GUESS THE WORD

PROGETTO 2022/2023 DI LABORATORIO DI SISTEMI OPERATIVI

Jonathan Borrelli N86004049, Agostino Cesarano N86003587

1 SOMMARIO

2 INTRODUZIONE

Il gioco "Guess the Word" è un'applicazione Android multiplayer, gestita tramite socket, in cui gli utenti possono registrarsi, accedere e giocare insieme dentro le stanze di gioco. Gli utenti, durante la registrazione, possono personalizzare il proprio profilo, scegliendo un'immagine selezionabile tra 16 avatar. Ogni stanza può ospitare tra i 2 e i 10 giocatori e più stanze possono essere attive contemporaneamente. Una volta che una stanza raggiunge un numero sufficiente di giocatori, un utente detto "**chooser**", può scegliere una parola da far indovinare agli altri partecipanti, detti "**guesser**". Gli altri partecipanti possono tentare di indovinare la parola e se non ci riescono entro un certo limite di tempo, il server aggiunge una lettera in più.

3 STANZA DI GIOCO

È stato scelto di strutturare la stanza di gioco come una chat, consentendo in essa ad ogni partecipante di mandare un messaggio agli altri in qualsiasi momento. Durante il gioco, ogni messaggio inviato da un "guesser" viene confrontato con la parola da indovinare e, se corrisponde, il "guesser" vince e il gioco termina.

All'interno della chat esiste anche un ulteriore tipologia di messaggio, il messaggio di notifica, inviato dal sistema a tutti i partecipanti. Esso viene mostrato in occasione di eventi particolari:

- quando un utente entra o esce dalla stanza
- quando un utente vince il gioco (viene mostrato chi ha vinto e quanti punti ha vinto)
- quando viene scelto il chooser (viene mostrato a tutti chi è il chooser scelto dal server)
- quando comincia una partita
- quando viene aggiunta una nuova lettera
- etc....

Quando un nuovo utente entra in una stanza, se quest'ultima è già in gioco, il suo stato viene messo a "**spettatore**". Ciò significa che non può inviare messaggi nella chat né partecipare alla partita in corso. Deve attendere in questo stato fino alla fine della partita attuale, dopodiché potrà partecipare alle partite successive.

Una stanza è considerata in gioco non appena il "chooser" sceglie una parola e gli altri partecipanti cominciano a scrivere i tentativi per indovinarla. Durante la fase in cui il "chooser" sta ancora scegliendo una parola, la stanza non è considerata in gioco e qualsiasi nuovo utente che entra nella stanza diventa subito un "guesser".

4 FUNZIONAMENTO DEL GIOCO

Più nel dettaglio il gioco segue i seguenti step:

1. STANZA IN ATTESA

Se nella stanza si trova un solo giocatore il gioco non comincia finché non si trovano in stanza almeno 2 giocatori. Non appena entra un secondo giocatore, il server aspetta circa 5 secondi, sceglie il giocatore detto chooser e comincia la prossima fase

2. STANZA IN ATTESA CHE IL CHOOSER SCELGA LA PAROLA

Nella stanza si trovano 2 o più giocatori e Il server ha appena scelto un chooser. A questo punto per gli altri guesser non cambia nulla, ancora non è iniziato il gioco per loro, mentre al chooser compare una schermata in cui potrà scegliere tra 10 parole.

Le 10 parole, è stato scelto di generarle tramite un API online che genera n parole casuali tra 4 lingue diverse (inglese, italiano, spagnolo e tedesco). Il sito dell'API è il seguente: https://random-word-api.herokuapp.com. Infatti, nell'applicativo è possibile scegliere in che lingua impostare la stanza di gioco al momento della sua creazione.

Il chooser avrà **30 secondi** per scegliere una delle 10 parole, altrimenti a tempo scaduto ne verrà scelta una casuale. In ogni caso la parola scelta sarà inviata al server, che, mandandola a sua volta a tutti gli altri giocatori, farà partire il gioco vero e proprio

3. STANZA IN GIOCO

La parola da indovinare sarà visibile in cima alla schermata della chat sottoforma di "_ _ _ _ _ _". Tutti i guesser potranno provare ad indovinarla scrivendo la parola, che hanno pensato, in chat, se essa compare in chat come messaggio normale, vuol dire che non era la parola giusta, se invece compare il messaggio di vittoria allora la parola era giusta, vengono aggiunti i punti al guesser vincitore e la partita finisce.

