

ANALISI TECNICA

1. Architettura Generale

- Tipo: Architettura *client-server* con comunicazione in tempo reale.
- Tecnologie consigliate:
 - Frontend
 - Backend
 - Database: MySQL
 - Realtime layer: WebSocket
 - Controllo qualità (hardware): sensori ottici + modulo integrato in backend via API locali o seriali

2. Componenti Tecnici Principali

2.1 Backend

Componente	Funzionalità
Data Receiver	Ricezione dei dati dai robot
Processo dati	Aggregazione per ora e minuto
WebSocket	Invio dati al frontend in tempo reale
Salvataggio dati	DataBase SQL

2.2 Frontend

Componente	Funzionalità	Tecnologie
------------	--------------	------------

Dashboard	Visualizzazione di grafici interattivi	??
Filtro per Colore	Selettore di prodotti	Selezione
Gestione Avvisi	Notifiche in caso di errore o assenza di dati	Banner Dinamici
Log Produzione	Lista prodotti con annessi dettagli	Tabella interattiva
Responsività	Adattamento a più dispositivi diversi	CSS/HTML/BootStrap

3. Flussi di Dati

3.1 Acquisizione Dati

- I robot inviano i dati (colore, timestamp, codice prodotto).
- Il backend riceve i dati.
- I dati vengono salvati nel DB e inviati contemporaneamente al frontend tramite WebSocket.

3.2 Elaborazione

- I dati vengono aggregati per intervalli di tempo dinamica.
- I risultati sono utilizzati sia per il grafico in tempo reale, sia nel log.

3.3 Visualizzazione

- Il frontend riceve i dati in tempo reale.
- Il grafico si aggiorna dinamicamente in base ai filtri selezionati.
- Il log viene aggiornato in tempo reale, con evidenziazione dell'ultimo elemento.

4. Database Design (Relazionale)

Tabella: prodotti

Campo	Tipo	Descrizione
id	INT	Identificativo Univoco
timeStamp	DATA	Data e ora di produzione
colore	INT	Codice Colore HEX
codiceProdotto	CHAR	Codice identificativo
esitoControllo	BOOLEAN	Esito del controllo di qualità
altro	JSON	Parametri Aggiuntivi

Tabella: aggregati_orari

Campo	Tipo
id	INT
ora	DATA
numProdotto	INT
numeroErrore	INT

Tabella: aggregati_minutari

| Struttura simile a quella oraria, ma per intervalli da 1 minuto |

5. Gestione Errori / Inattività

- Il backend rileva se per un certo intervallo (es. 5 min) non sono stati ricevuti dati → genera un messaggio di inattività.
- Il frontend, ricevuto l'evento, mostra:
 - Banner rosso o giallo in alto
 - Notifica a comparsa (toast)
- Logging su file o DB per ogni anomalia riscontrata

6. Requisiti Non Funzionali

Categoria	Dettagli
Performance	Bassa latenza nella trasmissione dati (< 1s)
Scalabilità	Architettura non modulare, possibilità di scalare più impianti
Affidabilità	Controlli ridondanti sulla validità dei dati ricevuti
Sicurezza	Autenticazione

Usabilità	Interfaccia semplice, accessibile anche da utenti non esperti
Manutenibilità	Codice modulare, documentato

7. Riepilogo delle tecniche / tecnologie da adottare

Livello	Tecnologie
Frontend	CSS / HTML / BootStrap
Backend	Java / ?
Database	MySQL
Realtime	WebSocket
Sincronizzazione	GitHub

8. Conclusione

L'analisi tecnica qui proposta parte dalla tua dettagliata analisi funzionale e fornisce una base solida per la progettazione, sviluppo e deployment del sistema. I prossimi passi potrebbero includere:

- Definizione dei casi d'uso e user flow
- Progettazione dettagliata UML / ER
- Creazione dei mock-up UI

- Pianificazione sprint (metodologia Agile)