

# Gioco Multiplayer TicTacToe

## Descrizione

L'applicazione fornisce all'utente un'interfaccia utente dinamica divisa in una parte di autenticazione, una lobby in cui gli utenti possono visualizzare la classifica e sfidare altri utenti uno alla volta, infine una parte in cui si svolge il gioco.

## Tecnologie utilizzate

- **Lato Server** : applicazione express per node js per hostare il server in localhost con ip statico
- **Lato Client** : utilizzo di template css e html personalizzati in base all'esigenza e javascript per l'interazione con il server.

## Librerie

- **socket.io**: comunicazione bidirezionale realtime tra i web client e i server.
- **mysql** : per l'interazione con il database
- **express**: per creare applicazioni web

## Editor e IDE

- Web Storm
- NotePad++

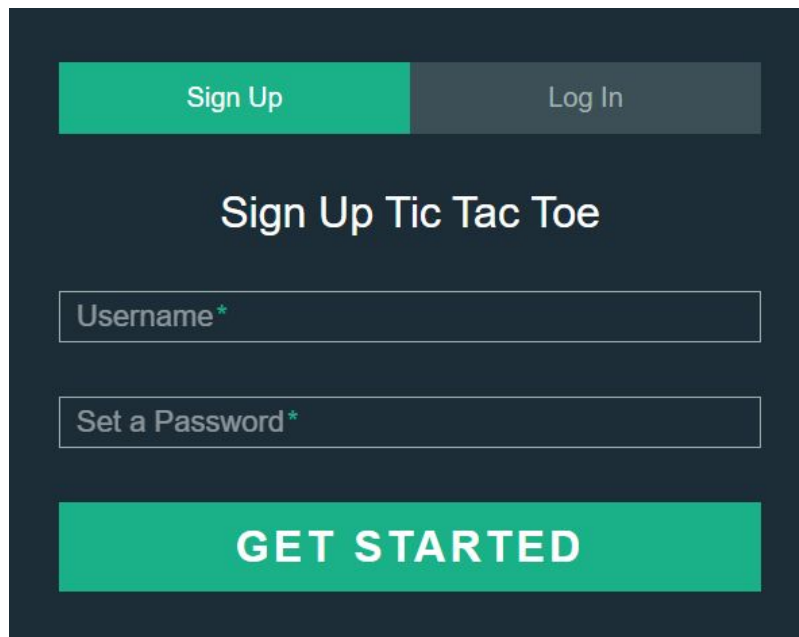
## Funzionamento

### Connessione

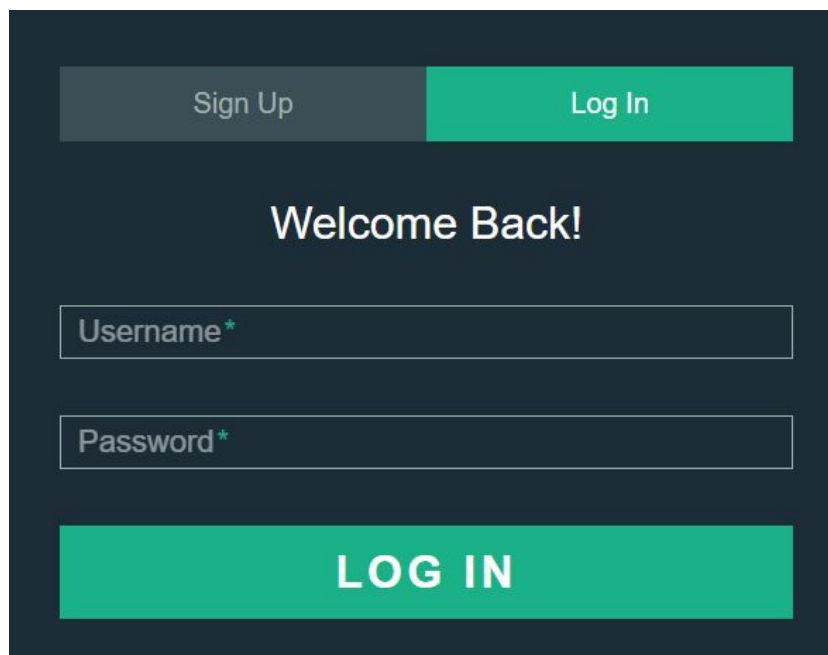
Lato server rimane in ascolto (su porta e ip statici) di eventuali richieste di connessione di client e di eventuali richieste da parte di client in ascolto. Ad ogni connessione il server aggiunge ad una lista le socket utilizzate per la connessione definendoli "ospiti" perchè non autenticati e fornisce la pagina html per l'autenticazione.

