Gioco P2P di combattimento tra elicotteri

Sviluppato da:
Lazzari Matteo

Data inizio progetto:

08/03/2021

Indice

Specifica dei requisiti

Lato server:

Per poter giocare, è necessario avviare il server node col comando node server.js

Per poter giocare, è necessario avviare un server peerjs alla porta 9000 col comando peerjs –port 9000

Il server necessita del database MYSQL progettato per poter funzionare

Il server necessita dei moduli MYSQL, socketio, express, crypto per poter funzionare

Lato client:

Il client necessita delle immagini/gifs per poter visualizzare l'interfaccia grafica (le immagini non sono inviate dal server)

Il client necessita di un minitor con risoluzione 1920x1080. Il gioco non è responsive, è comunque possibile giocarci con risoluzioni minori/maggiori, ma non si avrà la completa visione della mappa

Per giocare è necessario che entrambi i giocatori abbiano il gioco aperto e non in scheda, altrimenti provoca problemi di sincronizzazione

Progettazione del sistema

Il progetto è strutturato secondo l'architettura client-server.

Il server è stato sviluppato con la tecnologia Nodejs. Il server utilizza socket TCP e SocketIO per la comunicazione tra client e server. Il server utilizza un database MYSQL per salvare le informazioni della partita terminata e gestire i login dei client.

Il server è progettato in modo che un client quando cerca una partita ottiene sempre il 1° client disponibile nella lista. Questo fa sì che la lista sia sempre di lunghezza 0 o 1

Per lo sviluppo del server è stato utilizzato Visual Studio Code

Il client è stato sviluppato utilizzando HTML5, insieme a CSS e JavaScript. Oltre a queste principali tecnologie ho utilizzato anche JQuery per la gestione di alcuni eventi e la gestione dinamica della pagina.

Per poter giocare è necessario avere le immagini del gioco. Queste sono incluse nella cartella del client. Questo però fa sì però che il client deve per forza avere un "client" di gioco per poter giocare, invece di una semplice pagina web che poi richiede le informazioni al server.

Per lo sviluppo del client è stato utilizzato Visual Studio Code

Come utilizzare il sistema in locale

Prima di iniziare:

È necessario avviare il database MYSQL, successivamente avviare il server NodeJS e il server PeerJS

Quando tutte le applicazioni sono in esecuzione, basta aprire il documento GameScreen.html per poter giocare.

È possibile eseguire la registrazione/login del proprio account cliccando sui bottoni “Registrati” o “Login”.

Se il login avviene correttamente, viene registrata nella memoria del web il peer e le vittorie del relativo giocatore (le vittorie vengono utilizzate per determinare quali elicotteri/razzi sono sbloccati)

La pagina cleanStorage.html viene utilizzata per cancellare i valori del peer e le vittorie dalla memoria

Una volta che si ha scelto l’elicottero e il razzo che si vogliono utilizzare, tra quelli sbloccati, è possibile giocare in 2 modi diversi: Giocare contro un giocatore preciso (inserire il peer del giocatore nella casella e cliccare gioca) oppure giocare contro un giocatore casuale. In entrambi i casi, quando si clicca che si vuole giocare, non è poi possibile cambiare l’elicottero/razzo scelti, ma è comunque possibile cliccare nuovamente sul bottone per impostare lo stato come “non pronto” e poter cambiare la scelta.

Nel caso in cui si giochi contro uno specifico giocatore, la partita inizierà nel momento in cui entrambi i giocatori avranno inserito il peer dell’avversario e cliccato di giocare.

Nel caso in cui si giochi contro un giocatore random, il giocatore manda un messaggio al server per richiedere se ci sono giocatori attualmente in attesa o meno. Nel caso ci siano, ottiene il peerID e avvia un handshake con il 2° giocatore per stabilire la connessione da entrambe le parti. Nel caso non ci siano giocatori in attesa, si riceve un messaggio di attesa. La partita prima di iniziare ha un countdown di 5 secondi

Quando la partita è iniziata, ogni giocatore si può muovere nella window con i tasti WASD (rispettivamente, su-sinistra-giù-destra), puntare l’elicottero verso la direzione del mouse e cliccare “Barra Spaziatrice” per sparare il razzo.

La partita termina quando uno dei due giocatori segna per primo 8 distruzioni sull’avversario.

Se un giocatore esce mentre la partita è in corso, l’altro giocatore vince automaticamente per abbandono. Se un giocatore esce mentre è in attesa di trovare una partita, viene rimosso dalla coda del server. Se esce una volta che ha dato conferma di giocare ad un altro giocatore (ma l’altro giocatore non ha ancora cliccato che è pronto), viene resettato lo stato nell’altro giocatore.

