Relazione app web full-stack per progetto "VITA"

1 Introduzione

L'implementazione del progetto ha seguito un approccio strutturato per garantire efficienza, modularità e usabilità. Sono state adottate tecnologie moderne e metodologie di sviluppo per assicurare un'architettura solida e scalabile. In questa relazione, verrà descritto nel dettaglio cosa è stato realizzato, come è stato sviluppato e come deve essere utilizzato il sistema.

2 Cosa è stato fatto

L'implementazione ha riguardato la realizzazione di un'applicazione che consente agli operatori di monitorare in tempo reale l'attività degli utenti che utilizzano un visore VR. L'applicazione è stata progettata per raccogliere, elaborare e visualizzare dati provenienti dalla sessione dell'utente, permettendo un'interazione fluida e precisa tra il visore e il sistema di monitoraggio.

Le principali componenti sviluppate includono:

- **Interfaccia Web**: Un pannello di controllo per gli operatori che consente la gestione delle sessioni e la visualizzazione dei dati.
- **Comunicazione via Socket**: Un sistema di scambio dati tra la web app e l'app Unity per garantire la sincronizzazione in tempo reale.
- **Database**: Archiviazione strutturata dei dati relativi agli utenti e alle sessioni per un'analisi successiva.
- **Integrazione con il visore VR**: Un sistema di comunicazione che permette al visore di inviare dati all'applicazione.

3 Come è stato fatto

L'implementazione si è basata su un'architettura client-server. Il server è stato sviluppato utilizzando **Flask**, un framework leggero per il backend, con gestione delle richieste HTTP e WebSocket per la comunicazione in tempo reale. Il frontend è stato costruito con **HTML, CSS e JavaScript**, integrando librerie come **Bootstrap** per un'interfaccia moderna e reattiva.
Il database è stato implementato utilizzando **SQLite**, garantendo semplicità nella gestione dei dati. Le sessioni degli utenti vengono registrate e associate a ID univoci per una tracciabilità efficace.

L'integrazione con Unity è stata realizzata attraverso **C# e WebSocket**, consentendo al visore di trasmettere informazioni sullo stato dell'utente e di ricevere istruzioni dal server.
Per la gestione della comunicazione, sono stati definiti protocolli specifici per garantire che i messaggi scambiati tra il visore e l'applicazione siano chiari e coerenti. Ogni pacchetto di dati inviato segue una struttura predefinita per evitare errori di interpretazione.

4 Come deve essere usato

L'applicazione è progettata per essere utilizzata dagli operatori che monitorano le sessioni degli utenti con il visore VR. Il flusso di utilizzo è il seguente:

1. **Avvio del server**: L'operatore deve avviare il server Flask per gestire le richieste e stabilire la comunicazione con il visore.

2. **Connessione del visore**: Il visore VR, attraverso Unity, si collega al server e inizia a trasmettere dati.

3. **Monitoraggio in tempo reale**: L'operatore può visualizzare in un'interfaccia web i dati dell'utente in tempo reale, verificando il corretto svolgimento della sessione.

4. **Registrazione dei dati**: I dati raccolti vengono salvati nel database per analisi successive.

5. **Chiusura della sessione**: L'operatore può terminare la sessione, salvando un report finale sui progressi dell'utente.

L'interfaccia è stata progettata per essere intuitiva e accessibile, con grafiche chiare e strumenti interattivi per facilitare l'analisi delle sessioni.

5 Conclusione

L'implementazione del progetto ha seguito un approccio modulare, permettendo un'integrazione fluida tra il visore VR e il sistema di monitoraggio. L'uso di tecnologie moderne e protocolli di comunicazione ha reso possibile uno scambio dati affidabile e in tempo reale. Il sistema è pronto per essere utilizzato dagli operatori per migliorare il monitoraggio delle sessioni VR e garantire un'esperienza più efficace per gli utenti finali.