Progetto “CourseWebServices”

Versione 1.0

Premessa

Il presente progetto estende il progetto *CourseWeb* relativo al corso di Ingegneria del Web (si veda http://www.di.univaq.it/gdellape/content.php?page=students&crs=iw17). Per prima cosa, quindi, sarà necessario leggere questa specifica di base.

Tuttavia, non si dovrà realizzare la *web application* con tutte le funzionalità descritte nella specifica base, ma solo la relativa base di dati, a cui poi affiancare un servizio web RESTful che esporrà questi dati e alcune delle relative funzionalità di accesso e modifica. Si dovrà infine implementare, utilizzando HTML5 e JQuery, un semplicissimo client per questo servizio, utile anche a testarne la correttezza.

Nota: il servizio richiesto realizza, in pratica, un disaccoppiamento della gestione dati (e di alcune funzionalità correlate), che risiede necessariamente sul server, dalla logica di business e dal rendering dell'interfaccia, che potrebbero essere totalmente gestite dal client con Javascript, come spesso accade nelle SPA (single page application, applicazioni a pagina singola).

Il Servizio

Basandosi sulla base di dati appena descritta, si costruisca il servizio RESTful specificato di seguito. La logica del servizio potrà, *opzionalmente*, utilizzare una libreria ORM per accedere ai dati.

Note: nella specifica che segue gli elementi tra parentesi graffe sono *placeholder*, la parentesi quadre indicano elementi opzionali e le parentesi tonde, insieme al carattere *pipe* “|”, sono utilizzate per indicare delle alternative. Se non altrimenti specificato, il formato dati per l'input e l'output è JSON, quindi andranno progettate opportune strutture per contenere al meglio i dati richiesti in questo formato.

URL: auth/login

VERBO: POST

Prende in input un oggetto contenente username e password di un utente e restituisce un oggetto contenente il relativo *token* di sessione/autenticazione.

URL: auth/logout

VERBO: POST

Prende in input un token di sessione/autenticazione e chiude la relativa sessione, invalidando il token.

URL: auth/{SID}/users

VERBO: GET

Restituisce la lista delle URI che rappresentano tutti agli utenti presenti nel sistema.

Esempio di output:

```
["http://courseweb.it/auth/123/users/1", "http://courseweb.it/auth/123/users/2",
"http://courseweb.it/auth/123/users/3"]
```

L'operazione viene eseguita nel contesto della sessione con token {SID} e solo se l'utente associato è un amministratore (altrimenti potreste restituire un errore 403 - *Forbidden*).

VERBO: POST

Inserisce un nuovo utente nel sistema. L'oggetto passato come payload conterrà tutti i dati del profilo utente, ad esempio

```
{"username": "pinco", "password": "in_chiaro!"}
```

L'operazione viene eseguita nel contesto della sessione con token {SID} e solo se l'utente associato è un amministratore (altrimenti potreste restituire un errore 403 - *Forbidden*).

URL: auth/{SID}/users/{ID}

VERBO: GET

Restituisce il profilo dell'utente {ID} sotto forma dello stesso oggetto utilizzato come payload per l'inserimento (POST) tramite la URL auth/{SID}/users, ad esclusione della password.

L'operazione viene eseguita nel contesto della sessione con token {SID} e solo se l'utente associato è un amministratore (altrimenti potreste restituire un errore 403 - *Forbidden*).

VERBO: PUT

Aggiorna il profilo dell'utente {ID}. Il payload è lo stesso oggetto utilizzato per l'inserimento (POST) tramite la URL auth/{SID}/users.

L'operazione viene eseguita nel contesto della sessione con token {SID} e solo se l'utente associato è un amministratore (altrimenti potreste restituire un errore 403 - *Forbidden*).

VERBO: DELETE

Elimina il profilo dell'utente {ID}.

L'operazione viene eseguita nel contesto della sessione con token {SID} e solo se l'utente associato è un amministratore (altrimenti potreste restituire un errore 403 - *Forbidden*).

Ognuna delle URL che seguono potrà essere preceduta dal prefisso auth/{SID}/, in cui {SID} è un token di sessione restituito da auth/login. Se presente, questo segmento farà sì che le operazioni richieste siano eseguite nel contesto della sessione con token {SID} e quindi con i permessi dell'utente associato. In assenza di questo prefisso, il sistema eseguirà le operazioni come se fossero richieste da un utente anonimo (con le relative restrizioni!). *Suggerimento: se strutturate bene il servizio, sarà semplice rispondere alle URL seguenti con o senza il prefisso di autenticazione usando lo spesso codice.*

URL: courses/({YEAR}|current)?{FILTER}

VERBO: GET

Restituisce la lista delle URI che rappresentano tutti i corsi presenti nel sistema, includendo solo quelli con informazioni associate all'anno accademico corrente (*current*) o a quello specificato con {YEAR}. La lista potrà essere ulteriormente filtrata tramite la query string {FILTER} che può contenere tutti i parametri previsti per la ricerca sul catalogo dei corsi nel progetto CourseWeb (ad esempio nome, codice, ecc.).

