Report finale del progetto del corso di
Sistemi Distribuiti

Straccia Camicia

Galassi Meshua

Indice

1 Introduzione

Per il progetto del corso di Sistemi Distribuiti, ho pensato di realizzare un applicazione che rappresenti il noto gioco di carte Straccia camicia.
Si tratta di un gioco di coppia, eventualmente espandibile in termini di numero di giocatori, che prevede l'uso di un mazzo di quaranta carte.
Un utente, per poter giocare deve, innanzitutto, registrare un proprio alias con cui verrà identificato. Dopodichè, potrà richiedere di creare una nuova lobby o di accedere ad una già esistente.
Nel caso in cui richieda di creare una nuova lobby, quest'ultima verrà istanziata nella versione privata per cui solamente un altro giocatore a conoscenza del codice identificativo della specifica lobby, potrà accedervi.
Nel caso in cui, invece, scelga di unirsi ad una lobby già esistente si distinguono due casi:

- l'utente conosce il codice di una lobby specifica e vuole accedere ad essa. Si tratta quindi di una lobby privata creata intenzionalmente da un altro giocatore. Attraverso il codice, il secondo giocatore verrà unito alla lobby;

- l'utente non conosce il codice di una lobby ed intende unirsi ad una qualsiasi lobby disponibile. In questo caso, è prevista un'ulteriore distinzione: se esiste una lobby pubblica già creata ma che ancora non ha raggiunto il numero totale di giocatori previsto, il giocatore attuale verrà unito a quest'ultima. Se invece non fosse presente una lobby esistente ma incompleta in termini di numero di giocatori presenti, ne verrà creata una, in maniera trasparente all'utente e si resterà in attesa dell'arrivo di un secondo giocatore.

Quando la lobby avrà raggiunto il numero di partecipanti richiesto, verrà avviata la partita.
All'interno della singola lobby, non ci sarà differenza tra i giocatori esistenti, in termini di ruoli. Ogni giocatore avrà un proprio mazzo di carte che manterrà ordinato e coperto.
La partita si svolge a turni. Ogni giocatore, durante il proprio turno può trovarsi in due situazioni:

- la carta giocata dal giocatore precedente è una carta qualsiasi che va dal quattro al Re di qualsiasi seme. In questo caso il giocatore attuale non deve

rispondere in maniera particolare ma, gioca semplicemente la carta che si trova in cima al proprio mazzo.

- la carta giocata dal giocatore precedente è una carta qualsiasi compresa tra l'Asso e il tre di un qualsiasi seme. In questo caso, il giocatore attuale deve fornire al giocatore precedente, se possibile, un numero di carte pari al valore della carta. Nel caso in cui, tra le carte giocate dovesse uscire nuovamente una carta con valore compreso tra l'Asso e il tre, sarà il giocatore successivo a dover fornire il numero di carte indicato e così via finché non viene soddisfatta una richiesta. A questo punto, il giocatore la cui richiesta è soddisfatta, si guadagnerà l'intero mazzo al centro del tavolo.

Il gioco termina quando tutti i giocatori della lobby avranno esaurito le carte nel proprio mazzo eccetto uno.

2 Obiettivi e requisiti

Si vuole realizzare un'applicazione che consenta agli utenti di giocare al gioco di carte Straccia camicia.
Ogni utente, si registrerà all'interno del sistema indicando un alias che verrà utilizzato per identificarlo.
Una volta registrato, potrà richiedere di creare una nuova lobby, accedere ad una lobby specifica, oppure accedere ad una lobby casuale alla quale mancano ancora partecipanti.
Al momento della creazione della lobby, verrà creato un identificativo della stessa, che verrà restituito al creatore e ad ogni utente che vi si unisce, in modo tale da consentirgli di comunicare l'identificativo ad altri utenti con cui eventualmente vorrà giocare.
Non è prevista alcuna distinzione tra gli utenti: ogni utente può scegliere di creare una nuova lobby, unirsi ad una lobby specifica oppure unirsi ad una lobby casuale.
Una volta completata la lobby in termini di numero di giocatori, verrà avviata automaticamente la partita. Una volta stabilito l'ordine dei giocatori, verranno distribuite le carte in ugual numero seguendo tale ordine e la partita si svolgerà con la stessa sequenza di giocatori fino al termine.
Ad ogni turno, il giocatore attuale girerà dalla cima del proprio mazzo, tante carte quante ne richiede la carta giocata dal giocatore precedente. Ovvero, se la carta giocata dal giocatore precedente non corrisponde ad un Asso, un Due o un Tre di qualsiasi seme, il giocatore attuale giocherà una sola carta, diversamente, dovrà giocare un numero di carte pari a quello indicato dalla carta del giocatore che lo precede a meno chè una di queste carte non sia a sua volta un Asso, un Due oppure un Tre di qualsiasi seme. A mano a mano che i giocatori esauriscono il proprio mazzo, si creerà una classifica della partita.

