



2024/2025



STUDENT SOFTWARE HOUSE

HOME MONITORING

by: **Ivan Torre** e **Sofia Barillaro**

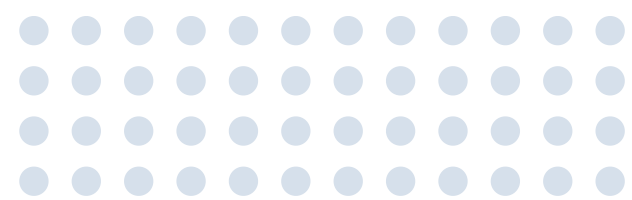
referenti Interni: **Lorena Marassi** ed **Emanuela Manzini**

referenti esterni: **Luca Martinelli** ed **Alessandro Bigi**.



START PRESENTATION





DOMOTICA: UNA CASA SMART

La domotica è l'insieme delle tecnologie e dei sistemi che consentono di automatizzare e controllare a distanza funzioni e dispositivi all'interno di un'abitazione o di un edificio.

Rientrano quindi nella domotica:

Sistemi di videosorveglianza

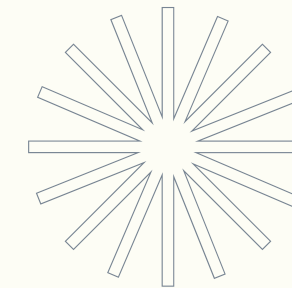
Elettrodomestici smart



Controllo energetico

Sistemi di sensoristica

→ PERCHÉ HOME MONITORING?



Il progetto richiedeva la costruzione di un architettura che:

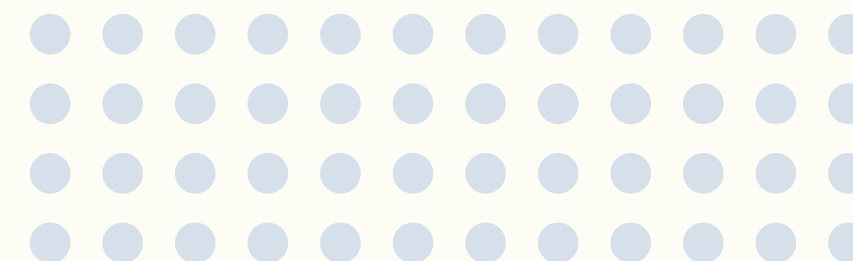
- rendesse possibile l'interfacciamento ad un sensore da remoto,
- desse la possibilità all'utente di inviare comandi al sensore e attraverso una **dashboard** di conoscere le misurazioni effettuate, relativamente a:

Ossido di Azoto (NO)

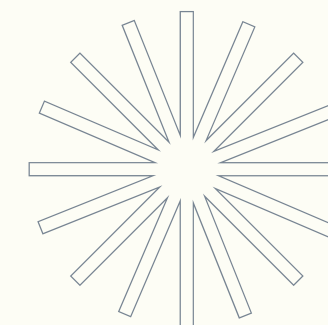
Umidità

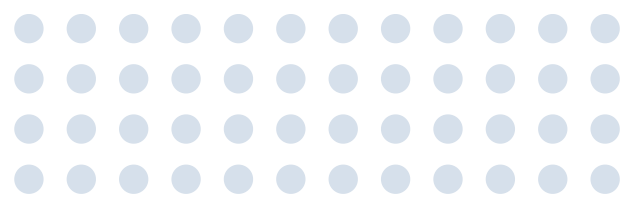
Temperatura

Pressione



- l'architettura può facilmente espandersi con l'utilizzo di più dispositivi.





PROGETTAZIONE

Durante la fase di progettazione, in assenza di vincoli infrastrutturali, l'architettura è stata selezionata valutando varie caratteristiche:

Hardware: ESP32 (MCU) + sensori ambientali + moduli di notifica.