### Autenticazione

- **Registrazione**: lato client consiste in un'interfaccia grafica in cui è possibile inserire il proprio username da utilizzare all'interno del gioco (diverso per ogni utente) e una password. Infine scatenando un evento javascript premendo il bottone apposito per la registrazione si inviano le credenziali al server che registrerà sul database le credenziali. Se positivo aggiungerà l'utente ad un dizionario chiave valore (username,socket) per tenere traccia degli utenti online e risponderà al client fornendo l'esito, l'username, la lista di utenti online e la classifica.

A dark-themed sign-up form for 'Tic Tac Toe'. At the top, there are two buttons: 'Sign Up' (highlighted in green) and 'Log In' (greyed out). Below these is the title 'Sign Up Tic Tac Toe'. The form contains two input fields: 'Username\*' and 'Set a Password\*'. At the bottom is a large green button labeled 'GET STARTED'.

- **Login** : lato client consiste in un'interfaccia grafica in cui è possibile inserire il proprio username e password. Infine scatenando un evento javascript premendo il bottone apposito per la login si inviano le credenziali al server che verifica le credenziali tramite database. Se positivo aggiungerà l'utente ad un dizionario chiave valore (username,socket) per tenere traccia degli utenti online e risponderà al client fornendo l'esito, l'username, la lista di utenti online e la classifica.

A dark-themed login form titled 'Welcome Back!'. At the top, there are two buttons: 'Sign Up' (greyed out) and 'Log In' (highlighted in green). Below the title are two input fields: 'Username\*' and 'Password\*'. At the bottom is a large green button labeled 'LOG IN'.

### Lobby

Se l'autenticazione avviene con successo lato client tramite javascript si nasconde il div riguardante l'autenticazione e si mostra il div della lobby.

In particolare la lobby è composta da:

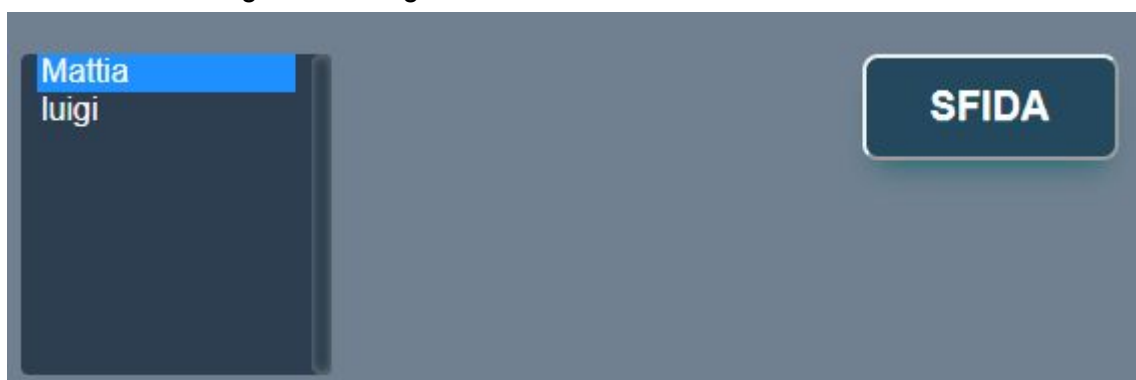
- **Classifica:** fornita dal server successivamente all'autenticazione che mostra il numero delle partite vinte, pareggiate e perse dagli utenti.

| Rank | Name    | Win | Draw | Lose |
|------|---------|-----|------|------|
| 1    | Mattia  | 7   | 8    | 8    |
| 2    | martina | 1   | 1    | 0    |
| 3    | a       | 1   | 0    | 0    |
| 4    | luigi   | 1   | 0    | 0    |
| 5    | r       | 0   | 0    | 1    |

- **Utenti Online :** fornita dal server successivamente ad ogni autenticazione o disconnessione ai client in modo tale avere sempre la lista aggiornata ad ogni autenticazione o disconnessione.



- **Sfida :** dopo aver selezionato l'utente da sfidare all'interno della lista utenti online, viene inviata al server una richiesta di sfida contenente il proprio username e l'username dell'utente da sfidare. Il server toglie dalla lista utenti online i due utenti e invia la richiesta di sfida all'utente sfidato così da poter scegliere se accettare o rifiutare. In caso di rifiuto entrambi vengono rimessi online e l'utente che ha lanciato la notifica viene avvisato del rifiuto. In caso di accettazione il server inserisce entrambi in una "room" utilizzata per poter inviare dati ad entrambi definendo la specifica room. Ogni room ha un nome diverso dalle altre generato sul server da una funzione "getNameRoom()" e ogni nome viene salvato all'interno di una lista. Infine si invia tramite socket un evento "initalize" con cui i client nascondono il div della lobby e mostrano il div riguardante il gioco.



