Analisi funzionale: descrivi gli obbiettivi del tuo pezzo di progetto, le funzionalità principali e il flusso di lavoro.

Per la parte frontend del progetto Serra, ho sviluppato un'interfaccia web moderna e intuitiva che si compone di due pagine principali: una pagina di login e una dashboard di monitoraggio. La pagina di login presenta un design moderno con effetto vetro e gradiente verde-blu (#43cea2, #185a9d), fornendo un form di accesso con validazione degli input e toggle della password. L'interfaccia è stata realizzata utilizzando Bootstrap 5.3.2 per garantire un layout responsive e una UX ottimale su tutti i dispositivi. La dashboard principale è strutturata in tre sezioni di monitoraggio:

1. Temperatura Ambiente
2. Umidità del Terreno
3. Livello Acqua e Controllo Pompa

Ogni sezione utilizza un carousel per visualizzare grafici Grafana con viste giornaliere, settimanali e mensili. L'integrazione è realizzata tramite iframe, permettendo una visualizzazione in tempo reale dei dati. Il sistema di controllo della pompa è stato implementato con particolare attenzione al feedback visivo:

* Indicatore di stato con tre colori (rosso per inattiva, giallo per transizione, verde per attiva)
* Pulsanti di controllo

Per la parte tecnica, ho utilizzato:

* HTML5 e CSS3 con variabili per un tema consistente
* JavaScript vanilla per la gestione degli eventi
* Font Awesome 6.0.0 per le icone
* Design responsive con breakpoint ottimizzati
* Grafana per la visualizzazione dei dati tramite dei grafici in tempo reale

Ho posto particolare attenzione alla gestione degli errori e al feedback utente, implementando:

* Validazioni lato client per prevenire operazioni non valide
* Alert informativi per l'utente
* Transizioni animate per i cambi di stato

Il codice è stato strutturato in modo modulare e mantenibile, con una chiara separazione tra HTML, CSS e JavaScript, e ampio utilizzo di commenti per facilitare future modifiche o estensioni.

Analisi della strumentazione utilizzata: Elenca e descrivi i software, i protocolli, i dispositivi e gli strumenti utilizzati

Nell'implementazione del frontend della Dashboard Serra, ho effettuato un'attenta analisi comparativa delle tecnologie disponibili, valutando tre aspetti principali: framework JavaScript, framework CSS e protocolli di comunicazione.

**Framework JavaScript**

Ho considerato l'utilizzo di React e Angular, ma ho optato per Vanilla JavaScript con Bootstrap per diverse ragioni:

* La dimensione contenuta del progetto non giustificava l'overhead di un framework complesso
* Le funzionalità richieste erano ben definite e limitate
* La necessità di performance ottimali e caricamento veloce
* La semplicità di manutenzione e debug

**Framework CSS**

Tra Tailwind CSS, Material UI e Bootstrap, ho scelto Bootstrap 5.3.2 per:

* Rapidità di sviluppo grazie ai componenti predefiniti
* Sistema grid intuitivo e responsive
* Eccellente compatibilità con Vanilla JavaScript
* Facilità di personalizzazione tramite variabili CSS

**Visualizzazione Dati**

Ho confrontato Chart.js, D3.js e Grafana, scegliendo Grafana per:

* Specializzazione nella visualizzazione di dati in tempo reale
* Dashboard predefinite per il monitoraggio
* Integrazione semplice via iframe
* Riduzione significativa del codice custom necessario
* Interfaccia professionale con aggiornamenti fluidi
* Scalabilità e facilità di espansione

Questa combinazione di tecnologie si è rivelata ottimale per il progetto, offrendo un equilibrio tra performance, manutenibilità e velocità di sviluppo. La scelta di evitare framework JavaScript complessi ha mantenuto il codice leggero e debuggabile, mentre Bootstrap ha garantito un'interfaccia responsive professionale. Grafana ha completato il stack fornendo visualizzazioni dati avanzate con minimo sforzo di sviluppo.

Diagramma di sequenza UML: disegna o allega il diagramma UML che rappresenta l'interazione tra i componenti del tuo progetto.  
Spiega brevente il diagramma.