Altrimenti ogni 15 secondi verrà aggiunta una lettera in più fino a completare l'intera parola. se nessuno, entro il completamento della parola, riesce a indovinarla, vince il chooser, gli vengono assegnati i punti e la partita finisce.

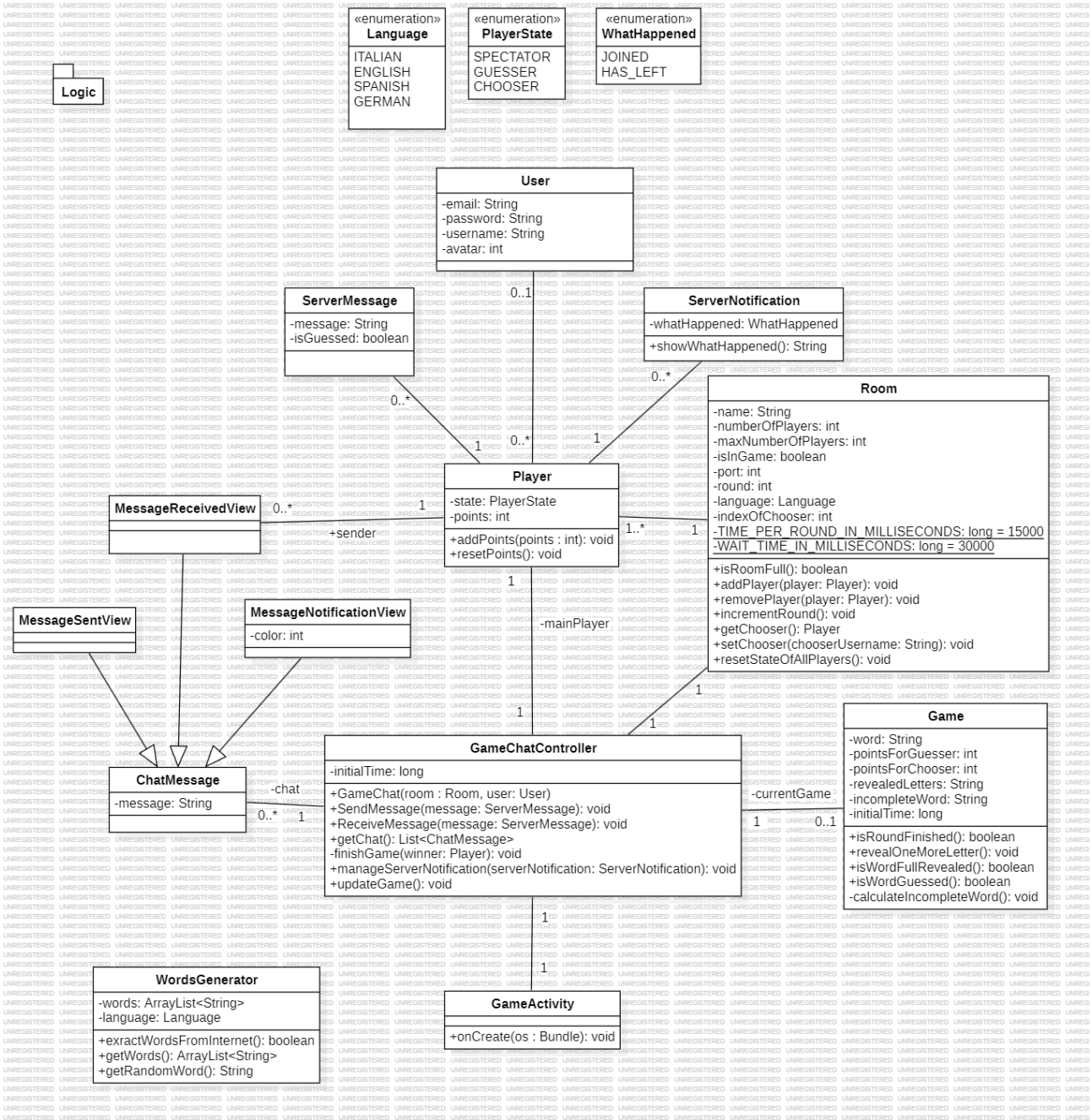
Dopo 5 secondi dalla fine di una partita, se ci sono sufficienti giocatori, ne comincerà una nuova, ripetendo questi ultimi due step. Se durante una partita i giocatori abbandonano la stanza e ne rimane solo uno si torna al primo step.

