Progetto di Ingegneria Informatica 2022/2023

Simple Virtual Tabletop Environment

Simone Ponginibbio
Matricola: 933724 Codice Persona: 10699608

Tutor: prof. **Giovanni Agosta**

1. Avvio programma 2

2. Specifiche ….....….....4

2.1 Obiettivo ….....….....4

2.2 Specifica ….....4

3. Implementazione ….....….....6

3.1 Connessione del primo client ….....….....6

3.2 Creazione di una nuova partita ….....….....6

3.3 Ingresso in una partita già creata ….....….....7

3.4 Uscita da una partita ….....7

3.5 Modifica dell'icona del profilo ...….....….....8

3.6 Modifica del tipo di griglia ….....8

3.7 Invio di messaggi via chat ….....9

3.8 Modifica dell'immagine del campo da gioco ….....9

3.9 Aggiunta di un nuovo token ….....….....10

3.10 Posizionamento di un token sul campo da gioco ….....10

3.11 Rimozione di un token dal campo da gioco ….....11

3.12 Cancellazione manuale di una partita tramite LocalStorage ….....….....11

3.13 Ricaricamento della pagina mentre si è in una partita ….....12

4. Implementazioni future ….....….....12

1. Avvio programma:

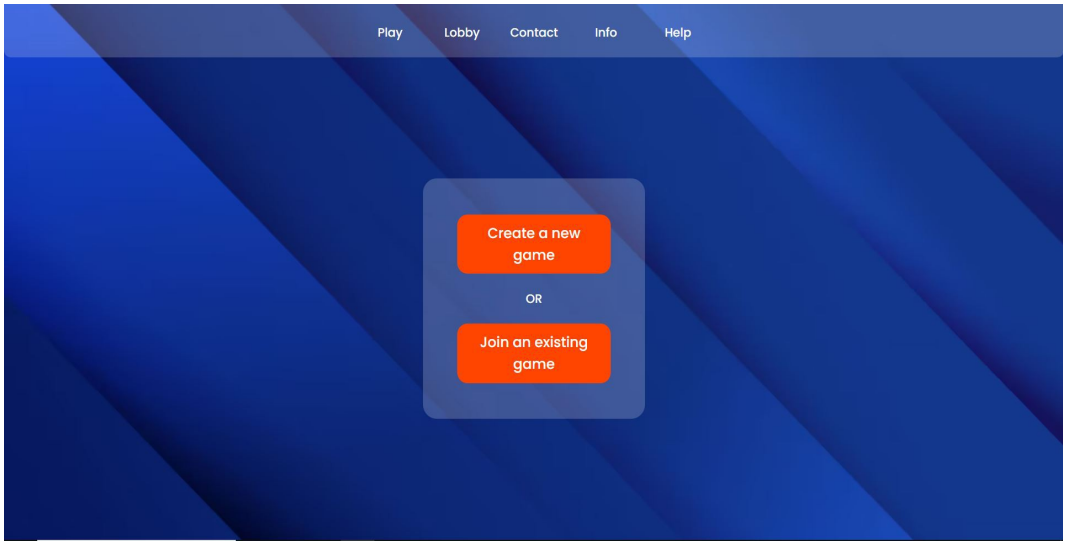
- Per scaricare e eseguire il codice del progetto usare la pagina GitHub: https://github.com/simonepongi/Simple_Virtual_Tabletop_Environment

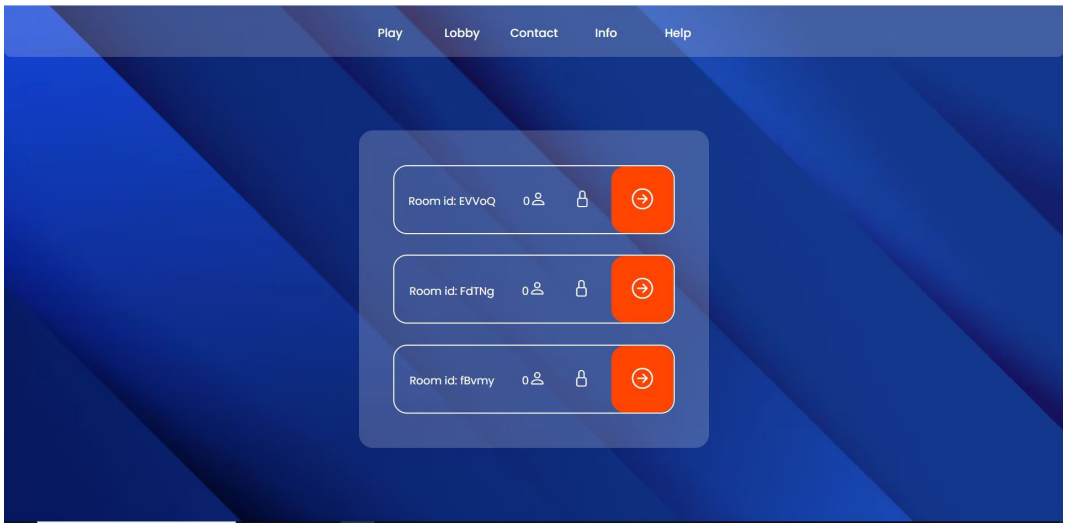
- Per poter utilizzare il programma occorre installare Visual Studio Code e Node.js. All'interno di Visual Studio Code bisogna inoltre installare l'estensione "Live Server" dall'apposita icona sul menu a sinistra. Per importare il progetto fare click su File, Open Folder e selezionare la cartella del progetto.
  Il server verrà avviato sulla porta 3000 di default ma è possibile modificare la porta andando a riga 2 del file "index.js" nella cartella "server", in caso di impossibilità di usare la porta predefinita.