Esempi di output:

Per la URL *courses/current*

```
["http://courseweb.it/courses/current/1", "http://courseweb.it/courses/current/2"]
```

Per la URL *courses/2015*

```
["http://courseweb.it/courses/2015/1", "http://courseweb.it/courses/2015/3"]
```

Per la URL *auth/123/courses/2015*

```
["http://courseweb.it/auth/123/courses/2015/1",
"http://courseweb.it/auth/123/courses/2015/3"]
```

VERBO: POST

Inserisce un nuovo corso nel catalogo dell'anno accademico {YEAR} o in quello corrente (*current*). Il payload è lo stesso oggetto restituito dalla lettura (GET) tramite la URL

courses/({YEAR}|current)/{ID}. I filtri {FILTER}, in questo caso, non sono ammessi (se presenti potreste restituire un errore 400 – *Bad Request*).

URL: courses/history/{ID}

VERBO: GET

Restituisce la lista degli anni accademici per cui esistono informazioni relative al corso {ID} insieme alle URL necessarie ad accedervi, sotto forma di un oggetto JSON del tipo

```
[{"year":2015,url:"http://courseweb.it/courses/2015/1"},
{"year":2016,url:"http://courseweb.it/courses/2016/1"}]
```

URL: courses/({YEAR}|current)/{ID}[/it|/en]

VERBO: GET

Restituisce tutte le informazioni inerenti il corso {ID} relative all'anno accademico corrente (*current*) o a quello specificato con {YEAR}. Aggiungendo uno dei due suffissi /it o /en verranno restituite solo le informazioni presenti nella lingua corrispondente, altrimenti l'oggetto conterrà i dati in entrambe le lingue. *Dovrete progettare voi un formato JSON adeguato per entrambi i casi.*

URL: courses/({YEAR}|current)/{ID}

VERBO: PUT

Aggiorna le informazioni inerenti il corso {ID} relative all'anno accademico {YEAR} o a quello corrente (*current*). Il payload è lo stesso oggetto restituito dalla lettura (GET) tramite la URL courses/({YEAR}|current)/{ID}.

VERBO: DELETE

Cancella il corso {ID} dal catalogo dell'anno accademico {YEAR} oppure da quello dell'anno accademico corrente (*current*).

URL: courses/({YEAR}|current)/{ID}/basic[/it|/en]

VERBO: GET

Restituisce un oggetto contenente le sole informazioni *di base* (così come descritte nella specifica del progetto CourseWeb), estratte dall'oggetto restituito dalla GET su courses/({YEAR}|current)/{ID}[/it|/en]

URL: courses/({YEAR}|current)/{ID}/syllabus[/it|/en]

VERBO: GET

Restituisce un oggetto con il solo sillabo/programma estratto dall'oggetto restituito dalla GET su courses/({YEAR}|current)/{ID}[/it|/en]

URL: courses/({YEAR}|current)/{ID}/teachers

VERBO: GET

Restituisce la lista dei docenti presenti nell'oggetto restituito dalla GET su courses/({YEAR}|current)/{ID}.

Esempio di output:

```
["Pinco Pallino","Paolino Paperino"]
```

VERBO: PUT

Aggiorna la lista dei docenti per il corso {ID} relativa all'anno accademico {YEAR} o a quello corrente (*current*). Il payload è lo stesso oggetto restituito dalla lettura (GET) tramite la URL courses/({YEAR}|current)/{ID}/teachers.

URL: courses/({YEAR}|current)/{ID}/relations

VERBO: GET

Restituisce la lista dei corsi in relazione con il corso {ID}, estratta dall'oggetto restituito dalla GET su courses/({YEAR}|current)/{ID}. L'oggetto restituito dovrà contenere tre liste di URI, rispettivamente per i corsi propedeutici, quelli mutuati e i moduli:

```
{"propedeutics": ["http://courseweb.it/courses/current/1",
"http://courseweb.it/courses/current/2"], "same":
["http://courseweb.it/courses/current/3"], "modules":[] }
```

VERBO: PUT

Aggiorna la lista dei corsi in relazione con il corso {ID} relativa all'anno accademico {YEAR} o a quello corrente (*current*). Il payload è lo stesso oggetto restituito dalla lettura (GET) tramite la URL courses/({YEAR}|current)/{ID}/relations.

URL: courses/({YEAR}|current)/{ID}/links

VERBO: GET

Restituisce i link relativi al corso {ID}, estratti dall'oggetto restituito dalla GET su courses/({YEAR}|current)/{ID}. L'oggetto restituito dovrà contenere tante chiavi quanti sono i tipi di link previsti dalla specifica:

```
{"homepage":"…", "forum":"…", "elearning":"…", resources:…""}
```

VERBO: PUT

Aggiorna i link relativi al corso {ID} per all'anno accademico {YEAR} o per quello corrente (*current*). Il payload è lo stesso oggetto restituito dalla lettura (GET) tramite la URL courses/({YEAR}|current)/{ID}/links.