4.1 GESTIONE DEL PUNTEGGIO

La classifica dei giocatori basata sui relativi punteggi è visualizzabile cliccando l'icona in alto a destra nella schermata della chat

- Se il "Guesser" vince gli viene assegnato come punteggio la quantità di lettere che ancora non sono state rivelate
- Se il "Chooser" vince gli viene assegnato come punteggio: (15 – la lunghezza della parola)

5 STRUTTURA DEL CODICE



6 COMUNICAZIONE CON IL SERVER

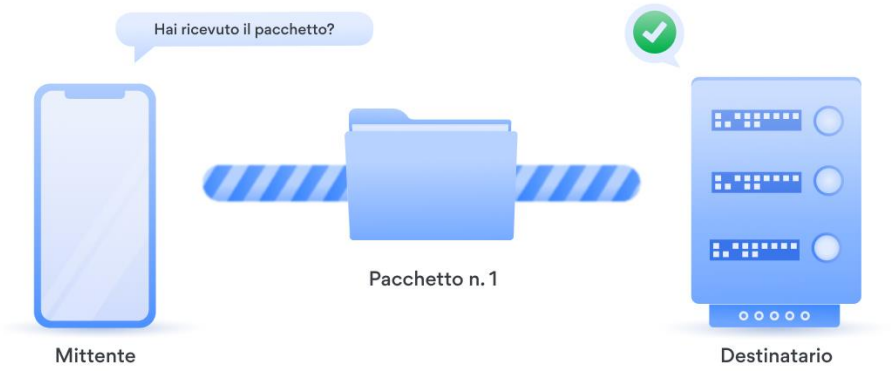
La comunicazione primaria del client con il server avviene attraverso una socket TCP per operazioni quali il login, la registrazione, la creazione e l'ingresso nelle stanze. Per la gestione effettiva delle interazioni nella stanza di gioco, inclusi l'invio dei messaggi e le notifiche al server, è stata scelta l'implementazione tramite Multicast UDP al fine di ottimizzare la velocità di trasmissione durante il gameplay.

6.1 COMUNICAZIONE TCP

La prima comunicazione con il server avviene tramite una socket TCP che instaura una connessione "punto a punto" con il client.

Infatti, TCP è un protocollo connessione-orientato, il che significa che stabilisce una connessione tra due computer prima di iniziare la trasmissione dei dati.

Questa connessione garantisce che i dati vengano inviati a un destinatario specifico e che non vengano intercettati da altri dispositivi.

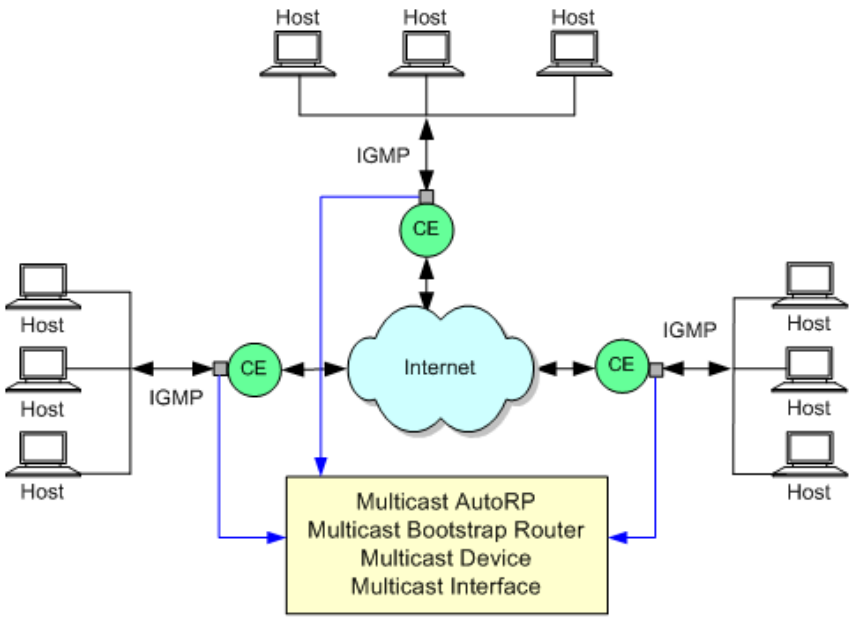


Questa prima socket è fondamentale per effettuare la fase di registrazione e login dell'utente, poi una volta autenticato, l'utente può creare, entrare o richiedere la lista delle stanze.