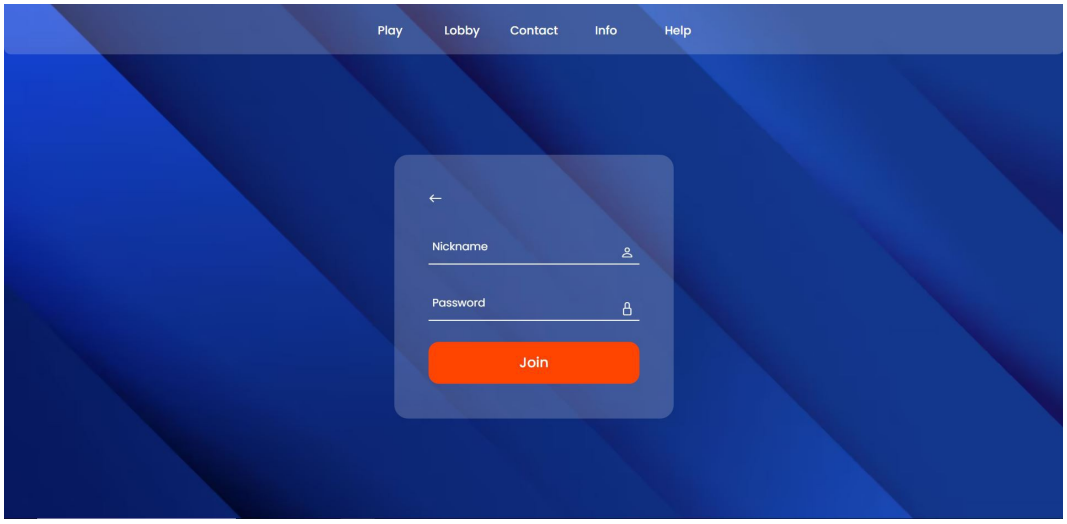
- Una volta scelta la cartella del progetto seguire le istruzioni del seguente video per avviare il server e i client: https://drive.google.com/uc?id=116WR8TshLBorOHA2PED_IXAVemIE67sc

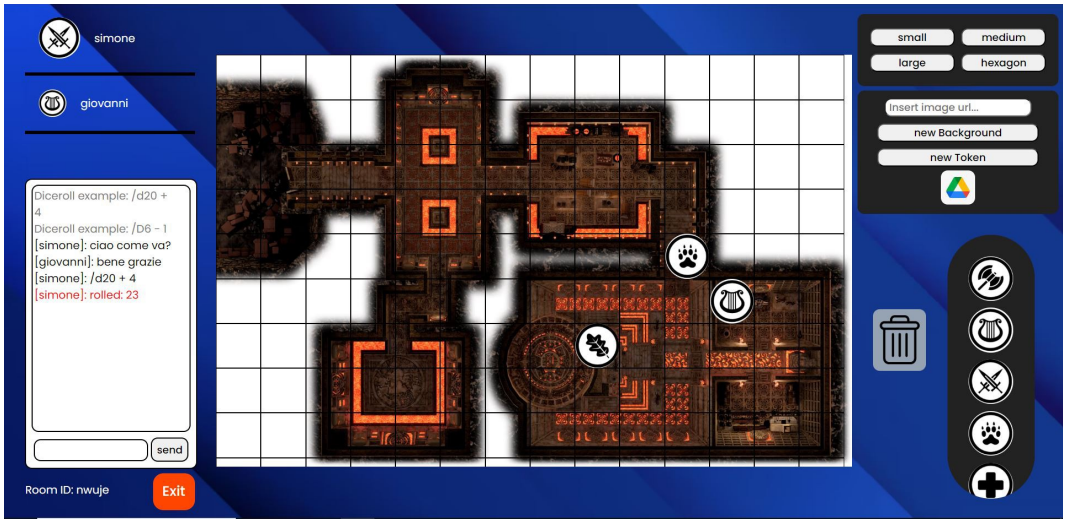
- Per un esempio di partita tra 2 client consultare il seguente video: https://drive.google.com/uc?id=16g6t8WR1fh3eOmsbyNMVx4FjN9j7zL6h

Ecco alcune schermate che si inontrano avviando l'applicativo:




Play
Lobby
Contact
Info
Help
Room id: EVVoQ 0
Room id: FdTNg 0
Room id: fBvmy 0


Play
Lobby
Contact
Info
Help
Nickname
Password
Join


simone
giovanni
Diceroll example: /d20 + 4
Diceroll example: /D6 - 1
[simone]: ciao come va?
[giovanni]: bene grazie
[simone]: /d20 + 4
[simone]: rolled: 23
send
Room ID: nwuje
Exit
small
medium
large
hexagon
Insert image url...
new Background
new Token

2. Specifiche:

2.1 Obiettivo:

L’obiettivo del progetto è lo sviluppo di una piattaforma open source di Virtual Tabletop (VTT) basata su tecnologie moderne come JavaScript, all’interno del browser.
Il modello di riferimento per le funzionalità è owlbear.rodeo, quindi si intende realizzare le funzionalità relative al vero e proprio tavolo di gioco (mappa condivisa con griglia, token, e possibilità di caricare/importare set di token e di mappe personali), ma non funzionalità specifiche di un particolare gioco e neanche la gestione della connessione audio/video.