Il Client

Si crei un client HTML5/Javascript con il supporto di JQuery per il servizio appena descritto, sotto forma di una semplice *single page application* che utilizza AJAX per richiedere dati al servizio mostrandoli poi opportunamente sulla pagina.

Il client dovrà essere strutturato come segue:

1) Un selettore permetterà di scegliere un anno accademico o l'opzione "corrente".
2) Un selettore permetterà di scegliere una lingua tra italiano e inglese.
3) Sarà visualizzata la lista dei corsi relativi all'anno accademico selezionato dal controllo al punto (1) con nome, codice, settore e docenti. I dettagli saranno mostrati nella lingua selezionata dal controllo al punto (2). *Per evitare eccessivo numero di chiamate al servizio, è opportuno paginare questa lista e caricare le informazioni su un corso (mettendole poi in una cache) solo quando viene richiesta la corrispondente pagina.*
4) Selezionando un corso dalla lista al punto (3), in un'area specifica della pagina (o in un *dialog box modale*, ci sono ottimi plugin di JQuery per gestirli) verrà mostrato il sillabo corrispondente, sempre nella lingua selezionata dal controllo al punto (2).

Questa è una specifica di base. Potete aggiungere altre funzionalità a vostra scelta che sfruttino le altre API del servizio, se volete, aumentando così il valore del progetto.

Nota: normalmente, una SPA di una certa complessità andrebbe realizzata sfruttando framework di più alto livello come AngularJS. Tuttavia, per applicazioni semplici, l'uso di JQuery a basso livello risulta essere più pratico, in quanto l'uso di framework come AngularJS aggiunge un notevole overhead al progetto.

Tecnologie da utilizzare

- Nel realizzare le pagine web, se necessarie, si utilizzi HTML5 (*con sintassi XML o Polyglot*).
- Per la programmazione lato *client*, oltre a JQuery, si possono liberamente includere nel progetto altre librerie e plugin, a patto che nella relazione siano citate e descritte.
- Per la programmazione lato *server,* il servizio potrà essere realizzato in *JAVA* o *PHP*. Sviluppando in Java, si utilizzi un'implementazione di *JAX-RS* (come Jersey o quella presente in *Spring*). In PHP, invece, è possibile usare un framework come *Slim* o un altro a scelta.
- Se il progetto/servizio prevede funzionalità ORM, in Java si utilizzi *JPA* con un provider a scelta, mentre in PHP si utilizzi *Doctrine*.

Svolgimento e Documentazione del Progetto

Le specifiche fornite potrebbero non risultare esaustive o completamente definite. Ogni funzionalità aggiunta o raffinata, anche tramite l'interazione con il committente o con gli utenti finali del sito, sarà adeguatamente valutata. Tutte le scelte progettuali vanno comunque discusse e motivate.

Il progetto, svolto secondo le linee guida date dalle specifiche, dovrà essere consegnato nella forma di una o più web application completamente funzionanti, i cui contenuti e le cui caratteristiche saranno valutati in sede d'esame. Le parti della specifica marcate come *opzionali*, se omesse, non renderanno il progetto insufficiente ma non gli permetteranno comunque di raggiungere il massimo dei voti. Nel caso si decida di realizzarle, non sarà necessario che siano perfette o complete, ma che dimostrino chiaramente il vostro impegno nell'affrontare una tematica avanzata.

La documentazione (**in formato elettronico**) che accompagna il progetto **deve** contenere almeno le seguenti informazioni:

- Indicazione delle dipendenze software (di quali librerie avete bisogno dal lato server e client?).
- Indicazione delle funzionalità realizzate e di quelle eventualmente non realizzate. Descrizione dettagliata delle eventuali funzionalità extra o opzionali inserite nel progetto.
- Schema relazionale della base di dati (se presente).

Nel caso di gruppi di lavoro composti da più componenti, *il contributo effettivo offerto da ciascun componente* alla realizzazione finale **deve** essere descritto nella documentazione. In sede di esame, i responsabili potranno essere chiamati a riferire sugli aspetti loro delegati.

Valutazione del Progetto

Nel valutare il progetto consegnato saranno prese in considerazione le seguenti caratteristiche (in ordine di importanza):

1. Rispetto delle specifiche.
2. Correttezza tecnica.
3. Uso appropriato delle tecnologie di base descritte nel corso.

A questa valutazione si aggiungerà quella generale derivata dalla discussione del progetto in sede d'esame.

Ulteriori Informazioni

Questa specifica è disponibile in formato PDF sulla pagina web del corso di Sviluppo Web avanzato, all'indirizzo http://www.di.univaq.it/gdellape. Ulteriori informazioni e chiarimenti sulle specifiche possono essere richiesti direttamente via email all'indirizzo giuseppe.dellapenna@univaq.it.

Si ricorda che i progetti vanno svolti in *piccoli* gruppi (tre persone è il numero consigliato). Eccezioni a questa regola andranno concordate direttamente col docente.

An English translation of this project specification is also available on the teacher's website (http://www.di.univaq.it/gdellape). To access it, open the didactic page for this course, select "English version", and go to the projects listing at the page bottom.