2.1 Scenari

Una volta avviata l'applicazione, l'utente troverà il server in attesa di richieste di registrazione di un alias e di creazione o ingresso in una lobby.
Nell'immagine 1 è rappresentato il diagramma dei casi d'uso che mostra le interazioni tra client e server al fine dello svolgimento di una partita.

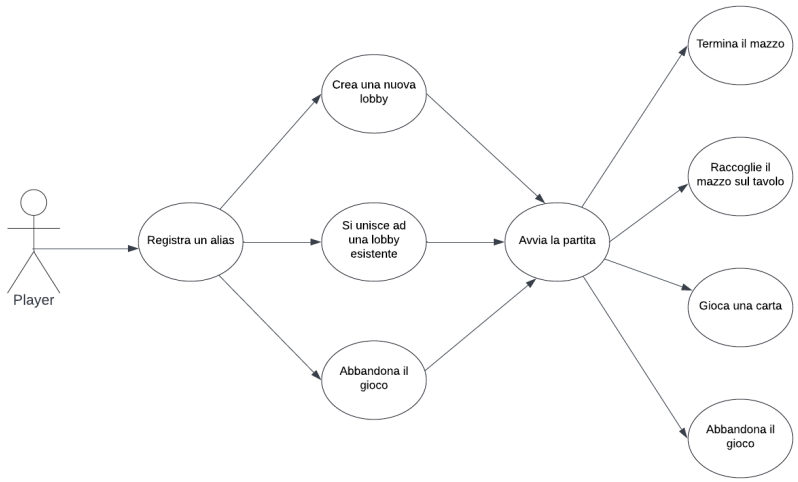

Figura 1: Diagramma dei casi d'uso.

Dato un utente intenzionato a giocare, una volta aperta l'applicazione, dovrà registrare il proprio alias col quale verrà identificato per tutta la sessione.
Una volta registrato, potrà decidere se creare una nuova lobby o accedere ad una lobby esistente. Se sceglie di creare una nuova lobby, quest'ultima verrà creata automaticamente con un massimo di 2 giocatori previsti e di tipologia privata perciò solo chi ha creato la lobby e chi conosce il codice potrà accedervi. Alla creazione della lobby, infatti, verrà restituito al creatore l'identificativo della lobby in modo tale che qualche altro giocatore vi si possa unire.
Nel caso in cui l'utente richieda di accedere ad una lobby esistente, possiamo distinguere due situazioni:

- conosce il codice di una lobby. Questo vuol dire che si tratta di una lobby privata, creata da un proprio conoscente, il quale gli ha comunicato il codice. In questo caso potrà inserire direttamente il codice e verrà unito alla partita dell'amico.

- non conosce il codice di una lobby perciò, l'unica scelta che ha è quella di accedere ad una lobby pubblica casuale. In questo secondo caso, qualora non dovessero essere presenti lobby disponibili, ne verrà creata una nuova di tipologia pubblica e si resterà in attesa di un giocatore che si unisca.

Una volta che tutti i partecipanti previsti si sono uniti alla lobby, si potrà far partire la partita. Verrà distribuito ad ogni giocatore un mazzo con un ugual numero di carte.
I giocatori giocheranno, a turno. A partire dal primo giocatore, ognuno giocherà la carta che si trova in cima al proprio mazzo in base a ciò che è stato giocato dal giocatore precedente:

- il giocatore precedente ha giocato una carta che va dal quattro al Re di un qualsiasi seme del mazzo. In questo caso, il giocatore attuale gioca una sola carta;

- il giocatore precedente ha giocato una carta compresa tra l'Asso e il Tre di qualsiasi seme. Il giocatore attuale deve giocare un numero di carte pari al valore della carta. A questo punto si possono identificare ulteriori tre scenari:

  - il giocatore attuale soddisfa interamente la richiesta del giocatore precedente. Questo vuol dire che il giocatore attuale ha giocato numero di carte pari a quello indicato dalla carta del giocatore precedente, tutte con un valore superiore al tre. Il giocatore precedente guadagnerà l'intero mazzetto presente al centro del tavolo, il quale verrà inserito alla fine del mazzo già in possesso dal giocatore;
  - il giocatore attuale non ha carte sufficienti per soddisfare la richiesta e tra le carte in possesso del giocatore attuale non ci sono carte con un valore compreso tra Asso e Tre. Il giocatore attuale perde la partita e, poichè l'applicazione prevede solo partite da due giocatori, il giocatore precedente vince la partita. In un ottica futura in cui si possono prevedere partite con un numero maggiore di giocatori, il giocatore precedente ottiene il mazzo di carte sul tavolo, indipendentemente dal fatto che il numero di carte giocate dall'attuale giocatore sia pari o inferiore al valore della carta giocata dal giocatore che lo precedeva e ricomincia il giro come giocatore di turno;
  - il giocatore attuale, nel soddisfare la richiesta del giocatore precedente, gioca a sua volta una carta di valore compreso tra Asso e Tre: sarà compito del giocatore successivo soddisfare la nuova richiesta e la richiesta

precedente semplicemente non verrà soddisfatta. Anche in questo caso, appena una delle richieste in catena viene soddisfatta, il giocatore precedente a quello che soddisfa la richiesta, guadagnerà l'intero mazzo creatosi al centro del tavolo e lo accoderà al proprio mazzo.

Indipendentemente dal fatto che un giocatore stia rispondendo alla richiesta di un altro utente o meno, se finisce il proprio mazzo perde la partita.
Un giocatore può scegliere di abbandonare la lobby anche durante una partita, in questo caso la partita verrà interrotta per tutti i giocatori ma, se lo desiderano, i giocatori rimanenti potranno avviare una nuova partita ripetendo la procedura prevista per l'accesso ad una lobby.
Al termine della partita verrà creata una classifica dei giocatori.

3 Analisi dei requisiti

L'obiettivo principale dell'applicazione è quello di gestire una partita a Straccia Camicia. A tale scopo, sono stati individuati alcuni requisiti funzionali e non funzionali.

3.1 Requisiti funzionali

- All'avvio dell'applicazione verrà richiesto di inserire il proprio alias che resterà valido per tutta la sessione.
- Dopo aver indicato l'alias, l'utente potrà decidere a quale lobby accedere:
  - creare una nuova lobby;
  - accedere ad una lobby pubblica scelta in maniera casuale;
  - accedere ad una specifica lobby, indicando il codice identificativo della stessa.
- Al completamento della lobby verrà avviata la partita.
- Il sistema distribuisce ad ogni utente il proprio mazzo.
- La partita è organizzata in turni.
- L'utente, durante il proprio turno potrà giocare la carta in cima al proprio mazzo oppure raccogliere il mazzo dal tavolo qualora fossero presenti le condizioni necessarie, ovvero dopo che il giocatore ha giocato una carta con valore compreso tra Asso e Tre, il giocatore seguente ha giocato un numero di carte pari al valore della carta giocata dal primo tutte con valore superiore a tre.
- L'utente potrà abbandonare la lobby sia prima che durante la partita. In questo secondo caso, la partita verrà interrotta per tutti i giocatori. I giocatori che restano in gioco dovranno ripetere la procedura di accesso alla lobby.

3.2 Requisiti non funzionali

- Il servizio deve poter gestire richieste provenienti da più utenti contemporaneamente.
- Il design deve essere scalabile.
- Il sistema deve essere costruito in modo tale da permettere una facile estensibilità in caso di integrazioni future.
- L'architettura dev'essere di tipo client server in cui il server gestisce interamente la partita mentre il client fornisce solamente le carte giocate dallo specifico giocatore.
- Come sistema di build automation verrà usato Gradle.
- Il server gestirà le richieste in maniera sequenziale.

3.3 Analisi e modello del dominio

Le entità di base che si intende sivluppare sono:

- il client che rappresenta lo specifico utente e che invierà messaggi al server relativi all'avvio, alla terminazione della partita ed alla gestione della stessa.
- il server che esporrà il servizio ai client. Implementa le diverse operazioni necessarie per gestire la partita.

