Specifiche Esame per superamento del Corso Programmazione Avanzata

A.A. 2021/2022

Gruppo: NO

Si chiede di realizzare un back-end utilizzando i seguenti framework / librerie:

- tic-tac-toe-ai-engine https://www.npmjs.com/package/tic-tac-toe-ai-engine
- Express
- Sequelize
- RDBMS a scelta dello studente (es. Postgres, MySQL, sqlite,…)

Descrizione del progetto:

Si realizzi un sistema che consenta di gestire il gioco del Tic Tac Toe. In particolare, il sistema deve prevedere la possibilità di far interagire due utenti (autenticati mediante JWT) o un utente contro l'elaboratore. Ci possono essere più partite attive in un dato momento. Un utente può allo stesso tempo partecipare ad una ed una sola partita. Si chiede di sviluppare anche la possibilità di giocare contro l'elaboratore (di seguito IA). Nel caso di IA si faccia riferimento a quanto riportato nella libreria tic-tac-toe-ai-engine. Si chiede di realizzare le seguenti funzionalità:

- Dare la possibilità di creare una nuova partita specificando (codifica a scelta dello studente):
  - Tipologia: utente contro utente o utente contro IA
  - e-mail dell'avversario che sarà usata poi per autenticare le richieste mediante token JWT (solo nel caso di partita utente contro utente).
- Per ogni partita viene addebitato un numero di token in accordo con quanto segue:
    - 0.50 all'atto della creazione se utente contro utente, 0.75 se utente con IA.
    - 0.015 per ogni mossa da parte degli utenti (anche IA)
  - Il modello può essere creato se c'è credito sufficiente ad esaudire la richiesta (se il credito durante la partita scende sotto lo zero si può continuare comunque).
- Creare la rotta per effettuare una mossa in una data partita verificando se questa è ammissibile o meno
- Creare la rotta per abbandonare una partita
- Creare una rotta per valutare lo stato di una data partita (di chi è il turno, se è terminata, vincitore,…)
- Creare una rotta per restituire lo storico delle mosse dando la possibilità:
  - Selezionare formato in uscita (CSV, JSON); codifica scelta dello studente.
  - Periodo temporale (data inferiore, data superiore, periodo)
- Creare una rotta per restituire la classifica (rotta pubblica senza autenticazione); si dia la possibilità di ordinare in modo crescente o decrescente. Devono essere riportate il numero di partite vinte, vinte per abbandono, perse e perse per abbandono, distinguendo anche se le partite sono state fatte contro utenti reali o IA. Il punteggio si ottiene calcolando il numero di partire vinte anche considerando il caso di partita vinta per abbandono. L'output è un JSON.

Le richieste devono essere validate (es. utente che scelga una partita non esistente).

Ogni utente autenticato (ovvero con JWT) ha un numero di token (valore iniziale impostato nel seed del database).

Nel caso di token terminati ogni richiesta da parte dello stesso utente deve restituire 401 Unauthorized.

Prevedere una rotta per l'utente con ruolo admin che consenta di effettuare la ricarica per un utente fornendo la mail ed il nuovo "credito" (sempre mediante JWT). I token JWT devono contenere i dati essenziali.

Il numero residuo di token deve essere memorizzato nel db sopra citato.

Si deve prevedere degli script di seed per inizializzare il sistema.

Si chiede di utilizzare le funzionalità di middleware.

Si chiede di gestire eventuali errori mediante gli strati middleware sollevando le opportune eccezioni.

Si chiede di commentare opportunamente il codice.

Note:

Nello sviluppo del progetto è richiesto l'utilizzo di Design Pattern che dovranno essere documentati opportunamente nel Readme.MD. È preferibile una implementazione in typescript.

I token JWT da usare possono essere generati attraverso il seguente link: https://jwt.io/ o mediante gli script mostrati a lezione

La chiave privata da usare lato back-end deve essere memorizzata un file .env e caricata mediante la libreria

Specifiche Repository

- Il codice deve essere reso disponibile su piattaforma github con repo pubblico
- Nel repository è obbligatario inserire un Readme.md che descriva:
    - Obiettivo del progetto
    - Progettazione
        - diagrammi UML
        - descrizione dei pattern usati motivandone la scelta
    - Come avviare il progetto mediante docker o docker-compose (preferibile) per comporre i servizi richiesti.
    - Test del progetto mediante chiamate effettuate con curl o wget o con Postman
- Il Readme.MD può essere redatto in lingua italiana o inglese (non vi saranno differenziazioni nel processo di valutazione)

Specifiche Consegna

- La consegna avviene esclusivamente mediante moodle all'indirizzo di seguito riportato dove dovranno essere indicati:
    - URL del repository pubblico
    - Commit id che verrà usata dal docente per effettuare la valutazione.
    - Data per lo svolgimento dell'esame
- Indirizzo per la consegna: https://learn.univpm.it/mod/assign/view.php?id=431110
-

Buon lavoro 😉

Il docente

Adriano Mancini