**Specifiche Esame per superamento del Corso Programmazione Avanzata**

**A.A. 2023/2024**

Gruppo: NO

---------------------------------------------------------------------------

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

* Node.JS
* Express
* Sequelize
* RDBMS a scelta dello studente (es. Postgres, MySQL, sqlite,…)
* <https://github.com/loks0n/rapid-draughts>

**Descrizione del progetto:**

Si realizzi un sistema che consenta di gestire il gioco della dama. In particolare, il sistema si basa su un utente (sempre autenticato con JWT) che gioca contro un altro giocatore o contro l’intelligenza artificiale. Ci possono essere più partite attive in un dato momento. Un utente può allo stesso tempo partecipare ad una ed una sola partita.

* Dare la possibilità di creare una nuova partita seguendo <https://github.com/loks0n/rapid-draughts> ; la partita deve essere Utente contro intelligenza artificiale (IA) scegliendo il livello di difficoltà.
* In particolare, è necessario validare la richiesta di creazione della partita. Per ogni partita viene addebitato un numero di token in accordo con quanto segue:
  + - 0.45 all’atto della creazione
    - 0.0125 per ogni mossa fatta dal giocatore
  + 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 (si consiglia di valutare quanto presente in *Board Configuration – JSON*)
* Creare una rotta per valutare lo stato di una data partita; una partita si considera chiusa quando:
  + Un utente / AI vince
  + L’utente abbandona.
  + Per ogni partita vinta al giocatore si dà 1 punto; per ogni partita interrotta -0.5 punti;
* Creare una rotta per restituire lo storico delle mosse.

Le richieste devono essere validate (es. utente che scelga un evento che 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.

I dati di cui sopra devono essere memorizzati in un database esterno interfacciato con Sequelize. La scelta del DB è a discrezione degli studenti.

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.

Implementazione in typescript.

I token JWT da usare possono essere generati attraverso il seguente link: <https://jwt.io/> (token JWT non deve contenere il payload della richiesta, ma solo i dati strettamente necessari per autenticare ed autorizzare le richieste).

La chiavi da usare lato back-end devono essere memorizzata un file .env

**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 (casi d’uso, diagrammi delle sequenze)
    - descrizione dei pattern usati motivandone la scelta
  + Come avviare il progetto mediante docker-compose per comporre i servizi richiesti (fornire tutti i file neceessari).
  + Test del progetto mediante chiamate effettuate Postman / Newman (fornire collection)
* 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=531167>

Buon lavoro 😉

Il docente, Adriano Mancini