Funzionamento tecnico del server

Il server mette a disposizione per i client diverse funzioni:

- IwantPlay: Metodo che viene chiamato dal client per richiedere al server se sono presenti altri giocatori che hanno cliccato di giocare con giocatori casuali. Nel caso la lista sia di lunghezza 1, allora è già presente un client e il server restituirà al client il peerID dell'altro giocatore. Se la lista è lunga 0, viene aggiunto il peerID alla lista e inviato al client il messaggio di wait
- notReady: Metodo chiamato dal client quando il giocatore clicca di non essere pronto. Per evitare conflitti, controlla prima la lunghezza se è maggiore di 0 (sempre verificata, ma giusto per evitare crash del server)
- needHelicopter: Metodo chiamato dal client quando il giocatore clicca su un elicottero. Viene passato il nome dell'elicottero, che rappresenta poi la chiave primaria sul database. Tramite una quary si ottengono tutte le informazioni dell'elicottero selezionato
- needRocket: Metodo chiamato dal client quando il giocatore clicca su un razzo. Viene passato il nome del razzo, che rappresenta poi la chiave primaria sul database. Tramite una quary si ottengono tutte le informazioni del razzo selezionato
- Login: Metodo che viene chiamato dal client quando il giocatore clicca per il login. Viene controllato se lo Username/Email inserito è valido e la password è corretta. La password è passata in chiaro, ma viene verificata dal server criptandola con l'hash function md5 (funzione migliore di SHA256). Le informazioni sono passate come unica stringa, ma separate tramite una ', '. La prima parte contiene l'email/username, la 2° la password. Se password e Username/Email coincidono, viene mandato indietro il messaggio di conferma del login.
- checkPresence: Metodo che viene chiamato dal client quando il giocatore clicca per registrarsi. Il messaggio inviato contiene diverse parti, divise da una ' , '. La 1° parte contiene il nome utilizzato come email/username, mentre la 2° parte contiene il tipo di ricerca da fare. Ritorna un valore Found o No in base a se è stato trovato o meno un giocatore con le stesse credenziali
- Register: Metodo chiamato dal client quando è stata verificata la disponibilità delle credenziali inserite. Registra il giocatore nel database
- getWins: Metodo utilizzato per ritornare il numero di vittorie del peer che ha effettuato la chiamata. Viene passato il peerID che è chiave primaria
- registerGame: Metodo chiamato a fine partita (o se un client si disconnette durante il game). Utilizzato per registrare nel database la partita con le relative informazioni (peerIDs, punteggi, razzi/elicotteri utilizzati…)

