Relazione Progetto

Wizard – Gioco di Carte Multiplayer

Autori:

Colombi Maria Chiara
Schianchi Andrea

Luglio 2025

1. Introduzione

Wizard è un'applicazione web multiplayer che implementa il gioco di carte *Wizard*, un gioco di strategia e previsione in cui i giocatori devono indovinare il numero di "prese" (trick) che realizzeranno in ogni round.

Il progetto è sviluppato secondo un'architettura **client-server**, utilizzando le seguenti tecnologie:

- **Frontend**: React.js
- **Backend**: Node.js con Express
- **Comunicazione in tempo reale**: WebSocket
- **Database**: MySQL per la persistenza dei dati

2. Architettura del Sistema

2.1. Frontend (Client)

Il frontend, sviluppato in React.js, gestisce tutte le interazioni con l'utente. Le schermate principali sono:

- `HomeScreen.js`: schermata iniziale con animazioni e pulsante di accesso
- `Login.js`: modale per login e registrazione
- `AccessRoomScreen.js`: dashboard per utenti registrati
- `GuestRoomScreen.js`: versione semplificata per accesso come ospite
- `RoomScreen.js`: interfaccia principale del gioco, con gestione carte e turni
- `Card.js e CardImages.js`: componenti per visualizzare le carte
- `App.js`: gestisce le varie componenti e la connessione

Funzionalità principali:

- Autenticazione (login, registrazione, ospite)
- Lobby per la gestione delle stanze
- Interfaccia responsive con animazioni e aggiornamenti in tempo reale

2.2. Backend (Server)

Il backend è realizzato con Node.js ed Express. Fornisce:

- API REST per autenticazione e gestione delle stanze
- WebSocket per la comunicazione in tempo reale tra giocatori
- Logica di gioco: distribuzione delle carte, gestione turni, calcolo punteggi

File principali:

- `index.js`: configurazione server (Express + WebSocket)
- `Room.js`: gestione logica della stanza di gioco
- `Player.js`: modello del giocatore (carte, punteggi)
- `authController.js`: interazioni col database (autenticazione e gestione utente)
- `saveGame.js`: salvataggio risultati nel database

2.3. Database

Il database MySQL gestisce la persistenza dei dati, con le seguenti entità principali:

- `users`: utenti registrati
- `matches`: partite giocate
- `users_matches`: relazione tra utenti e partite (risultati, punteggi)

3. Funzionalità Implementate

3.1. Autenticazione e Profilo

- Registrazione con validazione (es. username "guest" non permesso)
- Login con password crittografata (`bcrypt`)
- Accesso temporaneo come ospite
- Profilo utente con statistiche: partite giocate, vittorie, punteggio totale

3.2. Gestione Stanze

- Creazione stanza con nome e numero massimo di giocatori
- Join in stanze pubbliche
- Aggiornamenti in tempo reale tramite WebSocket (ingresso/uscita giocatori, avvio partita)

3.3. Logica di Gioco

- Distribuzione carte progressiva (2 al primo round, 3 al secondo, ecc.)
- Fase di dichiarazione (*bid*): ogni giocatore indica quante prese intende fare
- Fase di gioco:
  - Obbligo di seguire il seme, se possibile
  - Determinazione vincitore del trick in base alle regole del gioco

- In caso di disconnessione di un giocatore a partita iniziata, viene automaticamente inserito un **bot** che prende il suo posto e continua la partita.
- Calcolo punteggi:
  - +10 punti +5 per ogni presa se la dichiarazione è corretta
  - -10 punti se la dichiarazione è errata

3.4. Fine Partita e Persistenza

- Visualizzazione risultati finali e vincitore
- Salvataggio su database: nome stanza, punteggi, data

4. Tecnologie Utilizzate

Categoria	**Tecnologie**
Frontend	React.js, CSS-in-JS
Backend	Node.js, Express, WebSocket
Database	MySQL
Autenticazione	bcrypt
Gestione Stato	React Hooks, Context API