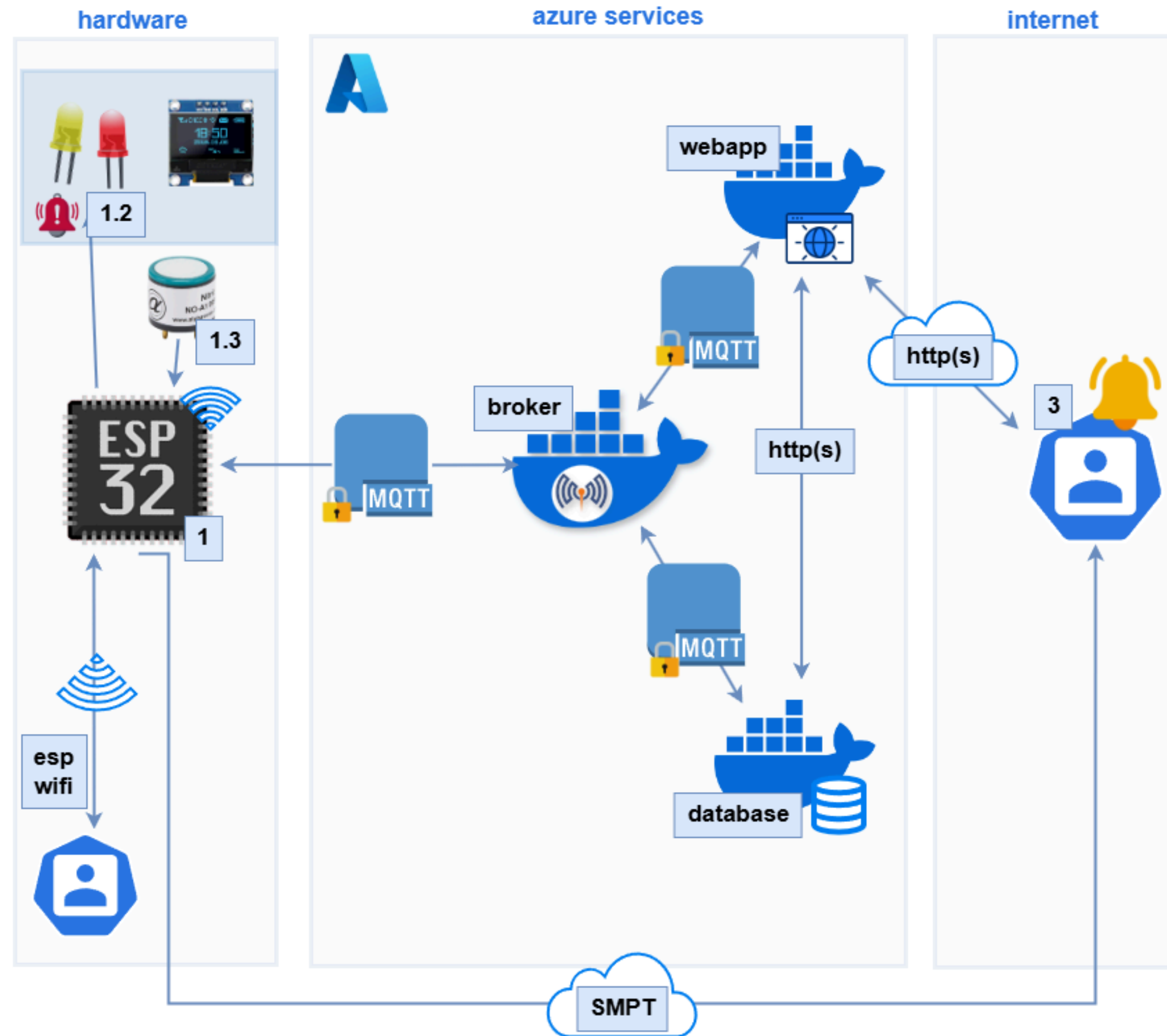
Software: valutate due soluzioni:

- Azure IoT Hub (Paas - Platform as a Service Microsoft, comunicazione in MQTT).
- Microservizi Docker su Azure Container Instances (ACI).



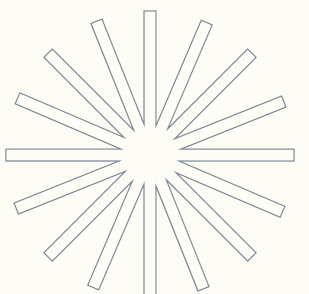
Si è adottata l'architettura a *microservizi* in ACI per maggiore flessibilità di sviluppo, controllo, e autonomia nel deployment, nonostante complessità superiore rispetto a IoT Hub.

