio viene invato con l'aplicazione TELEGRAM sullo smartphone:

Il caso presentato è reale dal server.

Alimentazione Smart del caschetto:

Il caschetto viene alimentato con batteria al litio da 3000 mA con ricarica USB ma migliorabile con sistema a induzione in modo da ricaricare il casco appendendolo nella portacaschi di fine lavoro. Come avviene per i nuovi smartphone. L'autonomia è per una giornata lavorativa di 10 ore per un consumo di 70mA del sistema acceso. Le parti elettroniche visibili sotto il casco possono essere inserite in minicontenitori posti ai lati del casco. Batteria su un lato ed elettronica sull'altro lato.


Controllo caschetti
Seleziona un utente:
Muratore - Mario Rossi
Giroscopio X
Giroscopio Y
Giroscopio Z
Accellerometro X
Accellerometro Y
Accellerometro Z
Altitudine
Latitudine: 41.3574141
Longitudine: 13.4293629
Motivo allarme:
Elevata accellerazione e perdita di quota.
Rilevata una caduta!
RESETTA ALLARME
SAMSUNG