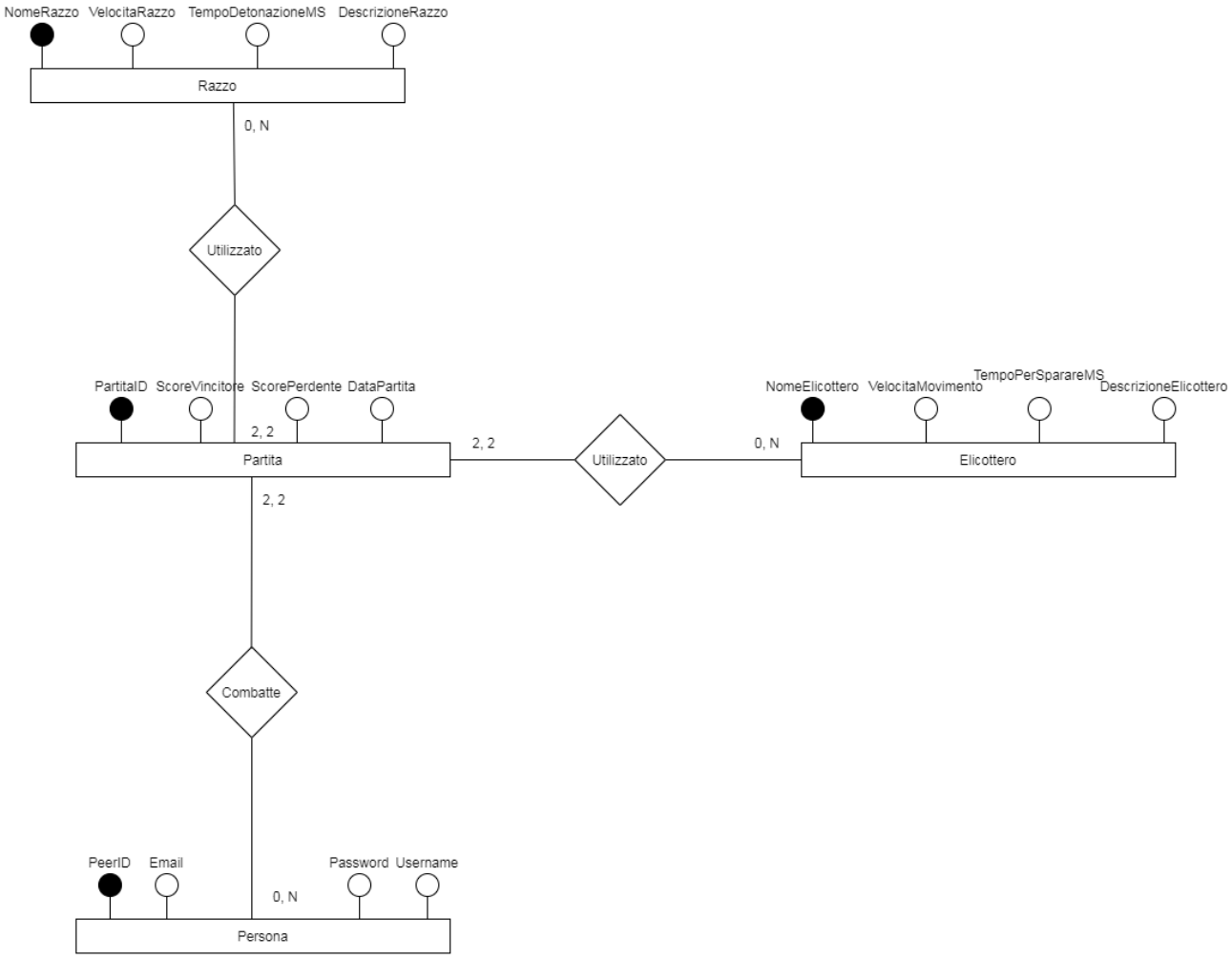
Funzionamento tecnico del client

Il client ha molti metodi che eseguono tutti funzioni diverse. Alcuni sono più importanti, altri sono più situazionali.

Questi sono solo i metodi principali per il funzionamento del client:

- moveHelicopter: Il metodo viene chiamato ogni 20ms. In base al tasto premuto (WASD), muove l'elicottero lungo gli assi (Su-Sinistra-Dietro-Destra)
- fireRocket: Metodo chiamato ogni 20ms. Quando viene premuta "Barra spaziatrice", viene sparato il razzo. Dopo che è stato sparato, viene chiamato il metodo waitForFire che blocca la possibilità di sparare altri razzi per il tempo prestabilito.
- moveRocket: metodo chiamato ogni 20ms. In base all'angolo del razzo, viene mosso lungo la direzione originale. Se il razzo ha un tempo di detonazione, viene decrementato in modo continuo fino a 0, che genererà una esplosione
- rotateRocket: Metodo che viene utilizzato per ruotare l'angolo dei razzi teleguidati. In base a dove punta il mouse e dove si trova, il razzo cambia angolazione, che poi verrà utilizzata da moveRocket per lo spostamento nella mappa
- checkGameStart: Metodo che viene chiamato ogni 40ms. Controlla continuamente se entrambi i giocatori che stanno giocando sono pronti e ancora non si è mai spawnati, allora chiama il metodo respawn per far comparire l'elicottero e avvia la partita
- checkExplosions: Metodo che viene utilizzato per gestire la lunghezza dell'animazione dell'esplosione. Ogni gif dell'esplosione dura 760ms. Tramite un contatore, quando quel valore arriva a 0, viene rimossa l'esplosione
- expandBlackhole: Metodo simile a checkExplosions. La differenza è che aumenta in modo continuo la grandezza dei buchi neri all'interno della mappa, fino a quando non raggiunge la dimensione massima prestabilita
- collider: Metodo molto importante, gestisce tutte le collisioni tra elicotteri-razzi ed elicotteri-buchi neri. Calcola una distanza tra il razzo e il giocatore. Se quella distanza è inferiore ad un determinato valore, viene triggerata l'esplosione.
- Respawn: Metodo che viene chiamato all'inizio della partita o quando un giocatore viene distrutto. Nel 1° caso, aggiunge semplicemente l'elicottero alla window e invia all'avversario le informazioni dell'elicottero. Nel 2° caso viene chiamato da collider quando un elicottero è stato colpito da un razzo. Genera una nuova posizione iniziale e la invia all'altro giocatore e incrementa il punteggio del nemico, inviando quello nuovo. Inoltre, quando finisce di aggiornare i punteggi, controlla la condizione di vittoria con checkWin
- checkWin: metodo utilizzato per controllare se la partita è stata vinta dall'avversario. Controlla l'avversario in quando checkWin è chiamato quando il giocatore corrente è stato distrutto dall'avversario, quindi l'avversario ha guadagnato punti. Se arriva ad una certa soglia, viene inviata la chiamata al server per registrare la partita e si viene riportati al menù tramite la funzione backToMenu dopo 5 secondi