Il diagramma di sequenza UML rappresenta le interazioni chiave nel frontend della Dashboard Serra, evidenziando quattro flussi principali:

1. **Processo di Login**: Mostra l'autenticazione dell'utente e l'accesso alla dashboard.
2. **Monitoraggio Dati**: Illustra come la dashboard integra i pannelli Grafana per visualizzare i dati ambientali in tempo reale.
3. **Attivazione Pompa**: Rappresenta la sequenza di azioni quando l'utente avvia la pompa, inclusi gli stati transitori (giallo) e lo stato finale (verde).
4. **Spegnimento Pompa**: Mostra il processo inverso, con la transizione dallo stato attivo a inattivo attraverso uno stato transitorio.

Il diagramma evidenzia:

* L'interazione tra i cinque componenti principali (Utente, Login, Dashboard, Grafana, WebSocket Server)
* Il feedback visivo immediato per ogni azione dell'utente
* La gestione granulare degli stati della pompa
* L'integrazione trasparente dei pannelli Grafana

Questa architettura garantisce:

* Responsività dell'interfaccia con feedback immediato
* Manutenibilità grazie alla chiara separazione dei componenti
* Scalabilità per future estensioni del sistema

Query sviluppata: scrivi la query principale realizzata per il progetto e spiega cosa fa

*SELECT*

*timestamp as time,*

*valore as value,*

*tipo as metric*

*FROM misurazioni m*

*JOIN sensore s ON m.sensoreid = s.sensoreid*

*WHERE s.modello = 'TMP-100'*

*ORDER BY timestamp;*

**Spiegazione:**

* **Scopo**: Recupera lo storico delle misurazioni di temperatura ambiente nel tempo
* **Struttura**:
* **SELECT**:
* timestamp as time: Timestamp della misurazione (rinominato per compatibilità Grafana)
* valore as value: Valore della temperatura rilevata
* tipo as metric: Tipo di misurazione come metrica
* **JOIN**: Collega le misurazioni ai dati del sensore
* **WHERE**: Filtra solo le misurazioni del sensore modello 'TMP-100'
* **ORDER BY**: Ordina cronologicamente i risultati

**Utilizzo:**

* Visualizzazione dell'andamento temporale della temperatura
* Creazione di grafici time-series
* Base per calcoli statistici (min, max, media)

*SELECT*

*timestamp as time,*

*valore as value,*

*s.serraid as serra\_name*

*FROM misurazioni m*

*JOIN sensore s ON m.sensoreid = s.sensoreid*

*WHERE $\_\_timeFilter(timestamp)*

*AND m.tipo = 'temperatura'*

*ORDER BY timestamp DESC*

*LIMIT 1*

**Spiegazione:**

* **Scopo**: Monitora l'ultima temperatura rilevata per il sistema di alerting
* **Struttura**:
* **SELECT**:
* timestamp as time: Momento della rilevazione
* valore as value: Ultima temperatura registrata
* s.serraid as serra\_name: Identificativo della serra per contestualizzare l'alert
* **WHERE**:
* $\_\_timeFilter(timestamp): Macro Grafana per filtrare il range temporale
* m.tipo = 'temperatura': Filtra solo le misurazioni di temperatura
* **ORDER BY DESC + LIMIT 1**: Recupera solo l'ultima misurazione

**Utilizzo:**

* Sistema di alerting in tempo reale
* Monitoraggio valori critici
* Trigger per notifiche

**Differenze Chiave:**

1. **Finalità**:

* Prima query: Analisi storica e visualizzazione trend
* Seconda query: Monitoraggio real-time e alerting

1. **Dati Restituiti**:

* Prima query: Serie temporale completa
* Seconda query: Singolo valore più recente

1. **Ottimizzazione**:

* Prima query: Ottimizzata per analisi storiche
* Seconda query: Ottimizzata per risposta rapida

1. **Parametri**:

* Prima query: Filtro statico sul modello sensore
* Seconda query: Filtro dinamico temporale con macro Grafana

**Vantaggi dell'Approccio:**

1. **Separazione delle Responsabilità**:

* Query distinte per visualizzazione e alerting
* Migliore gestione delle risorse del database

1. **Performance**:

* Query ottimizzate per i rispettivi casi d'uso
* Indici appropriati possono essere creati

1. **Manutenibilità**:

* Query chiare e ben strutturate
* Facili da modificare e debuggare

1. **Scalabilità**:

* Facilmente adattabili per nuovi sensori
* Possibilità di aggiungere nuove metriche