ARCHITETTURA





TECNOLOGIE UTILIZZATE



Cursor



Docker



Mosquito



Azure



Python

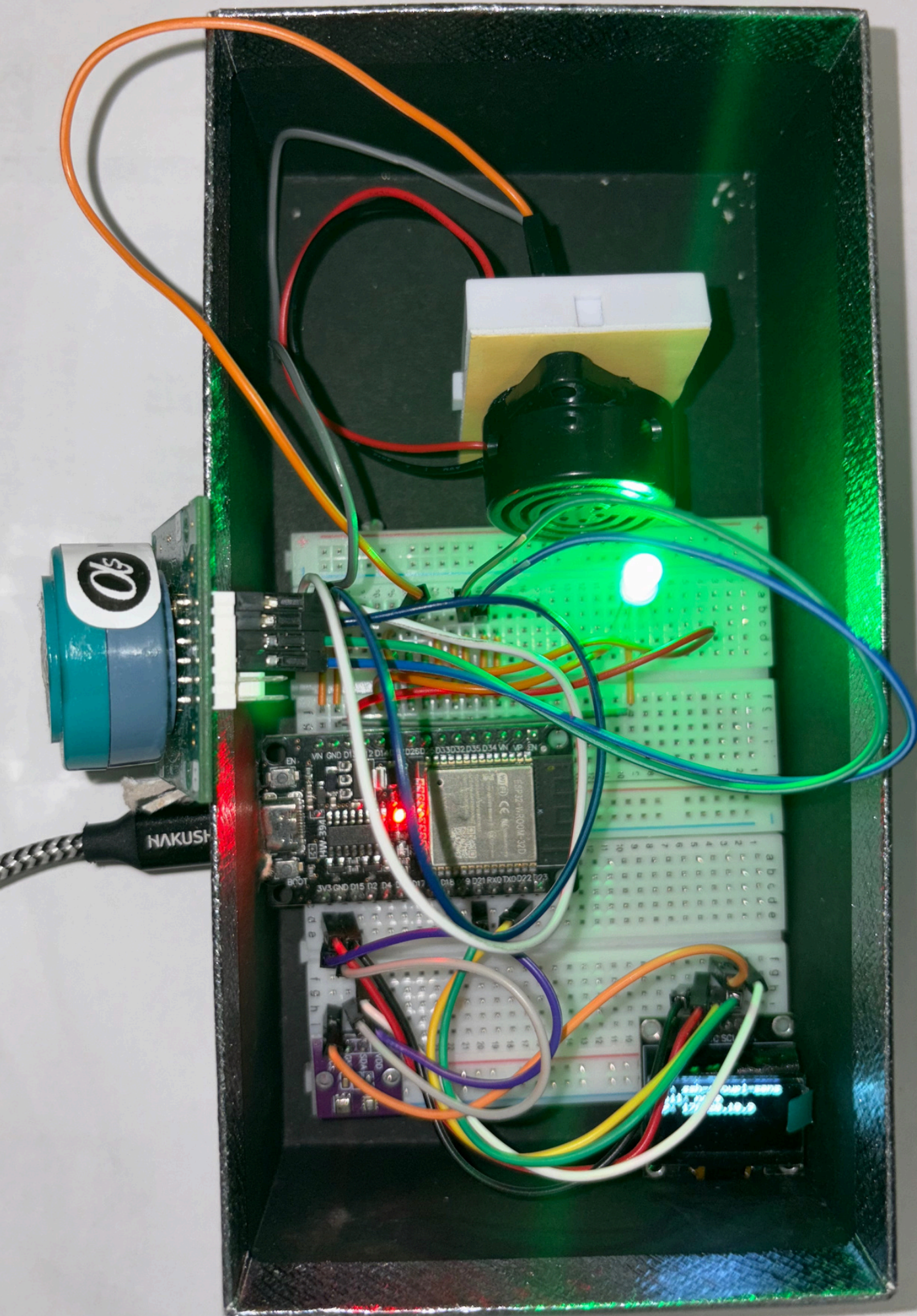
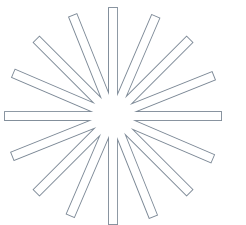
PROTOCOLLI UTILIZZATI