4 Design

Alla luce dei requisiti funzionali e non funzionali individuati in fase di analisi, è stata definita la struttura delle diverse entità e le loro relazioni. Il sistema è composto principalmente da due entità fondamentali, i client ed il server, che interagiscono tra loro.

4.0.1 Server

Il server è strutturato a partire da due elementi principali:

- la lobby che rappresenta una stanza in cui sono riuniti tutti i giocatori di una partita;
- la partita che, invece, rappresenta la partita vera e propria.

Il server, infatti, gestisce i vari gruppi di giocatori ed i diversi tavoli relativi alle partite in corso.
Ogni giocatore verrà registrato all'interno del server con un identificativo univoco e, successivamente, avrà la possibilità di scegliere se unirsi ad una lobby o crearne una nuova. Il giocatore verrà unito alla lobby scelta e resterà in attesa che quest'ultima raggiunga un numero di partecipanti pari al numero di giocatori previsti alla sua creazione.
Una volta completata la lobby, ovvero, riempito il tavolo di gioco, il server creerà ed avvierà una nuova partita in maniera automatica. L'ordine di gioco dei giocatori viene scelto dal server in maniera casuale ed in base a quest'ordine, verranno distribuite le carte ai vari giocatori.
A questo punto, il server si occuperà di mantenere memorizzate le carte relative al mazzo sul tavolo e vincolare i giocatori a giocare seguendo l'ordine di gioco attraverso la gestione dei turni che variano in base alle carte giocate.
Al termine della partita, quando in gioco è rimasto un solo giocatore, oppure quando un giocatore decide di abbandonare la partita, il server si occuperà di informare tutti gli utenti rimasti in gioco della fine della partite e, una volta che tutti sono stati informati, la partita verrà eliminata e per accedere ad una nuova partita bisognerà unirsi ad una nuova lobby.

4.0.2 Client

Entità che rappresenta il giocatore e comunica col server attraverso uno scambio di messaggi.
Prima dell'inizio della partita, il client permetterà all'utente di scegliere le caratteristiche del tavolo in cui vuole giocare: se ne vuole creare uno proprio, se vuole accedere ad un tavolo conosciuto oppure se preferisce essere inserito in maniera casuale all'interno di un tavolo già esistente.
Una volta che l'utente ha effettuato queste scelte, il client invierà un messaggio al server per richiedere di accedere alla lobby indicata e, dopo la conferma da parte del server, resterà in attesa di ricevere il proprio mazzo di carte.
A questo punto, ad ogni giro il client si informerà circa il giocatore di turno e, in caso sia il proprio turno giocherà una carta o raccoglierà il mazzo al centro del tavolo, in caso contrario resterà comunque in attesa di un aggiornamento relativo alle carte giocate dagli altri giocatori.
Ogni giocatore può, comunque, in qualsiasi momento, scegliere di abbandonare la lobby ed eventualmente, nel caso sia già stata avviata, anche la partita.
Al termine della partita, verrà informato della posizione in classifica raggiunta dall'utente.

5 Implementazione

5.1 Dettagli implementativi Server

Il server è basato su quattro classi principali che permettono di gestire i client, i giocatori, le lobby e le partite.

5.1.1 PlayerHandler

Corrisponde all'entità che memorizza tutti i giocatori registrati e collegati al server in un dato momento.

5.1.2 LobbiesHandler

Si tratta dell'entità che mantiene e gestisce tutte le lobby esistenti sul server in un dato momento. Corrisponde infatti ad una lista di lobby di cui non è possibile avere più istanze.

5.1.3 ClientsHandler

Si tratta di un thread indipendente che viene creato ed avviato dopo aver ricevuto una richiesta di collegamento da parte di un client e dopo che il server l'ha accettata. Rimane in ascolto sulla socket associata allo specifico client ed attende messaggi da quest'ultimo.
I possibili messaggi sono:

- newPlayer: corrisponde alla richiesta di registrazione dell'alias del nuovo player. Infatti, alla ricezione di questo messaggio ci si attende una seconda stringa che rappresenta l'alias da associare alla socket del client. Quando il server riceve questa richiesta, controlla che la stringa associata non corrisponda ad alcun alias già presente e in caso affermativo, risponderà al client inviando un messaggio di errore che notifica la presenza di un alias uguale, in caso contrario invierà una coferma dell'avvenuta registrazione.

