Specifiche

Requisiti Funzionali

L'applicazione consente agli utenti di registrarsi con un nickname univoco e autenticarsi tramite login. Dopo il login, l'utente può caricare nuovi brani musicali sul sistema (inserendo titolo, artista, album, anno, genere e file audio più copertina). I brani caricati sono associati al profilo dell'utente. L'utente può creare playlist personalizzate assegnando un nome alla playlist (univoco per utente) e selezionando uno o più brani da aggiungere. Nella versione in RIA è presente una funzionalità di **"reorder"** che permette di riordinare appunto i brani nella playlist. Cliccando su una playlist, l'utente vede i brani in essa contenuti (5 brani per pagina), con la possibilità di avviare la riproduzione di un brano grazie a un player audio integrato (pagina di player dedicata o 'mini-player' sovrapposto).

L'interfaccia offre notifiche di feedback all'utente, ad esempio messaggi di conferma tramite **toast** (piccoli popup) dopo una registrazione avvenuta, creazione di playlist o caricamento di brano riuscito, oppure messaggi di errore in caso di credenziali errate o dati non validi.

Requisiti non funzionali

L'applicazione è stata progettata ponendo attenzione a usabilità e responsività. L'interfaccia utente adotta un tema grafico ispirato ad **Apple Music**, con design **responsive** (layout adattivo e uso di unità relative/clamp per dimensioni di font e spaziature) e "componenti riutilizzabili" (ad esempio il mini-player flottante presente su più pagine, e modali riutilizzate per inserimento di dati).

Il sistema garantisce buone prestazioni grazie alla versione RIA che riduce i caricamenti completi di pagina utilizzando chiamate AJAX asincrone: questo migliora la reattività percepita, ad esempio quando si aggiungono brani a una playlist o si carica un nuovo brano, gli aggiornamenti avvengono senza ricaricare l'intera pagina.

Dal punto di vista della sicurezza e consistenza dei dati, sono implementati controlli sia lato client sia lato server: i form HTML utilizzano attributi di validazione (es. required, minlength, formato type=email/number etc.), mentre lato server sono presenti controlli sui parametri (ad es. verifica che l'anno sia un numero valido nell'intervallo atteso) e vincoli a livello di database (es. vincoli di unicità e integrità referenziale).

Inoltre, appositi **filtri** prevengono accessi non autorizzati e garantiscono che ogni utente interagisca solo con le proprie risorse.

Il codice è strutturato per essere **manutenibile**, con separazione in layer (DAO per l'accesso al database, Servlet controller per la logica di business, pagine/JS per la presentazione) e con due implementazioni (HTML e RIA).