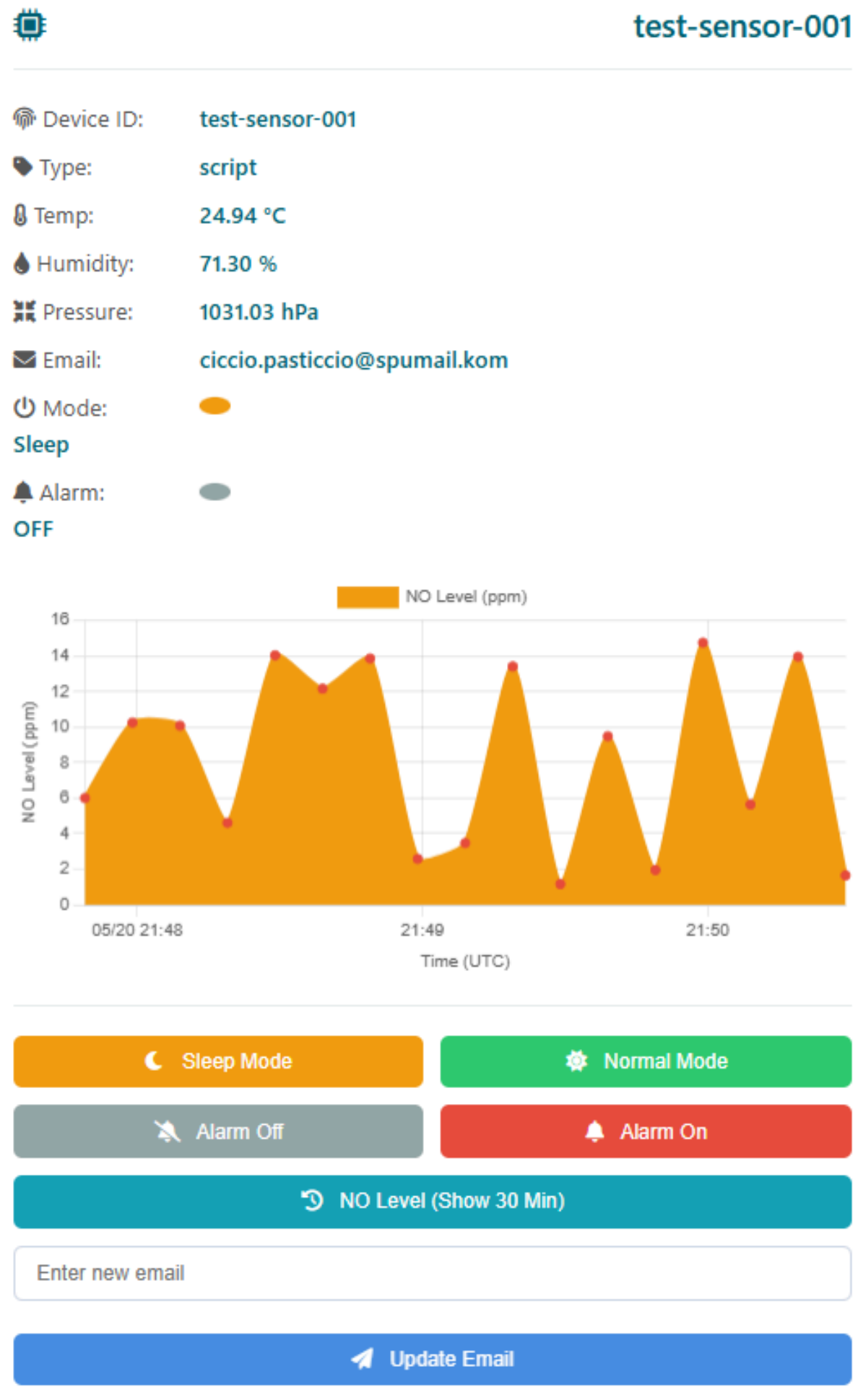
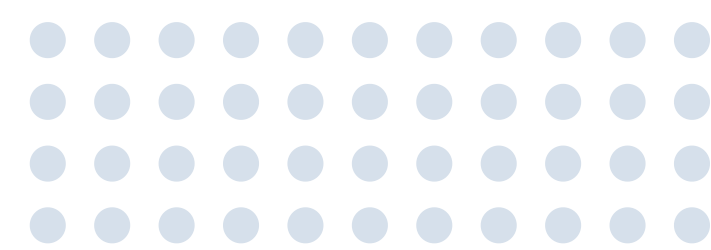
SVILUPPO HARDWARE

Il progetto ha previsto lo sviluppo completo della componente hardware del sensore, in particolare:

- Acquisizione di dati dal sensore NO AlphaSense seguendo indicazioni del datasheet (assenza di librerie specifiche)
- ESP32 con Wi-Fi provisioning per un architettura generale:
 - credenziali in flash e, in caso di connessione fallita, modalità AP con HTTP server per inserimento SSID/password.
- Sistema di notifica attraverso led + allarme

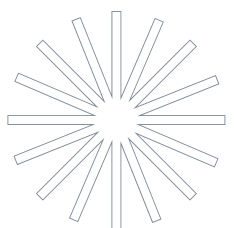


DASHBOARD

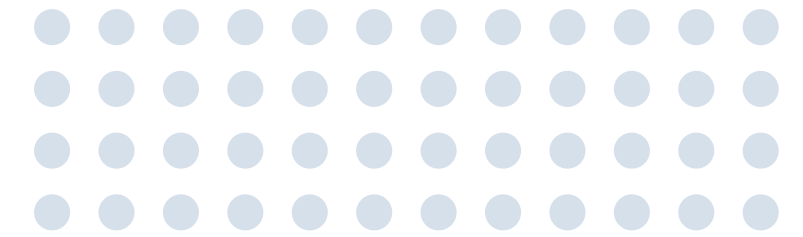


La dashboard permette la comunicazione con l'utente, al quale è permesso:

- L'inserimento o rimozione di più sensori attraverso SensorID
- La visualizzazione di:
 - parametri ambientali in tempo reale (temperatura, umidità, NO, ecc.)
 - Grafico dinamico dell'andamento di NO con evidenza colorata delle anomalie
- L'invio di comandi verso ESP32:
 - Attivazione/disattivazione allarme
 - Modifica frequenza di campionamento
 - Settaggio indirizzo email per notifiche