2.2 Specifica:

Il software è realizzato in HTML5, CSS3 e JavaScript per quanto riguarda la parte di client side. Per la server side è stato utilizzato node.js e la libreria ws (WebSocket) per la comunicazione tra client e server.
Il progetto è stato svolto individualmente seguendo le linee giuda proposte nella descrizione sul sito del “progetto di ingegneria informatica” e prendendo come punto di partenza il progetto di Alessandro Curà svolto l'anno accademico scorso.
Le funzionalità già presenti nel progetto di Alessandro (come la possibilità di tirare dadi, di mandare messaggi via chat, di personalizzare i token, l'immagine di sfondo e la griglia di gioco) sono state mantenute e rivisitate per essere compatibili con le nuove aggiunte.
Questo progetto realizza i miglioramenti suggeriti da Alessandro che riguardano l'aggiunta di token personali (che possono essere spostati solo dal giocatore che li ha posizionati sul campo da gioco), il miglioramento delle grafiche generali della pagina, l'implementazione di un sistema per la gestione e il salvataggio dei dati delle partite.
Quest'ultima funzionalità, come da specifica, doveva essere creato in locale. Per questo motivo si è fatto uso di LocalStorage, un particolare oggetto introdotto in HTML5 che permette di salvare informazioni localmente e di renderle resilienti alla chiusura del browser o allo spegnimento del computer.
I dati presenti nel LocalStorage vengono trasmessi al server in corrispondenza della prima esecuzione di un client e aggiornati sia localmente che sul server ad ogni modifica delle informazioni sulle partite.
Il LocalStorage utilizzato per il progetto è strutturato come una mappa avente un room id (univoco) come key e un oggetto contenente i dati della partita come value. Non vengono invece memorizzati in locale le informazioni riguardanti i client non presenti in alcuna partita.

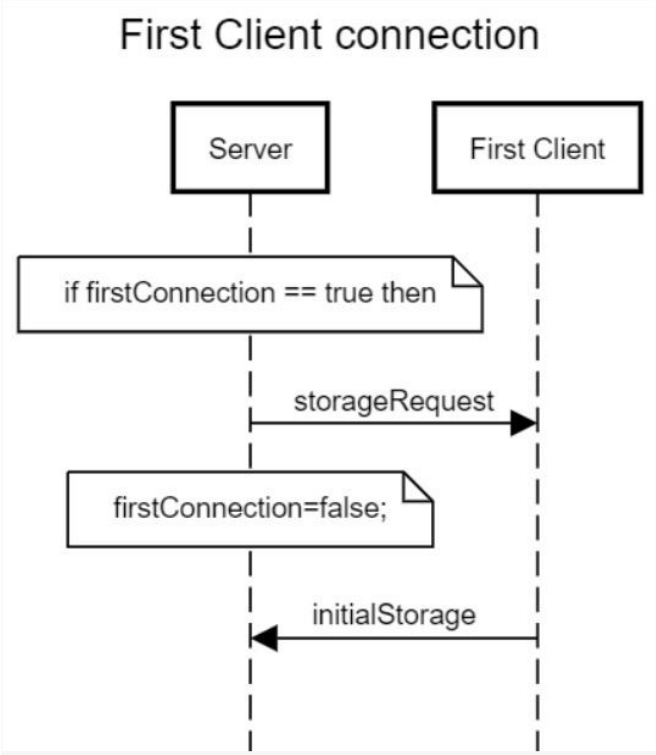
Nei dati della partita vengono memorizzati i seguenti campi:
password, background, size, players, chat, positions e tokens.

- **password**: contiene il valore della password di quella partita, eventualmente anche nullo se la stanza non la richiede;
- **background**: contiene il percorso/url dell'immagine del campo da gioco;
- **size**: contiene la dimensione della griglia di gioco (none, S, M, L, o H);
- **players**: contiene la lista dei giocatori presenti in quella partita, in particolare memorizza il nickname e il valore dell'icona del profilo;
- **chat**: contiene la lista dei messaggi scambiati tra i giocatori in chat, in particolare memorizza il nickname del mittente e l'effettivo testo scambiato;
- **positions**: contiene la lista dei token presenti sul campo da gioco, in particolare memorizza il nickname del proprietario del token, il valore del token e le coordinate del token rispetto al campo da gioco;
- **tokens**: contiene la lista dei percorsi/url dei token utilizzabili nella partita sia come icone di profili che come token da posizionare sul campo da gioco.

Oltre alle funzionalità proposte come miglioramento nel progetto precedente, è stata aggiunta la possibilità di creare stanze senza la password, di visualizzare una lobby con tutte le partite attualmente in memoria, di poter uscire da una partita notificandolo agli altri partecipanti della partita.
Inoltre è stato gestito il reload della pagina da parte di un giocatore e l'eliminazione di una partita in maniera manuale attraverso LocalStorage. Sono state aggiunte piccole modifiche grafiche e stilistiche come per esempio gli alert per l'uscita del giocatore dalla partita e per la rimozione manuale della partita, realizzate con sweetalert2.