Database

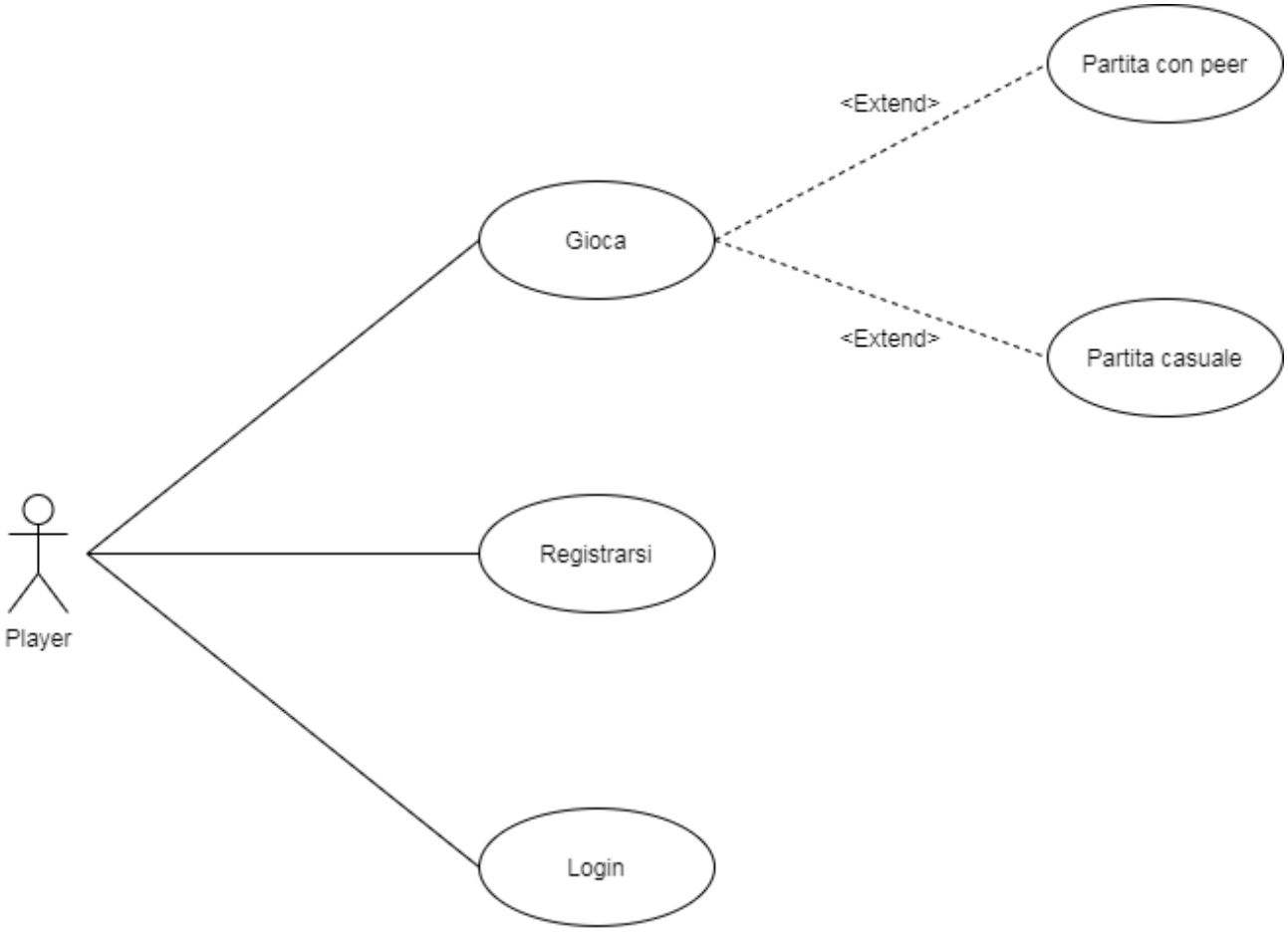


Schema logico

Persona (<u>PeerID</u>, Email, Password, Username)
Elicottero (<u>NomeElicottero</u>, VelocitaMovimento, TempoPerSpararеMS, DescrizioneElicottero)
Razzo (<u>NomeRazzo</u>, VelocitaRazzo, TempoDetonazioneMS, DescrizioneRazzo)
Partita (<u>PartitaID</u>, VincitoreID, PerdenteID, ScoreVincitore, ScorePerdente, DataPartita, ElicotteroVincitore, ElicotteroPerdente, RazzoVincitore, RazzoPerdente)

Use cases

Menù:



In gioco: