DOCUMENTAZIONE PROGETTO SVILUPPO WEB AVANZATO

• FUNZIONALITÀ REALIZZATE (REST-services) :

LOGIN UTENTE
URL: /auth/login
VERBO: POST
prende in input un oggetto contenente mail e password di un utente e restituisce un oggetto contenente il relativo token di sessione/autenticazione, autenticando l'utente se le credenziali di accesso sono corrette.

LOGOUT UTENTE
URL: /auth/logout
VERBO: POST
Prende in input un token di sessione/autenticazione e chiude la relativa sessione, invalidando il token.

INSERIMENTO NUOVO UTENTE NEL SISTEMA
URL: /users
VERBO: POST
Inserisce un nuovo utente nel sistema. L'oggetto passato come payload conterrà tutti i dati del profilo utente, ad esempio:

```
{
	"nome":"pinco",
	"cognome":"pallino",
	"email":"pp@gmail.it",
	"pwd":"abcdef",
	"dataDiNascita":"1995-04-27",
	"città":"Roma",
	"telefono":3456789012,
	"titoloDiStudi":"Laurea Base in Informatica",
	"imgProfilo":"profilo.png"
}
```

e restituisce la url necessaria ad accedere ai dati dell'utente appena creato, ad esempio http://localhost:8080/teachTime/MainApplication/rest/users/1.

Nota: imgProfilo contiene il nome con il quale è salvata l'immagine del profilo dell'utente sul server. La procedura di salvataggio dell'immagine sul server è stata implementata tramite la servlet UploadFoto contenuta nel package "server". In un'eventuale operazione di registrazione quindi, dovrà essere chiamata prima tale servlet in modo che l'immagine caricata dall'utente venga spedita sul server, dopodichè, utilizzando il nome con la quale è stata salvata, la si può indicare nel JSON mostrato sopra associandola all'utente.

ESTRAZIONE UTENTE PER CHIAVE:
URL: users/{ID} - auth/{SID}/users/{ID}
VERBO: GET
Restituisce il profilo dell'utente {ID} sotto forma dello stesso oggetto JSON utilizzato come payload per l'inserimento (POST) tramite la URL /users, ad esclusione della password. Il servizio può essere raggiunto indifferentemente dalle due url specificate, senza variazioni di contenuto.

AGGIORNAMENTO PROFILO UTENTE PER CHIAVE:
URL: auth/{SID}/users/{ID}
VERBO: PUT
Aggiorna il profilo dell'utente {ID} che è loggato con session id {SID}. Il payload è lo stesso oggetto utilizzato per l'inserimento (POST) tramite la URL /users eccetto che per i campi nome,cognome,email,pwd,dataDiNascita che non sono presenti in questo caso in quanto non aggiornabili.
NOTA: Qualunque utente loggato ha la possibilità di aggiornare solo il proprio profilo, quindi se {ID} non coincide con l'id dell'utente che ha session id {SID}, il servizio non ha esito positivo.

INSERIMENTO RIPETIZIONE:
URL: auth/{SID}/ privateLessons
VERBO: POST
Inserisce nel sistema una nuova ripetizione che ha come tutor l'utente con session id {SID}. L'oggetto passato come payload conterrà tutti i dati della ripetizione, ad esempio:

```
{
"città":"Roma",
"luogoIncontro":"a scelta dello studente",
"costo":12,
"descr":"descrizione della ripetizione",
"categoria_key":1,
"materie":[
                    {  "nome":"programmazione python"},
                    {  "nome":"database"}
          ]
}
```

e restituisce la URL necessaria per accedere alla risorsa appena creata, ad esempio http://localhost:8080/teachTime/MainApplication/rest/privateLessons/1

NOTA: se le materie indicate non sono presenti nel db, verranno aggiunte come materie appartenenti alla categoria con chiave categoria_key.

ESTRAZIONE RIPETIZIONE PER CHIAVE:
URL: privateLessons/{ID} - auth/{SID}/privateLessons/{ID}

VERBO: GET
Restituisce le informazioni relative alla ripetizione {ID} sotto forma dello stesso oggetto utilizzato come payload dell'inserimento (POST) tramite la url /privateLessons, con in più le informazioni relative alla categoria a cui la ripetizione appartiene e quelle relative al tutor che la offre.
Il servizio può essere raggiunto indifferentemente dalle due url specificate, senza variazioni di contenuto.

ESTRAZIONE DELLE RIPETIZIONI OFFERTE:
URL: auth/{SID}/privateLessons/my
VERBO: GET
Restituisce la lista delle ripetizioni offerte dall'utente loggato con session id {SID}, ognuna nello stesso formato utilizzato per l'inserimento (POST) di una ripetizione tramite la url auth/{SID}/privateLessons, con l'aggiunta delle informazioni riguardanti il tutor che le offre e la categoria a cui appartenegono le materie

ESTRAZIONE DELLE RIPETIZIONI OFFERTE PER FILTRO:
URL: privateLessons?FILTER - auth/{SID}/privateLessons?FILTER
VERBO: GET
Restituisce la lista delle ripetizioni (sottoforma di oggetti della stessa forma di quello utilizzato per l'inserimento (POST) di una ripetizione tramite la url auth/{SID}/privateLessons, con l'aggiunta delle informazioni riguardanti il tutor che le offre e la categoria a cui appartenegono le materie) filtrate in base ai parametri contenuti in FILTER che possono essere: città, categoria, materia e tutor_id. La città oppure il tutor_id devono essere indicate obbligatoriamente. Nel caso in cui venga indicato il parametro città, ad esso potranno essere combinati opzionalmente entrami i parametri categoria e materia oppure solo uno di essi. Nel caso in cui venga indicato il parametro tutor_id invece non saranno valide combinazioni con gli altri parametri.
Il servizio è raggiungibile dalle due url indicate, ma nel caso della url nella quale è indicato il session id {SID}, la lista delle ripetizioni non includerà quelle offerte dall'utente a cui appartiene il session id {SID}.

AGGIORNAMENTO RIPETIZIONE PER CHIAVE (prezzo/ora,descrizione,luogo incontro, lista materie):
URL: auth/{SID}/privateLessons/{ID}
VERBO: PUT
Aggiorna le informazioni relative alla ripetizione {ID} se questa è offerta dal tutor autenticato con session id {SID}. Il payload è lo stesso oggetto utilizzato come payload dell'inserimento (POST) tramite la url /repetitions, ad esclusione di "tutor_key", che verrà ricavata in base al valore di {SID}, e di "categoria_key".
NOTA: le materie verranno completamente sostituite, quindi se si vuole lasciare inoltrato l'elenco delle materie è necessario indicarlo, mentre se si vuole aggiungere o rimuovere qualche materia è necessario indicare l'elenco preesistente e aggiungere/rimuovere da esso la materia specifica.

CANCELLAZIONE RIPETIZIONE PER CHIAVE:
URL: auth/{SID}/privateLessons/{ID}

VERBO: DELETE
Elimina la ripetizione {ID} dal sistema se la ripetizione è offerta dall'utente loggato con session id {SID}.

INSERIMENTO PRENOTAZIONE RELATIVA AD UNA RIPETIZIONE:
URL: auth/{SID}/privateLessons/{ID}/bookings
VERBO: POST
Inserimento nel sistema di una nuova prenotazione alla ripetizione {ID} da parte dell'utente autenticato con session id {SID}. L'oggetto passato come payload conterrà tutti i dati della prenotazione, ad esempio:

{
"data":"2007-09-25T15:40:00+02",
"materia_key":2,
"stato":0,
"descr":"ripetizione descrizione"
}

dove materia_key è la chiave della materia (tra quelle contenute nella ripetizione) sulla quale vuole ricevere la ripetizione.
Resitituisce la URL necessaria ad accedere alla risorsa appena inserita, ad esempio http://localhost:8080/teachTime/MainApplication/rest/privateLessons/86/bookings/1

ESTRAZIONE PRENOTAZIONI PER CHIAVE STUDENTE:
URL: auth/{SID}/users/{ID}/bookings
VERBO: GET
Restituisce la lista delle prenotazioni effettuate dallo studente {ID} sotto forma dello stesso oggetto utilizzato come payload dell'inserimento (POST) tramite la url /repetitons/{ID}/bookings, con l'aggiunta delle informazioni riguardanti la ripetizione relativa alla prenotazione e il tutor che la offre.
(Questa API serve a recuperare la lista delle prenotazioni effettuate da uno studente per le quali lo studente non ha ancora rilasciato un feedback.)

INSERIMENTO FEEDBACK RELATIVO AD UNA RIPETIZIONE:
URL: auth/{SID}/privateLessons/{pl_ID}/bookings/{ID}/feedbacks
VERBO: POST
Inserimento nel sistema di un feedback da parte dell'utente autenticato con session id {SID} relativamente alla ripetizione {pl_ID} per la quale ha effettuato la prenotazione {ID}.
L'oggetto passato come payload conterrà tutti i dati necessari a rilasciare un feedback, ad esempio:

{
"voto":5,
"recensione":"ottimo"
}

LISTA FEEDBACK RICEVUTI DA UN TUTOR:
URL: users/{ID}/feedbacks - auth/{SID}/users/{ID}/feedbacks
VERBO: GET
Restituisce la lista dei feedback ricevuti dal tutor {ID}, ognuno sottoforma dello stesso oggetto JSON usato come payload nell'inserimento (POST) di una prenotazione tramite la url auth/{SID}/privateLessons/{ID}/bookings con l'aggiunta dei campi voto e recensione.
Il servizio può essere raggiunto indifferentemente dalle due url specificate, senza variazioni di contenuto.

ESTRAZIONE VOTO MEDIO TUTOR:
URL: /users/{ID}/feedbacks/avg - auth/{SID}/users/{ID}/feedbacks/avg
VERBO: GET
Restituisce il voto medio dell'utente calcolato sulla base dei voti contenuti in ogni feedback da esso ricevuto.
Il servizio può essere raggiunto indifferentemente dalle due url specificate, senza variazioni di contenuto.

RECUPERO LISTA MATERIE PER CATEGORIA:
URL: /categories/{ID}/subjects
VERBO: GET
Restituisce la lista delle materie appartenenti alla categoria {ID}.

- ## FUNZIONALITÀ REALIZZATE (client)

Il client sviluppato (SPA) permette di cercare le ripetizioni filtrandole per città, categoria e materia. Una volta effettuata la ricerca viene visualizzata la lista delle ripetizioni coerenti con i parametri inseriti. Ogni ripetizione mostra tutte le informazioni necessarie all'utente che sta effettuando la ricerca. Per implementare questa funzionalità sono stati usati due dei servizi rest implementati, quello che permette di filtrare le ripetizioni e l'altro che permette di recuperare la lista delle materie appartenenti ad una certa categoria.
Inoltre il client permette all'utente di eseguire la procedura di login. Una volta loggato, l'utente può accedere a due funzionalità. La prima è quella di ricerca, offerta anche agli utenti non loggati. Nel caso in cui l'utente è loggato viene restituita la lista delle ripetizioni delle quali l'utente non è tutor e che risultano coerenti con i parametri inseriti.
L'altra funzionalità permette all'utente loggato di inserire una nuova ripetizione nel sistema, indicando tutte le informazioni necessarie.
Infine all'utente loggato è (ovviamente) permesso di effettuare il logout e di tornare anonimo.

- ## FUNZIONALITÀ OPZIONALI REALIZZATE:

Implementazione della servlet UploadFoto che permette di salvare sul server l'immagine del profilo che l'utente sceglie nel form di registrazione.

- ## DIPENDENZE SOFTWARE:

CLIENT: jQuery, Bootstrap, Font Awesome

SERVER: Jersey, Jackson

- **SCHERMA RELAZIONALE DELLA BASE DI DATI:**