3. Implementazione:

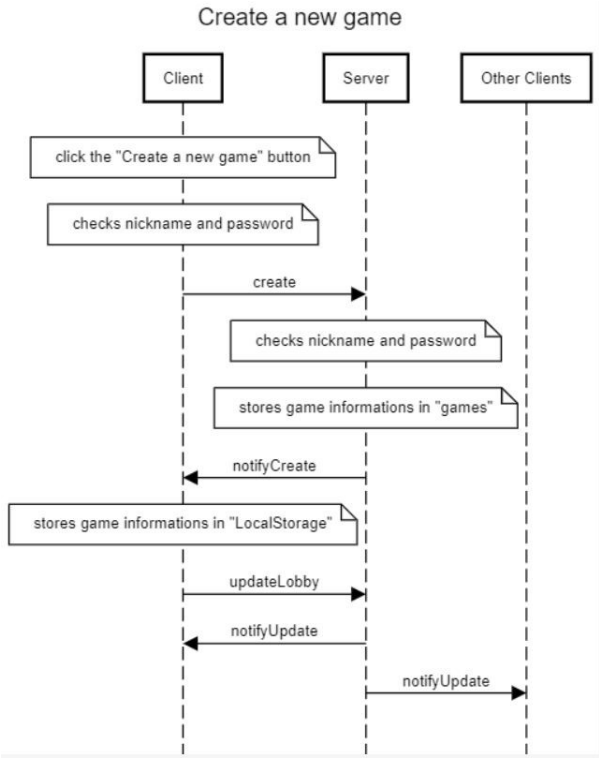
3.1 Connessione del primo client:



- **storageRequest**: contiene la stringa “storageRequest”;

- **initialStorage**: contiene l'array [“initialStorage”, *il contenuto del LocalStorage*]. *(il contenuto del LocalStorage viene inserito in una mappa e convertito in una stringa tramite JSON.stringify per consentirne la serializzazione)*

3.2 Creazione di una nuova partita:



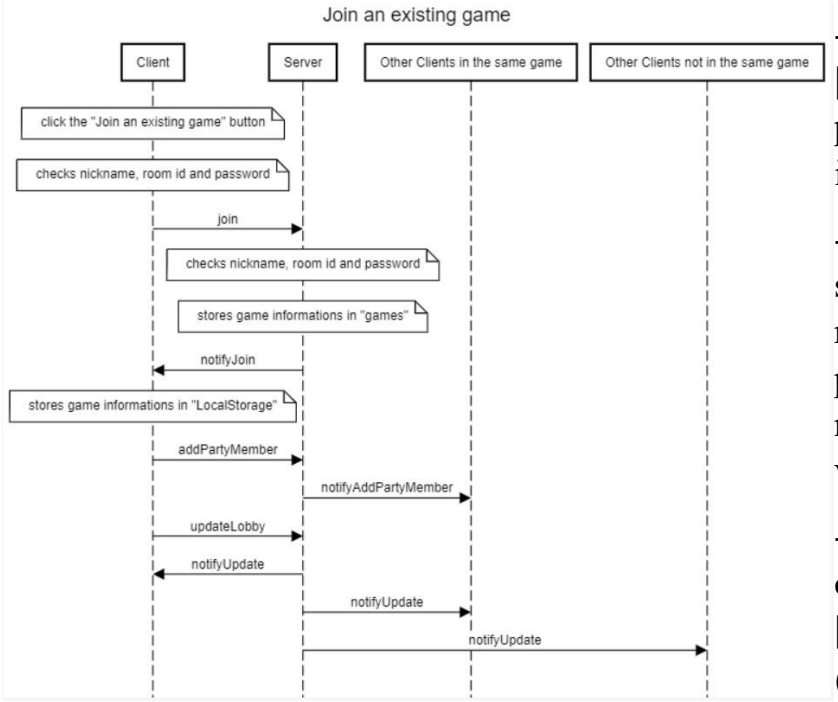
- **create**: contiene l'array [“create”, *la password*, *se si vuole utilizzare la password(true/false)*, *il nickname del giocatore che vuole creare la partita*];

- **notifyCreate**: contiene la stringa “notifyCreate,” + *la room id* + “,” + *la password* + “,” + *il nickname del giocatore che vuole creare la partita*;

- **updateLobby**: contine l'array [“updateLobby”]; *(consente di mandare a tutti i clienti il messaggio di notifyUpdate)*

- **notifyUpdate**: contiene la stringa “notifyUpdate”. *(consente al client di aggiornare in locale la lobby appena modificata da un evento)*

3.3 Ingresso in una partita già creata:



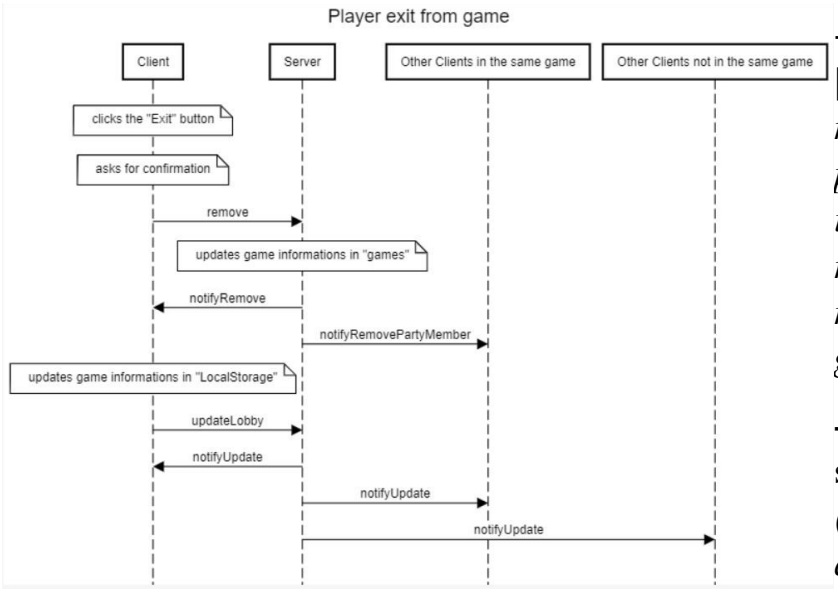
- **join**: contiene l'array [“join”, *il room id*, *la password*, *il nickname con il quale si vuole entrare*];

- **notifyJoin**: contiene la stringa “notifyJoin,” + *la room id* + “,” + *la password* + “,” + *il nickname con il quale si vuole entrare*;

- **addPartyMember**: contiene l'array [“addPartyMember”]; *(consente di mandare a tutti gli altri client presenti nella partita interessata il messaggio di notifyAddMember)*

- **notifyAddPartyMember**: contiene la stringa “notifyAddPartyMember,” + *il nickname del nuovo giocatore entrato*. *(consente al client di aggiornare in locale l'elenco dei giocatori presenti nella partita con le caratteristiche del giocatore aggiunto)*

3.4 Uscita da una partita:

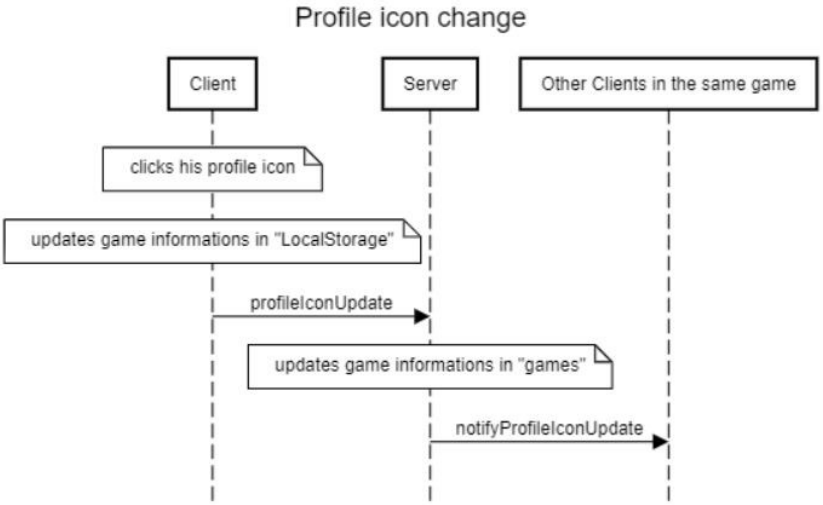


- **remove**: contiene l'array [“remove”]; *(consente di mandare a tutti i client presenti nella partita interessata il messaggio di notifyRemovePartyMember o notifyRemove se si tratta del giocatore uscente)*

- **notifyRemove**: contiene la stringa “notifyRemove”; *(consente al client di aggiornare la sua schermata e tornare nella home page)*

- **notifyRemovePartyMember**: contiene la stringa “notifyRemovePartyMember,” + *il nickname del giocatore uscito dalla partita*. *(consente ai client della stessa stanza del giocatore uscito di aggiornare in locale l'elenco dei giocatori presenti)*

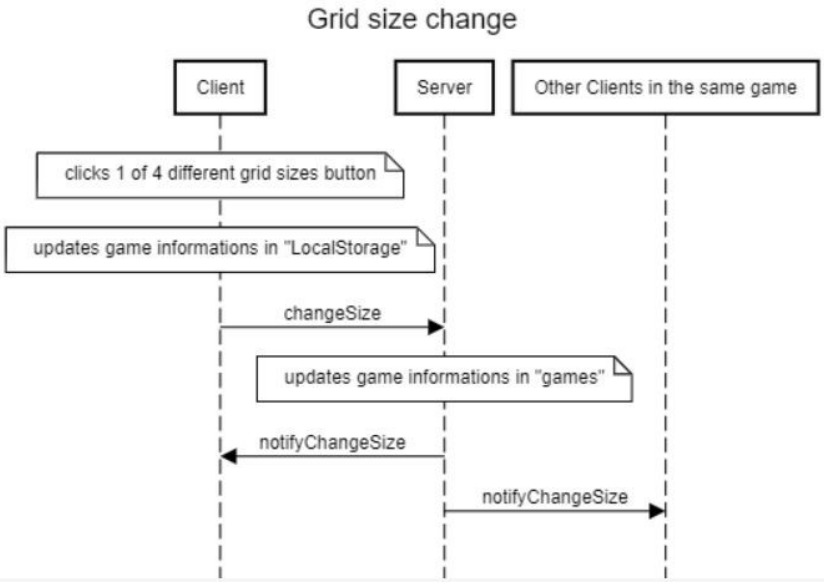
3.5 Modifica dell'icona del profilo:



- **profileIconUpdate**: contiene l'array [“profileIconUpdate”, *il valore della nuova icona del profilo del giocatore*]; *(consente di mandare a tutti gli altri client presenti nella partita interessata il messaggio di notifyProfileIconUpdate)*

- **notifyProfileIconUpdate**: contiene la stringa “notifyProfileIconUpdate,” + *il nickname del giocatore che ha cambiato l'iconadel profilo* + “,” + *il valore della nuova icona del profilo del giocatore*. *(consente ai client della stessa stanza del giocatore che ha cambiato icona del profilo di aggiornare in locale questa caratteristica nell'elenco dei giocatori della partita)*

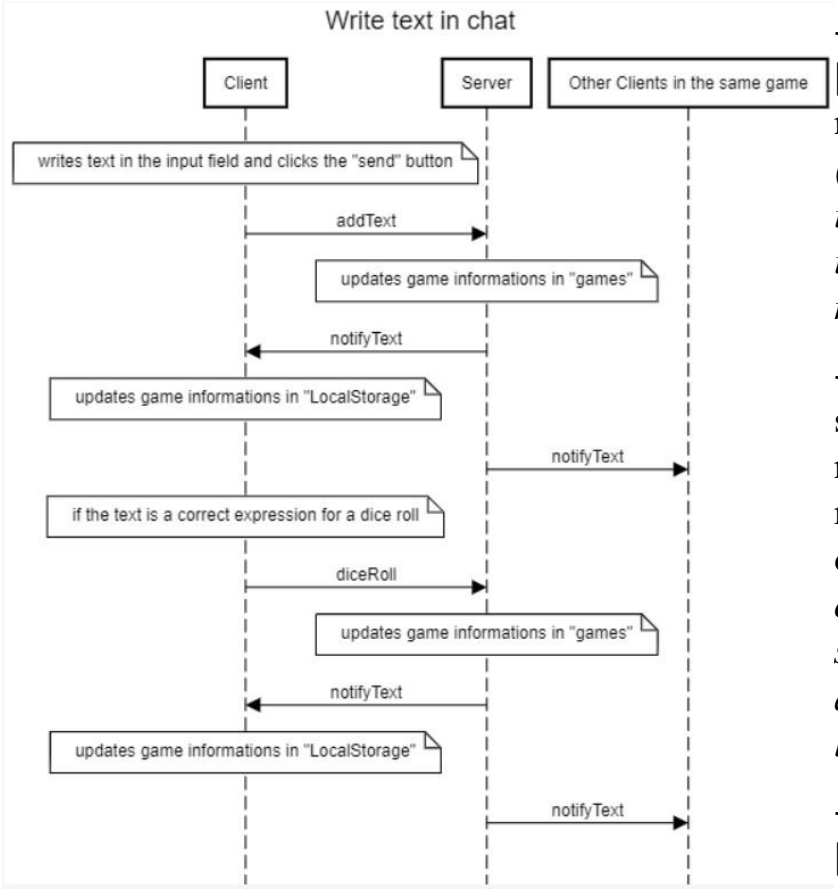
3.6 Modifica del tipo di griglia:



- **changeSize**: contiene l'array [“changeSize”, *valore della nuova griglia (S, M, L o H)*]; *(consente di mandare a tutti i client presenti nella partita interessata il messaggio di notifyChangeSize)*

- **notifyChangeSize**: contiene la stringa “notifyChangeSize”. *(consente ai client della stanza dove è avvenuto la modifica di cambiare in locale il tipo di griglia del campo da gioco)*

3.7 Invio di messaggi via chat:

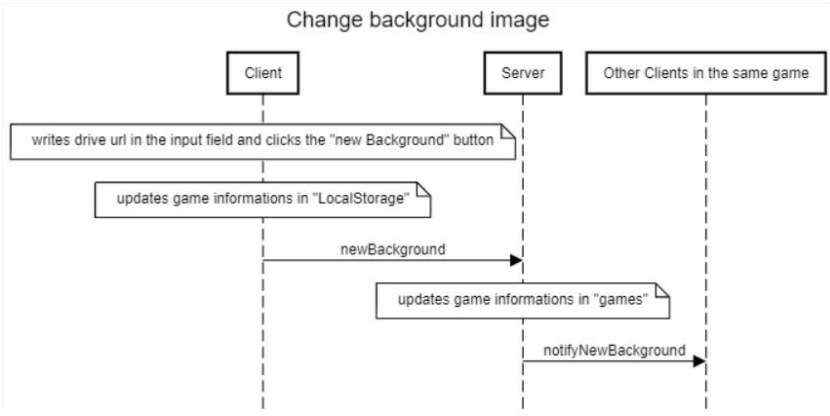


- **addText**: contiene l'array [“addText”, *il testo del mesaggio da inviare*]; *(consente di mandare a tutti i client presenti nella partita interessata il messaggio notifyText)*

- **notifyText**: contiente la stringa “notifyText,” + *il nickname del mittente del messaggio* + “,” + *il testo del messaggio*. *(consente ai client della stanza dove è stato inviato il messaggio di aggiungerlo alla chat in locale)*

- **diceRoll**: contiene l'array [“diceRoll”, *la formula per tirare un dado*]; *(consente di mandare a tutti i client presenti nella partita interessata il messaggio notifyText con il valore del risultato del tiro del dado)*

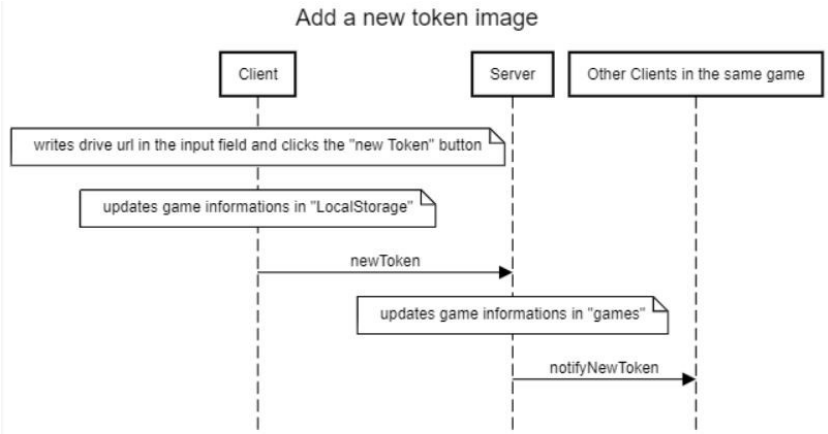
3.8 Modifica dell'immagine del campo da gioco:



- **newBackground**: contiene l'array [“newBackground”, *l'url di google drive dell'immagine di sfondo*]; *(consente di mandare a tutti gli altri client presenti nella partita interessata il messaggio notifyNewBackground)*

- **notifyNewBackground**: contiene la stringa “notifyNewBackground,” + *l'url di google drive dell'immagine di sfondo*. *(consente ai client della stanza dove è stato cambiato l'immagine di sfondo di aggiornarla in locale)*

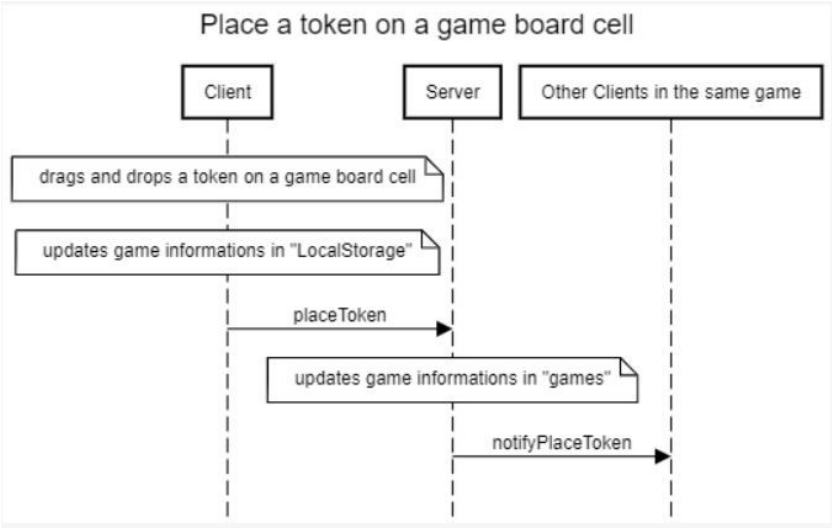
3.9 Aggiunta di un nuovo token:



- **newToken**: contiene l'array [“newToken”, *l'url di google drive dell'immagine del token*]; *(consente di mandare a tutti gli altri client presenti nella partita interessata il messaggio notifyNewToken)*

- **notifyNewToken**: contiene la stringa “notifyNewToken,” + *l'url di google drive dell'immagine del token*. *(consente ai client della stanza dove è stato aggiunto un nuovo token di aggiornarne l'elenco in locale)*

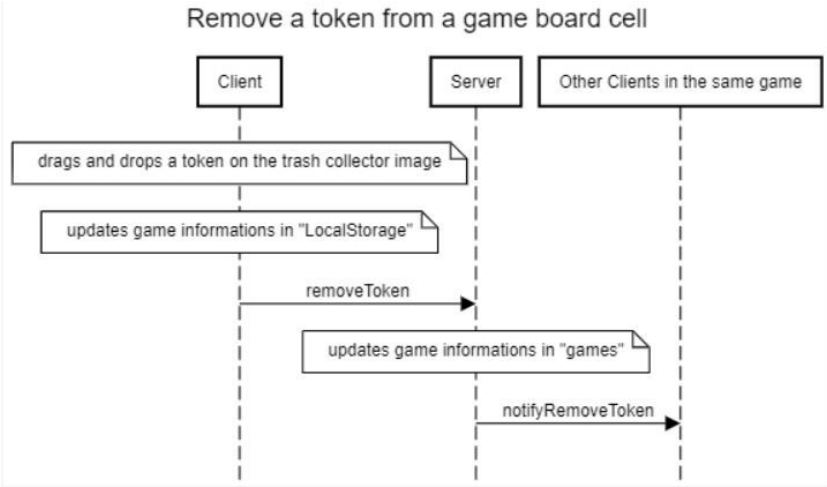
3.10 Posizionamento di un token sul campo da gioco:



- **placeToken**: contiene l'array [“placeToken”, *il valore dell'immagine del token*, *le coordinate nella quale si vuole inserire il token*, *le coordinate nella quale era situato il token (possono essere nulle se il token è nuovo e quindi non viene spostato)*]; *(consente di mandare a tutti gli altri client presenti nella partita interessata il messaggio notifyPlaceToken)*

- **notifyPlaceToken**: contiene la stringa “notifyPlaceToken,” + *il nickname del giocatore che sposta il token* + “,” + *il valore dell'immagine del token* + “,” + *la coordinata x della nuova posizione del token* + “,” + *la coordinata y della nuova posizione del token* se il token era già presente sul campo da gioco allora viene aggiunta anche la stringa “,” + *la coordinata x della vecchia posizione del token* + “,” + *la coordinata y della vecchia posizione del token*. *(consente ai client della stanza dove è stato posizionato un token di aggiungerlo al campo da gioco in locale e di rimuoverlo nella posizione vecchia qualora fosse stato spostato)*

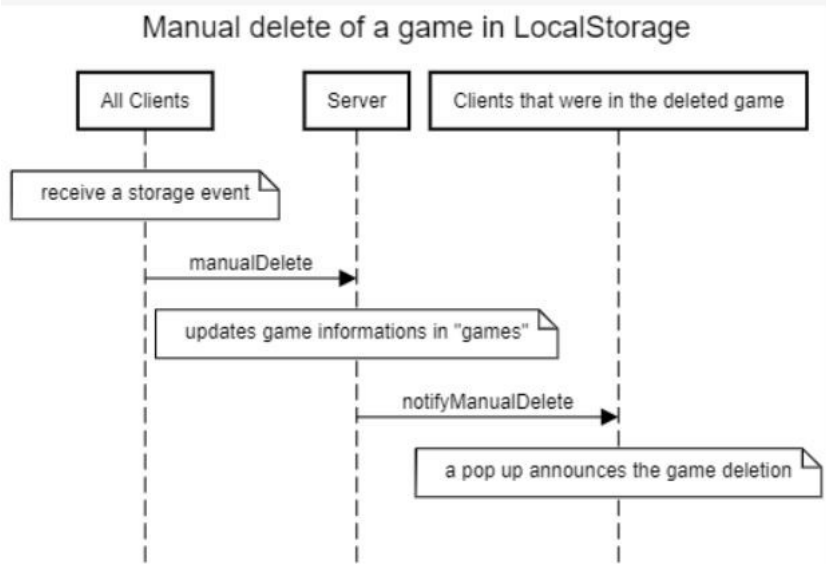
3.11 Rimozione di un token dal campo da gioco:



- **removeToken**: contiene l'array [“removeToken”, *le coordinate del token da rimuovere*]; *(consente di mandare a tutti gli altri client presenti nella partita interessata il messaggio notifyRemoveToken)*

- **notifyRemoveToken**: contiene la stringa “notifyRemoveToken,” + *la coordinata x della posizione del token* + “;” + *la coordinata y della posizione del token*. *(consente ai client della stanza dove è stato rimosso un token di rimuoverlo dal campo da gioco in locale)*

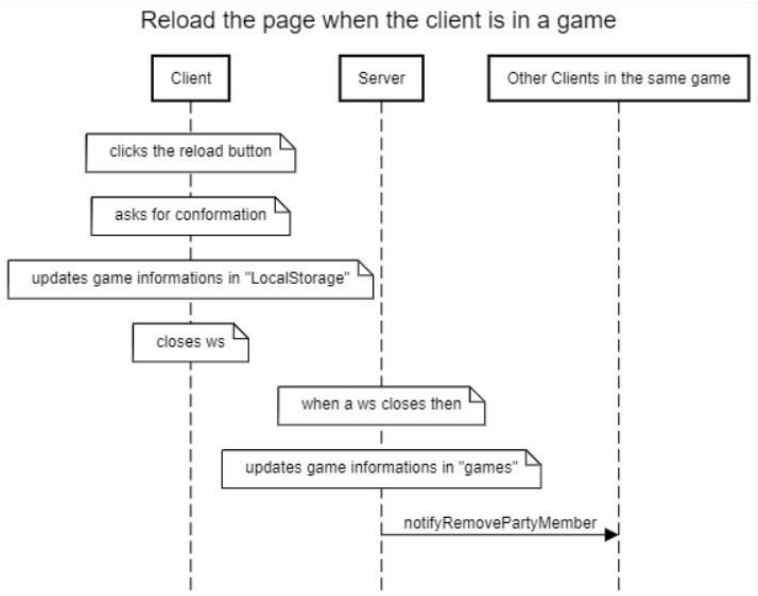
3.12 Cancellazione manuale di una partita tramite LocalStorage:



- **manualDelete**: contiene l'array [“manualDelete”, *la room id della partita eliminata*]; *(consente di mandare a tutti i client presenti nella partita interessata il messaggio notifyManualDelete)*

- **notifyManualDelete**: contiene la stringa “notifyManualDelete”. *(consente ai client della stanza rimossa di ricevere l'informazione tramite alert ed essere rimandati alla home page)*

3.13 Ricaricamento della pagina mentre si è in una partita:



- **notifyRemovePartyMember**: contiene la stringa “notifyRemovePartyMember,” + *il nickname del client che ha effettuato il ricaricamento della pagina*. *(consente ai client della stanza dove il client ha ricaricato la pagina di aggiornare in locale l'elenco dei giocatori presenti nella partita)*

4. Implementazioni future:

Il LocalStorage è una modalità di memorizzazione semplice e facilmente accessibile ma ha il difetto di essere pubblica e modificabile da tutti i client connessi. Un'implementazione futura può essere l'utilizzo di un file esterno o di un database per la memorizzazione delle partite e delle credenziali dei giocatori così da rendere più privati i dati memorizzati.
Altre possibili implementazioni riguardano le grafiche che possono essere rese più adatte all'utilizzo con smartphone/tablet e a nuove funzionalità del gioco stesso.