Ruba Mazzetto

Traccia

Il “ruba mazzetto” è un gioco di carte, variante semplificata della “scopa”, molto diffuso in Italia.

Si gioca con le carte da gioco italiane da 40 carte suddivise in 4 semi da 10 carte ciascuno. Spesso è giocato solo da 2 persone ma si può giocare anche in 3 e in 4.

Ogni giocatore, nel rispettivo turno, può prendere solo una carta dello stesso valore dal tavolo. Ogni qual volta viene effettuata una presa, il giocatore deve tenere scoperta l’ultima carta del proprio mazzo: se l’avversario ha in mano una carta dello stesso valore può “rubare” il mazzo (che deve essere sempre tenuto girato).

Dopo l’ultima mano, le carte rimaste sul tavolo vengono aggiunte a quelle del giocatore che ha effettuato l’ultima presa. Lo scopo del gioco è prendere quante più carte possibile: il punteggio è dato semplicemente dalla conta delle carte che ogni giocatore ha preso.

Implementare un sistema che permetta, a giocatori registrati, di sfidarsi nel gioco. Il sistema deve prevedere la possibilità di salvare lo stato della partita e riprenderlo in un secondo momento.

Infine, il sistema deve anche conservare (e naturalmente mostrare) lo storico dei risultati di ogni giocatore e relative statistiche.

Pianificazione

Il software richiesto necessita di diverse strutture dati e funzionalità che elencherò qui sotto:

- **utente** (struttura dati), necessita di:
  - funzionalità di registrazione e accesso con relativi controlli sulla validità di nome utente o password
  - persistenza su memoria di massa degli utenti registrati, possibilmente con password non in chiaro
- **carta**, **mazzo**, e **tavolo di gioco** (strutture dati) necessari per il vero e proprio svolgimento del gameplay e la sua rappresentazione
- **giocatore** (struttura dati) contiene informazioni sull’utente ed è provvisto di una **mano** di 3 carte al massimo e un suo **mazzo** che verrà usato per la conta dei punti
- **dati di gioco** (struttura dati), per memorizzare e manipolare i dati di una partita e poter successivamente salvare lo stato del gioco su richiesta o caricare una partita salvata, necessita di:
  - funzionalità di inizializzazione della partita da utenti che hanno effettuato l’accesso specificando il numero di giocatori che può variare fra i 2 e i 4

        - funzionalità di salvataggio e caricamento dei dati di gioco di una partita
        - ciclo di gioco con fasi di distribuzione delle carte, turni dei vari giocatori, attesa della mossa di un giocatore e schermata dei risultati
    - **storico match** (vettore di struttura dati) contenente per ogni giocatore l'esito e le informazioni su una partita giocata
    - **statistiche giocatore** (struttura dati) contenente rateo di vincita, vittorie, sconfitte e statistiche su punti e mazzetti rubati (minimi, medi, massimi) in base al numero di giocatori

Funzionamento del software

Il software è stato diviso fra backend sviluppato interamente in C e frontend sviluppato in python html css e javascript. La IPC (Inter-process communication) fra backend e frontend avviene attraverso un piccolo server con socket TCP in ascolto su localhost sulla porta 11666.

Le richieste vengono inoltrate dal client python in plain text con comandi e argomenti separati da uno spazio e viene inoltrata una risposta JSON che permette al client di visualizzare le varie informazioni richieste attraverso la sua GUI.

Backend

Il funzionamento del backend è relativamente semplice, all'avvio del programma viene inizializzata la memoria per utenti, storico partite, statistiche e partite salvate, e se esistenti dati binari per gli stessi vengono caricati in memoria. Viene impostato il seed randomico e viene avviato il server che resta in ascolto per un client.

Non appena il client (frontend) si connette al server, il backend si mette in ascolto di messaggi da parte del frontend e ogni volta che ne riceve uno lo passa al tokenizer che provvede a dividere la stringa per ogni spazio che incontra.

Il tokenizer completata la divisione della stringa riconosce I vari token sequenzialmente e se riconosce un comando valido utilizza I parametri del comando per dialogare con i vari moduli del programma,

Il modulo User

Il modulo user contiene oltre al vettore degli utenti registrati, un vettore contenente gli id degli utenti loggati. Durante la sua inizializzazione, alloca i vettori sopracitati e carica dal file degli utenti eventuali utenti registrati durante una sessione precedente.

Il modulo user fornisce funzionalità di registrazione, login e aggiornamento della password di un utente, per essere valido un nome utente deve essere formato da almeno 5 caratteri alfanumerici o underscore, mentre una password deve essere contenere almeno 1 lettera maiuscola, 1 carattere speciale e 1 cifra ed essere lunga almeno 8 caratteri… il massimo di lunghezza per password e nome utente è di 25 caratteri.

Alla registrazione di un nuovo utente vengono aggiunti ad esso due timestamp uno per la data di creazione, uno per la data di modifica (che viene aggiornato in caso di update della password) e vengono generate le statistiche dell'utente con valori standard.

Il modulo History

Il modulo history contiene lo storico di tutte le partite effettuate e prevede le funzionalità per filtrare lo storico per utente. All'avvio viene inizializzata la memoria per il vettore contenente lo storico e se il file binario esiste vengono caricati i dati dal file.

Il modulo Statistics

Il modulo statistics contiene le statistiche per ogni utente divise per categoria (2 giocatori, 3 giocatori, 4 giocatori). Contiene le funzionalità per aggiornare le statistiche di un utente in base al suo storico partite, leggere le statistiche per id utente o ottenere una leaderboard con i migliori giocatori per categoria. All'avvio viene inizializzata la memoria per il vettore contenente le statistiche e se il file binario esiste vengono caricati i dati dal file.

Il modulo Game

Il modulo game contiene un flag che indica se la partita e in corso o meno, un enum indicante lo stato della partita, 4 players, numero di giocatori nella partita, tavolo di gioco e mazzo del mazziere. Contiene le funzionalità per l'avvio di una nuova partita, salvataggio della partita corrente, caricamento di una partita salvata, avanzamento della partita corrente (divisa per stati), lancio di carta sul tavolo, presa di una carta dal tavolo e presa del mazzetto di un avversario. Al termine di una partita viene aggiunta la partita corrente nello storico partite e vengono aggiornate le statistiche per ogni giocatore.

All'avvio viene inizializzata la memoria per il vettore contenente le partite salvate e se il file binario esiste vengono caricati i dati dal file.

Frontend

Il frontend è un client connesso in locale al socket del backend, è composto da una piccola webapp sviluppata in flask contenente due principali routes:

- / → da qui cominciano tutte le route delle varie pagine della webapp
- /api/ → da qui cominciano tutte le route per effettuare richieste al backend e ritornano tutte una risposta JSON elaborata dal server

l'aggiornamento del contenuto delle pagine avviene attraverso la chiamata ajax di jquery che permette di effettuare una richiesta alle route /api/ ed elaborare successivamente la risposta JSON in modo da aggiungere gli elementi html della pagina in questione nei loro rispettivi div.

In tal modo non è necessario effettuare il refresh della pagina per ogni azione sulla GUI e di conseguenza evitare di complicare la comunicazione fra javascript e python che altrimenti dovrebbe prevedere un sistema per aggiornare il contesto del template della pagina richiesta.