## Gioco

Dopo aver accettato la richiesta di sfida, entrambi i giocatori possono iniziare a giocare e inizia per primo colui che ha lanciato la sfida ed avrà come simbolo la "X". In particolare ogni utente può effettuare una mossa per turno che viene comunicata al server, scatenando un evento cliccando sulla casella del gioco; il server tramite "room" invia la mossa effettuata ad entrambi per poter aggiornare la grafica. Tramite javascript si aggiorna in modo dinamico la grafica in base agli eventi scatenati e si controlla la fine della partita in caso di tris o di pareggio. Alla fine della partita l'utente che ha il turno comunica l'esito della partita al server ed entrambi ritornano alla lobby. Il server riceve rimette gli utenti online e aggiorna la classifica in base al risultato della partita.

localhost:3000 dice

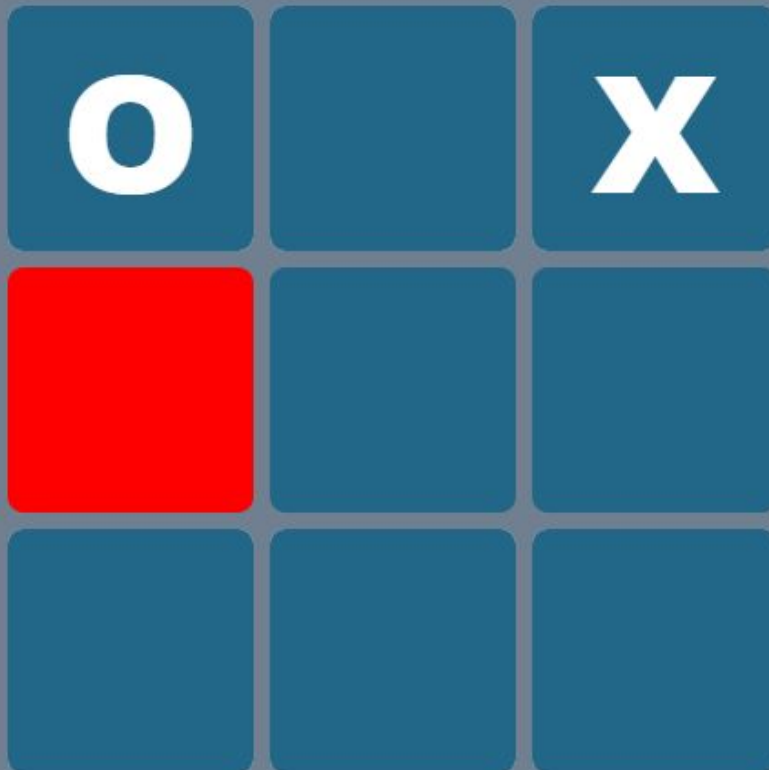
TI SFIDA : Mattia, ACCETTI ?

OK

Annulla

# Tic Tac Toe

Avversario : Mattia



## Database

Con la libreria mysql il server si collega al database instaurando una connessione appena viene avviato. Il database è hostato su sito di hosting gratuito chiamato "alwaysdata.com" utilizzato per molti altri progetti che includevano l'utilizzo di un database. All'interno del database sono presenti due tabelle una chiamata "UTENTE" contenente le credenziali degli utenti e una chiamata "SCORE" contenente il punteggio di ogni utente.



## Criticità

- Tutte le criticità dell'applicazione sono dovute a disconnessione del client non previste in fase di accettazione della richiesta di sfida e in fase di gioco.
- Inoltre se si utilizza il gioco sulla stessa macchina su chrome, al momento della conferma della sfida c'è la necessità di avere entrambe le finestre attive altrimenti il "confirm" di javascript non viene visualizzato perciò bisogna usarlo su due macchine differenti.

## Istruzioni d'uso

**Premessa:** i client che si connettono al server devono essere su macchine differenti in quanto il browser chrome fa partire i messaggi di popup come "confirm" solo se la finestra è attiva. Quindi se si è sulla stessa macchina non è permesso e si ha il rifiuto della sfida per giocare.

Qualora si voglia usare i client sulla stessa macchina bisogna commentare sul file

**handler.js** da riga 130 a 136 per poi decommentare riga 137 e mettere "esito=true" in modo tale che la richiesta di sfida sia sempre accettata.

- Se i client si connettono dalla stessa macchina in cui è hostato il server non effettuare modifiche altrimenti Settare ip statico a riga 2 del file **handler.js**
- Avviare server file app.js con comando "node app.js"
- Connettersi al server tramite browser tramite ip statico assegnato a handler.js
- Effettuare una registrazione inserendo un nome ed una password.
- Nella lobby se sono presenti altri utenti oltre a quello corrente è possibile selezionarlo e sfidarlo premendo il bottone sfida.
- Accettare richiesta di sfida sull'altro utente
- Nel gioco inizia l'utente che ha inviato la richiesta di sfida con il simbolo "X"

- Eseguire mossa premendo sulla casella in cui si vuole mettere il proprio simbolo, un turno per volta, con l'obiettivo di effettuare un tris.
- Alla fine della partita si ritorna alla lobby