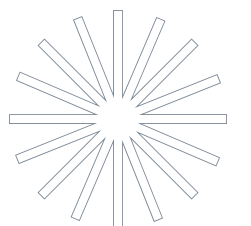


DIFFICOLTÀ RISCONTRATE

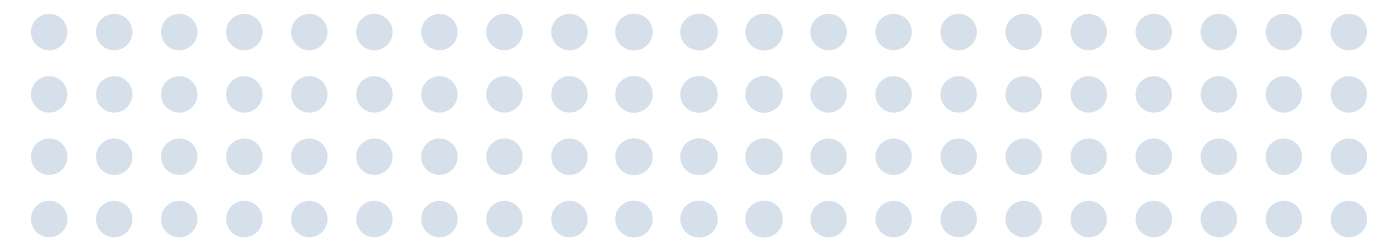


- Imparare come utilizzare ACI, ACL e Azure in generale
- Necessità di valutare numerose architetture
- Far funzionare efficacemente elementi separati che comunicano fra loro
- Difficoltà nel management dei docker su Azure e necessità di creare script di start, stop e push
- Interfacciamento al sensore

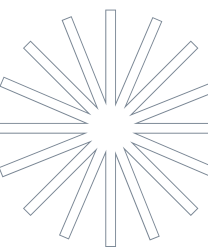
Viste le numerose difficoltà è stata faticosa anche la gestione delle tempistiche, soprattutto considerando l'approccio progressivo degli incontri con i referenti

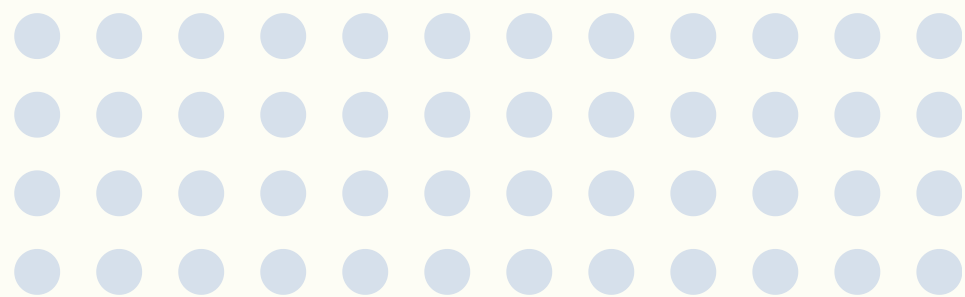
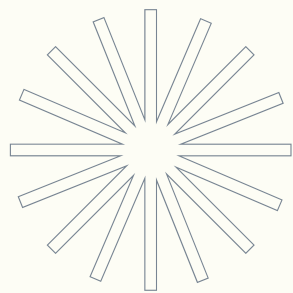


EVOLUTIVE FUTURE



- HTTP e MQTT in chiaro per rapid prototyping; file di configurazione già predisposti per implementare **TLS** in produzione
- Aggiunta di sistema di **autenticazione** da implementare una volta presente TLS
- Abilitare **aggiornamenti software** via HTTPS.
- **Machine Learning**: introdurre modelli predittivi quando il dataset sarà sufficientemente ampio.

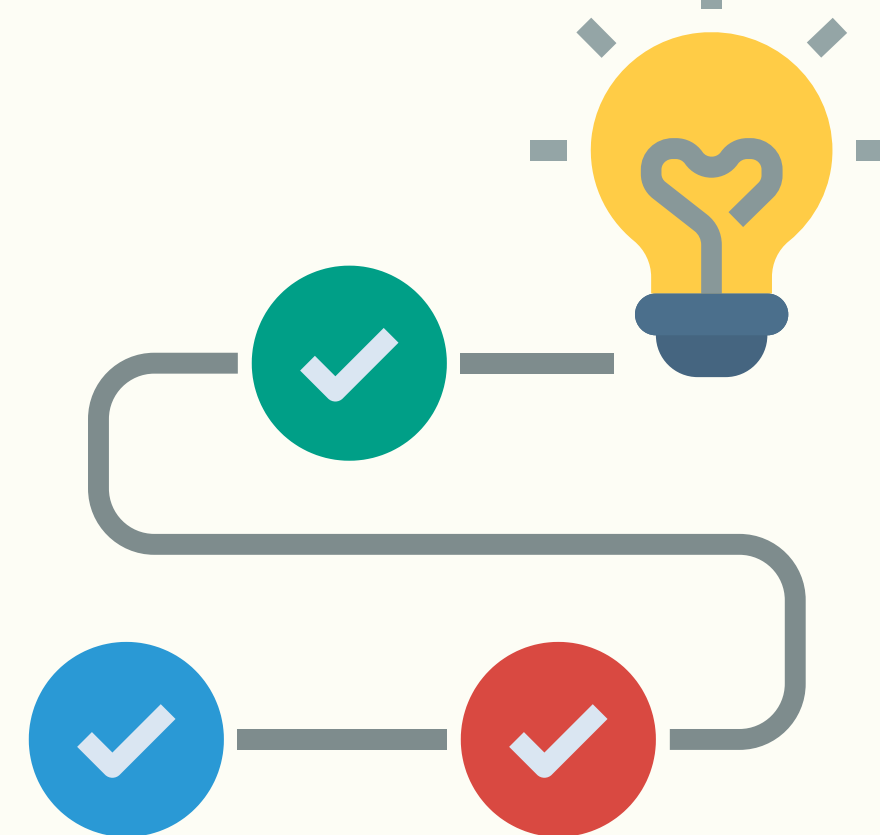


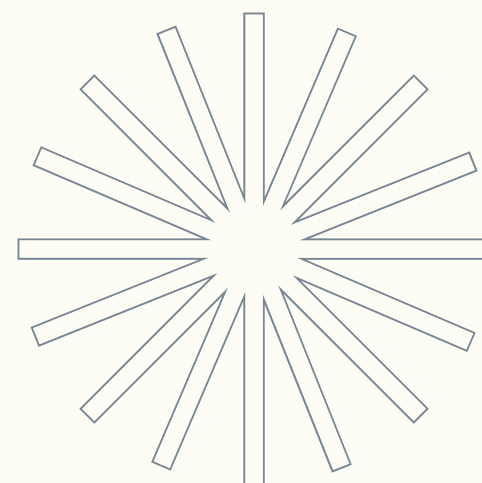
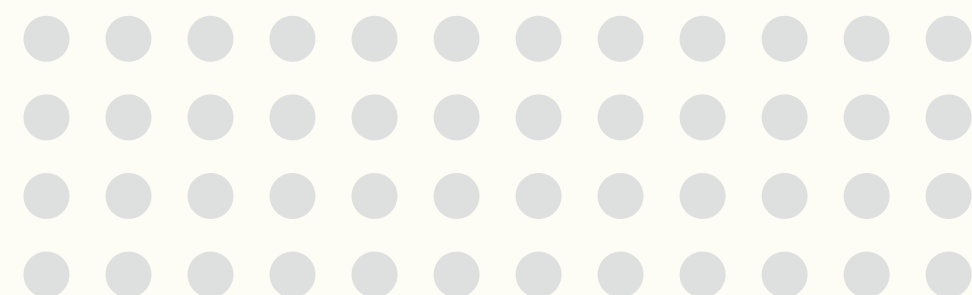


CONCLUSIONI

La soluzione progettata:

- risponde all'input iniziale,
- integra tecnologie apprese nell'anno scolastico con competenze acquisite durante lo svolgimento del progetto,
 - potrebbero rivelarsi un valore aggiunto in un futuro ambito aziendale (Microservizi docker, Azure)
- Rappresenta un interessante prototipo, pur non essendo ancora una soluzione pronta per un utilizzo finale, data la complessità e le potenzialità di espansione del progetto.
- In futuro potrebbe essere migliorata e trasformata in un sistema di domotica smart completo e scalabile.



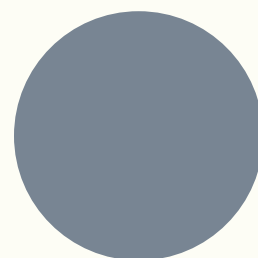


BIBLIOGRAFIA

Documentazione Azure

Documentazione broker mqtt

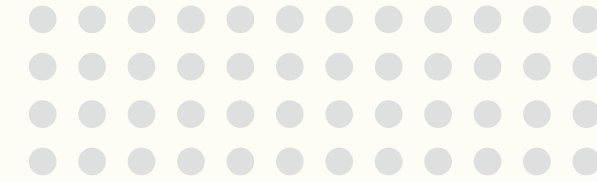
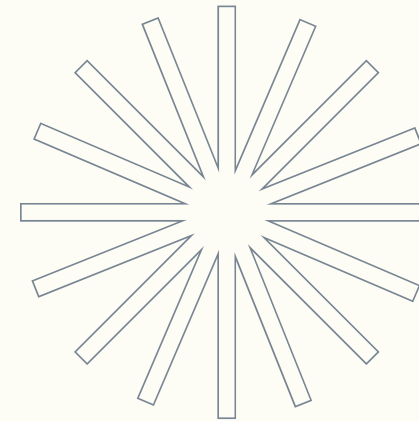
Documentazione del sensore



Si precisa che la documentazione del sensore ci è stata direttamente fornita dal referente esterno



“Efficiency opens up more time to focus on the things that really matter.”



THANK YOU!



nota: le pagine successive alla slide finale servono per rispondere ad eventuali domande specifiche o per aggiungere informazioni se dovesse rimanere tempo.
Non seguono quindi uno schema logico e contengono informazioni limitate

12:22 

 5G  52

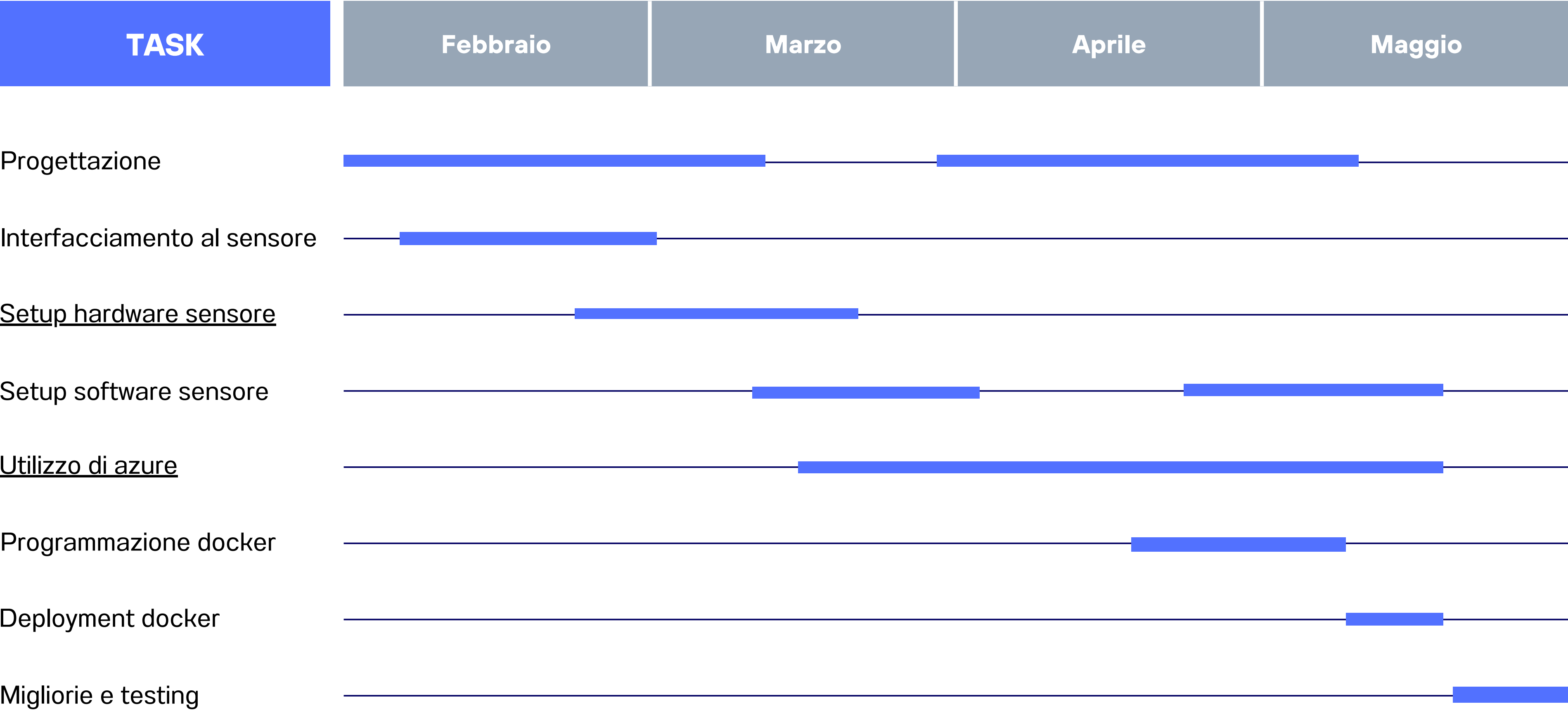
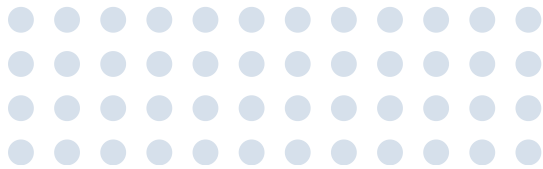
WiFi Setup

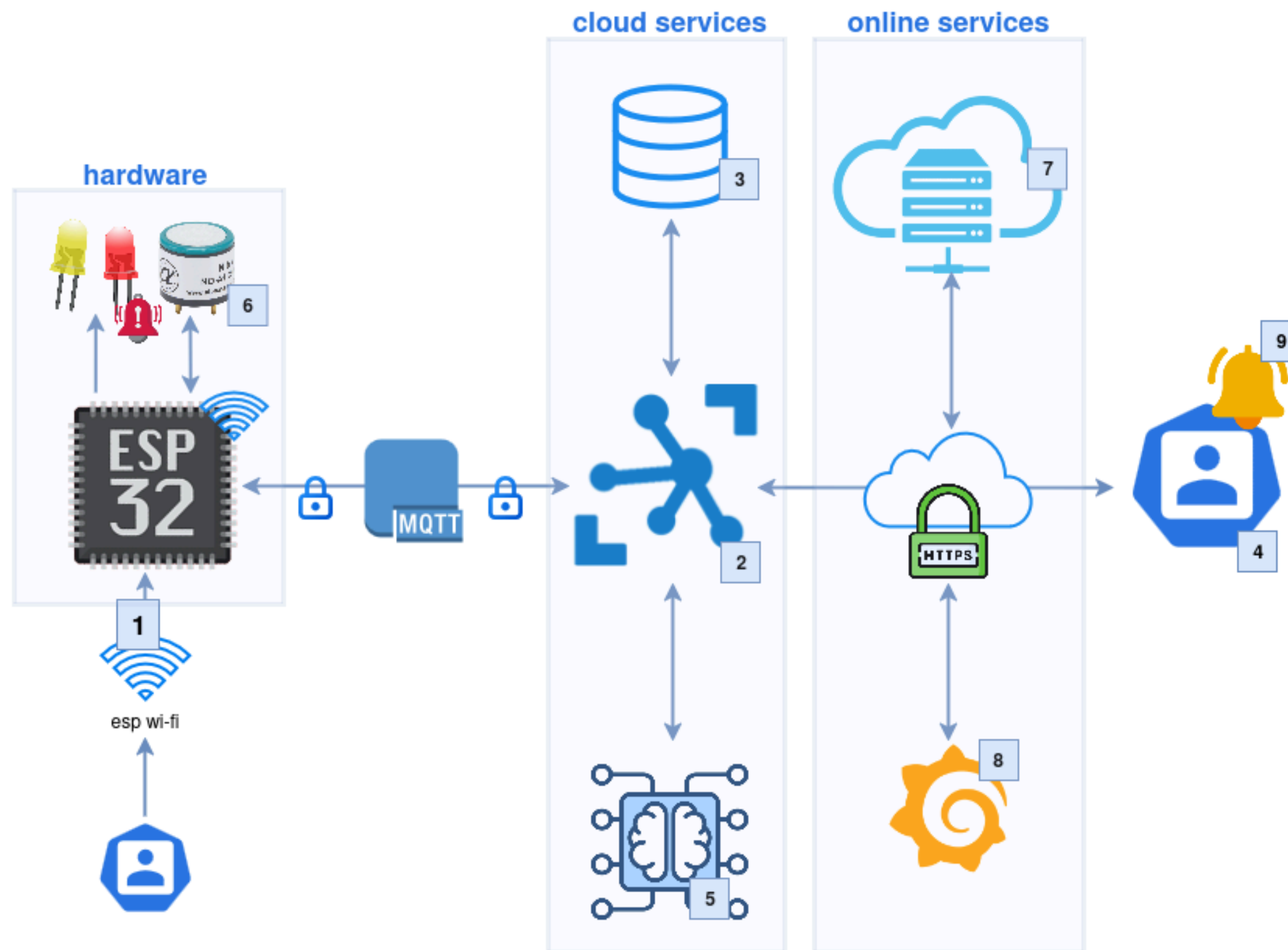
SSID:

Password:

Save

Gantt diagram





PROGETTAZIONE

L'esp necessita di una connessione a internet. Può salvare fino a 3 credenziali per connettersi al wifi (1) in memoria. Se l'esp non riesce a connettersi a nessuno dei wifi in memoria, diventa lui stesso un AP per ottenere dall'user delle nuove credenziali.

Trovare un modo per salvare le credenziali in modo sicuro, decidere come connettersi all'esp quando non è connesso al wifi. (bluetooth, wifi esp?)

2) L'esp comunica attraverso MQTT con un servizio in cloud (ad esempio azure iot hub) che si occupa di:

- salvataggio dei dati (es degli ultimi 30 giorni) in un database (3), in cloud.
- running di uno script di machine learning (5) il quale effettua delle analisi sul funzionamento del sensore
- pubblicazione dei dati su uno web server (7) e su una dashboard grafica es grafana (8), ai quali l'utente può accedere con delle credenziali.

il programma ha 3 modalità:

- **sleep**, una misurazione al min?
- **normal**, una misurazione ogni 30 sec
- **danger**, si attiva automaticamente se la presenza di NO supera un certo valore. una misurazione ogni 10? 15? secondi. Se i valori tornano nella norma si resetta dopo n minuti.

La mode si seleziona dallo web server, in caso contrario la scelta è fatta dal sensore.

7) web server

- comunica con l'utente, mostra i dati del sensore + alcune informazioni,

da la possibilità di mandare alcuni comandi. (selezionare la mode, modificare il limite, spegnere l'allarme)

all'utente vengono inviate delle notifiche (9) (mail/ messaggio) in alcune situazioni, esempio:

- si attiva la modalità danger,
- NO supera una certa soglia e il problema non viene risolto.
- possibili guasti (analisi del machine learning (5), i dati arrivano corrotti / non arrivano)

possibili migliorie:

pannelli solari ?? gabbia di faraday per il sensore? Sono necessari messaggi di hello?

LIBRERIE/TECNOLOGIE USATE

descrizione esaustiva

- Docker & Docker Compose
- Eclipse Mosquitto v2.0 (MQTT Broker)
- Python (Paho-MQTT, Flask)
- SQLite
- React & Chart.js
- ESP32 (C/C++ firmware + HTTP provisioning)
- Arduino(C)
 - PubSubClient
 - Adafruit
 - Esp mail client
- Azure Container Registry (ACR)
- Azure Container Instances (ACI)
- Azure CLI