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:

- Node.JS
- Express
- Sequelize
- RDBMS a scelta del gruppo (es. Postgres, MySQL, sqlite,…)
- https://www.npmjs.com/package/ngraph.path

Descrizione del progetto:

Si realizzi un sistema che consenta di gestire la creazione e valutazione di modelli di ottimizzazione su grafo. In particolare, il sistema deve prevedere la possibilità di gestire l'aggiornamento di pesi effettuato da utenti autenticati (mediante JWT). Il progetto simula il concetto del crowd-sourcing dove gli utenti possono contribuire attivamente. Un esempio di applicazione è quello di tenere traccia dei minuti che sono necessari per percorre un determinato tratto di strada. Di seguito il dettaglio di quanto si deve realizzare:

- Dare la possibilità di creare un nuovo modello (anche orientato) seguendo l'interfaccia definita nella sezione API di https://www.npmjs.com/package/ngraph.path ed in particolare di specificare il grafo con i relativi pesi
    - In particolare, è necessario validare la richiesta di creazione del modello
    - Per ogni modello valido deve essere addebitato un numero di token in accordo con quanto segue:
        - 0.25 per ogni nodo
        - 0.01 per ogni arco
    - Il modello può essere creato se c'è credito sufficiente ad esaudire la richiesta.
- Eseguire il modello; per ogni esecuzione deve essere applicato un costo pari a quello addebitato nella fase di creazione. Si deve dare all'utente la possibilità di scegliere tra diversi algoritmi previsti come `aStar`, `aGreedy`, `nba`. Dare la possibilità anche di scegliere la metrica e l'euristica da usare (a tal proposito si realizzino oltre che la norma2 come già nella documentazione anche la norma 1 sui valori di x, la norma 1 sui valori di y, o la distanza Manhattan.
- Ritornare il risultato sotto forma di JSON. Il risultato deve anche considerare il tempo impiegato per l'esecuzione. L'esecuzione del modello deve prevedere ovviamente la necessità di specificare start, goal. Ritornare il percorso ed il costo associato a tale percorso (per costo si intende non i termini di token, ma in termini di percorso ottimo sul grafo considerando i pesi).
- Gestire le richieste di cambio peso per uno o più archi da parte degli utenti autenticati.
    - Si procede con aggiornare il peso dell'arco mediante una media esponenziale del tipo p(i,j) = alpha * p(i,j) + (1 – alpha)*p_new(i,j) dove p(i,j) è il precedente costo associato all'arco che collega nodi i-j e p_new è il nuovo costo suggerito dall'utente. alpha è uguale per tutti i modelli e deve essere gestito mediante una variabile di env. alpha deve essere > 0 e < 1; valori errati nella variabile di env devo determinare l'avvio del sistema con una configurazione pari ad alpha = 0.9
- Restituire l'elenco dei modelli associati all'utente filtrato per (filtri in AND):
    - Numero di nodi
    - Numero di archi

- Restituire l'elenco delle esecuzioni riportando i dati come id esecuzione, tempo esecuzione, costo esecuzione, id del modello, costo relativo alla soluzione ottima, start, goal
- Effettuare una "simulazione" che consenta di variare il peso relativo ad un arco specificando il valore di inizio, fine ed il passo di incremento (es. start = 1 stop = 2 e passo 0.05 significa trovare in modo iterativo la soluzione e percorso a costo minimo provando con peso prima 1.0 poi 1.05, poi 1.1 ,….); l'utente deve poter specificare anche il metodo risolutivo, la funzione di distanza e l'euristica.
    - Le richieste di simulazione devono essere validate (es. range non ammissibili).
    - È necessario ritornare l'elenco di tutti i risultati; ritornare anche il best result con la relativa configurazione dei pesi che sono stati usati.

Le richieste devono essere validate.

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. Nella fase di dimostrazione (demo) è necessario prevedere almeno 2 modelli diversi con almeno due versioni con una complessità minima di 8 nodi e 16 archi.

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/

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=332114

Buon lavoro 😉

Il docente, Adriano Mancini