la connessione "punto a punto" di TCP è fondamentale per tenere traccia degli utenti online; infatti, se l'utente perde la connessione con il server ne abbiamo subito un riscontro e possiamo avvisare gli altri utenti in gioco.

6.2 COMUNICAZIONE UDP

La comunicazione in una stanza deve permettere il "broadcast" di messaggi tra i vari utenti connessi con quella stessa stanza. Dal punto di vista architetturale la soluzione più simile è multicast.



Multicast permette di creare più "gruppi" di socket UDP connesse tra di loro, a differenza del broadcast, dove un pacchetto viene inviato a tutti i dispositivi sulla rete, il multicast invia i dati solo a un gruppo di destinatari iscritti. Ogni gruppo del multicast corrisponderà ad una stanza di gioco.

6.3 GESTIONE DELL'INVIO DI UNA LETTERA OGNI 15 SECONDI

Allo scopo di massimizzare, dove possibile, l'indipendenza del client dal server, è stato scelto di evitare di inviare ogni volta una lettera in più, calcolata dal server, ma di far calcolare invece al client, in anticipo, al momento della scelta della parola, una stringa contenente già una successione delle lettere da sbloccare successivamente durante la partita e di inviare questa al server oltre alla parola scelta.

Per generare questa stringa, si prende la parola, le si tolgono i doppioni e si mischiano casualmente le lettere che sono rimaste. Questa operazione è effettuata dentro la classe WordChosen dal metodo **removeDuplicatesAndShuffle(String word).**

ES: Il chooser sceglie la parola "Successivo" da far indovinare agli altri giocatori. Essa viene mandata al server insieme alla seguente stringa (detta lettereMischiate):

SUCCESSIVO —*removeDuplicates*→ *SUCEIVO* —*Shuffle*→ *VSUCOIE*

così che queste due stringhe saranno ricevute da tutti i client. Adesso, senza alcuna ulteriore comunicazione con il server è possibile rivelare una lettera ogni 15 secondi uguale per tutti i client:

Passano 15 secondi e come prima lettera rivelata viene scelta la prima lettera di lettereMischiate: _ _ _ _ _ _ _ _ V _

Passano altri 15 secondi e come seconda lettera rivelata viene scelta la prossima lettera di lettereMischiate: S _ _ _ _ S S _ V _

E così via...

7 DOCKER E POSTGRESQL

Per salvare le credenziali di autenticazione dei vari utenti è stato usato un database **PostgreSQL**, provvisto della seguente tabella:

```
CREATE TABLE IF NOT EXISTS users (
    email VARCHAR(255) PRIMARY KEY,
    username VARCHAR(255) UNIQUE,
    password VARCHAR(255),
    avatar INT
)
```

PostgreSQL si sposa perfettamente con Docker. Utilizzando l'immagine ufficiale "postgres" di Docker Hub, è possibile avviare e gestire un'istanza di PostgreSQL in modo semplice e veloce.

Dunque, è stato scelto di containerizzare il nostro database tramite **Docker**. Ciò ci fa godere di altri vantaggi tra cui:

- **Portabilità**: Le applicazioni in container funzionano in modo uniforme su qualsiasi sistema operativo, garantendo una facile distribuzione e scalabilità.
- **Efficienza**: I container condividono il kernel del sistema operativo, riducendo l'overhead e ottimizzando l'utilizzo delle risorse.

Il nostro container viene generato con i valori:

- Database name: postgres
- User: postgres
- Password: default
- Port: 5432

8 GITHUB

Durante lo sviluppo è stato usato GitHub per il controllo delle versioni del codice sorgente e dei file di progetto. sono state usate due repository:

- Server: https://github.com/agostino-code/CSoketLSO/commits/main/
- Android: https://github.com/agostino-code/LSOProject/commits/master/