- createLobby: riguarda la richiesta, da parte del client, di creazione di una lobby privata. Il server si occuperà di creare una lobby con le caratteristiche richieste e di inviare una conferma dell'avvenuta creazione associando alla risposta anche il codice della lobby appena creata.

- appendPrivate: è la richiesta che verrà effettuata da parte di giocatori che conoscono l'identificativo di una lobby restituito dopo aver chiesto la creazione di una lobby privata. Infatti, a questo messaggio deve essere associata una stringa relativa al codice della lobby a cui il client si vuole unire.

- appendPublic: è la richiesta che il server riceverà da coloro che desiderano giocare ma che non hanno alcun codice relativo ad una lobby. In un'ottica di miglioramento futuro in cui saranno previste partite con più di due giocatori, questo messaggio prevede di avere associato anche il numero di giocatori con cui si vuole condividere la partita.

- removePlayer: messaggio ricevuto ogni volta che un player intende abbandonare l'attuale sessione.

5.1.4 GameHandler

Consiste in un thread indipendente che viene creato ed avviato dal ClientHandler che ha gestito l'ingerimento di un giocatore che ha rappresentato l'ultimo player entrato in una determinata lobby prima che questa raggingesse il numero di giocatori richiesti.
Rappresenta la partita nel suo complesso e gestisce tutte le interazioni dei client associati ad una determinata partita.
A turno, legge i messaggi ricevuti dai client della specifica lobby ed aggiorna lo stato della partita di conseguenza.
I possibili messaggi sono:

- startGame: messaggio con cui il client notifica al server di essere pronto ad avviare la prtita, a cui il server risponderà inviando il mazzo di carte associato al player.

- requireTurn: informa il client circa il giocatore attualmente di turno.

- updateState: aggiorna il client con l'ultima mossa effettuata da un qualsiasi altro giocatore.

- playCard: gestisce la ricezione di una carta giocata.

- getTableDeck: alla ricezione di questo messaggio, se lo si riceve da parte di un player che ne ha diritto, il server si occupa del trasferimento del mazzo centrale di carte al player che ne ha fatto richiesta.
- deckEmpty: il client informa il server che ha terminato il proprio mazzo e perciò ha perso la partita.
- leaveGame: un giocatore ha deciso di abbandonare la partita. Il server cambierà lo stato della partita indicandola come terminata.

Qualsiasi messaggio di risposta del server alle richieste dei client, oltre alle informazioni richieste dal client, prevede anche l'invio di uno status che sarà significativo per il client per capire se la partita è ancora in corso, se è terminata perchè è rimasto un solo giocatore in gioco, oppure se è terminata perchè un player ha deciso di abbandonarla.

5.2 Dettagli implementativi Client

Il client è caratterizzato da due classi principali che si occupano di inviare i messaggi al server relativi al momento del settaggio delle impostazioni delle partite ed alla gestione della partita stessa.

5.2.1 ClientSetter

Si tratta della classe a cui si fa riferimento per indicare le impostazioni che si desidera per la partità quali: alias del giocatore e caratteristiche della lobby. Definite queste informazioni, questa classe si occupa di inviare al server le richieste relative.

5.2.2 Game

Si tratta della classe con cui ci si interfaccia per la gestione delle operazioni possibili mentre la partita è in corso.
É la classe che invierà al server i messaggi contenenti le richieste delle varie operazioni e processerà le risposte ricevute.

5.3 Self-assessment

Per testare il software prodotto sono stati utilizzati test automatici JUnit e test manuali di utilizzo dell'applicazione.

6 Istruzioni per il deployment

Si tratta di un'applicazione Java, per eseguirla basta una qualsiasi macchina dotata di JVM su cui mandare in esecuzione l'applicazione che non necissita di nessuna installazione o configurazione particolare.
Occorre avviare il server richiamando, dalla directory principale del progetto

```
./gradlew server:run
```

oppure, entrando nella directory del server con:

```
cd server
./gradlew run
```

Il server verrà eseguito in locale e resterà in ascolto di richieste da parte dei client, sulla porta 11000.
Una volta avviato il server, si potranno avviare i diversi client sempre a partire dalla directory principale del progetto eseguendo il comando:

```
./gradlew client:run
```

oppure, entrando nella directory del client eseguendo:

```
cd client
./gradlew run
```

7 Esempi d'uso

L'applicazione è sprovvista di interfaccia grafica, perciò, gli utenti interagiranno col client da riga di comando.
All'avvio dell'applicazione, verranno richieste alcune informazioni necessarie per settare il tavolo e gestire i giocatori presenti.

```
Connection established to host localhost: 11000
Inserisci il tuo alias: Matteo
Puoi premere Q in qualsiasi momento per uscire.
Vuoi entrare in una lobby esistente (E) o crearne una nuova (N)?
E
NHai il codice di una lobby? (Y/N)
N
Ti sei unito alla lobby: L9916
In attesa...
Hai 20 carte.
E' il tuo turno, vuoi giorcare una carta (P), raccogliere il mazzo (R), uscire dal gioco (Q)?
```

Figura 2: Impostazioni iniziali.

Infatti, come mostrato in figura 2, le prime cose richieste riguardano:

- l'alias che verrà assegnato allo specifico giocatore. Nel caso in cui l'alias sia già presente all'interno del server, verrà mostrato un messaggio apposito e verrà richiesto di inserire un nuovo alias;
- una volta impostato l'alias si può scegliere se entrare in una lobby esistente o crearne una nuova;
- se si sceglie di crearne una nuova non occorre procedere con altre scelte;
- se si sceglie di accedere ad una lobby esistente si può indicare di essere in possesso di un codice di una lobby e quindi inserirlo. A quel punto, il giocatore verrà aggiunto alla lobby identificata col codice indicato. Se invece non si possiede un codice, il player verrà inserito in una lobby casuale;
- in qualsiasi momento del gioco, è possibile indicare che si intende abbandonare la sessione.

Ad eccezione dell'alias iniziale, qualsiasi informazione richiesta all'utente deve essere specifiata scrivendo la relativa lettera, indicata all'interno della richiesta, in carattere maiuscolo.

Una volta completata l'impostazione del tavolo di gioco, si resta in attesa che la partita venga avviata, ovvero, che si unisca un secondo giocatore oppure che arrivi il proprio turno di gioco.
Durante la partita, ogni giocatore, al momento del proprio turno potrà scegliere di:

- giocare una carta dal proprio mazzo
- raccogliere il mazzo centrale
- abbandonare la partita.

Un giocatore potrà giocare una carta dal proprio mazzo in ogni caso, ad eccezione del momento in cui, avendo precedentemente giocato una carta del valore compreso tra Asso e Tre, l'avversario ha soddisfatto la sua richiesta. Solo in questo caso può raccogliere il mazzo centrale e, finchè non lo ha raccolto, non può giocare una nuova carta.
Se un utente dovese scegliere di abbandonare la partita dopo che quest'ultima è già stata avviata, la partità verrà interrota, l'utente avversario sarà avvisato e, come mostrato in figura 3 , avrà la possibilità di accedere ad una nuova lobby per cominciare una nuova partita.

```
Un giocatore ha lasciato la lobby, la partita è terminata.
Vuoi cambiare alias? (Y/N)
N
Puoi premere Q in qualsiasi momento per uscire.
Vuoi entrare in una lobby esistente (E) o crearne una nuova (N)?
N
Il codice della lobby creata è: L2141
In attesa...
```

Figura 3: Avviso che un giocatore ha lasciato la partita.

Al termine di una partita, ogni giocatore verrà avvisato della vittoria o della sconfitta.

8 Conclusioni

L'applicazione sviluppata riguarda un gioco di carte da giocare in coppia. Per farlo, ho deciso di strutturarla con un'architettura client-server in cui i client fungevano semplicemente da interfaccia tra utente e server e, di fatto la partita vera e propria restava interamente gestita dal server, ad eccezione del mazzo di carte in possesso dello specifico player.
Il client ed il servrer comunicano attraverso lo scambio di messaggi. Questo, durante lo sviluppo del progetto, mi ha permesso di approfondire questo aspetto di gestione delle interazioni.

8.1 Miglioramenti futuri

Restano comunque alcuni accorgimenti che si potrebbero adottare in un futuro con poche modifiche poichè il sistema in parte è già predisposto a questi miglioramenti:

- incrementare il numero di giocatori ammessi ad un tavolo da due fino a cinque.
- al termine della partita, mantenere la lobby attuale per gli utenti che intendono giocare una nuova partita all'interno della stessa lobby, o comunque permettere agli utenti di ricominciare una nuova partita senza dover ripetere interamente l'iter di registrazione sul server.