Tecnologie Web T
10 Luglio 2015 – Compito

Tempo a disposizione: 3 ore

La soluzione comprende la **consegna elettronica** dei seguenti file mediante l'apposito applicativo Web **esamix** (**http://esamix.labx**):

Dispatcher.zip	file zip contenente il sorgente java/class e pagine Web per punto 1
Recursion.zip	file zip contenente il sorgente java/class e pagine Web per punto 2
Concorso.zip	file zip contenente il sorgente java/class e file xml per punto 3

Ogni file .zip consegnato DEVE CONTENERE TUTTI e SOLI i file creati/modificati e/o ritenuti importanti in generale ai fini della valutazione (ad esempio, descrittori, risorse statiche o dinamiche, codice Java e relativi .class, ecc.) e NON dell'intero progetto

N.B. Per superare la prova scritta di laboratorio ed essere ammessi all'orale, è necessario totalizzare almeno 18 punti (su un totale disponibile di 33), equamente distribuiti sui tre esercizi, ovvero almeno 6 punti sul primo esercizio, 6 punti sul secondo esercizio e 6 punti sul terzo esercizio

---

Studenti in debito di Tecnologie Web L-A
Viene richiesto lo svolgimento dei soli esercizi 1 (17 punti) e 2 (16 punti). Tempo a disposizione: 2 ore.
I 18 punti necessari per l'ammissione all'orale sono così distribuiti: almeno 10 punti sul primo esercizio e almeno 8 punti sul secondo

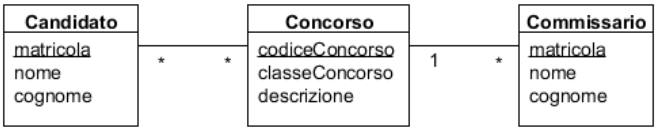
---

ESERCIZIO 1 (11 punti)
Si realizzi un'**applicazione Web** che permetta di inserire una coppia di operandi e di vedere eseguita una operazione scelta casualmente da una servlet dispatcher; l'applicazione deve essere in grado di visualizzare tutte le operazioni già svolte in precedenza per uno stesso utente e deve essere basata principalmente su **tecnologie Java servlet, JSP e JSON**.
In particolare, l'applicazione dovrà essere costituita da una pagina Javascript **accesso** che controlli che i due operandi inseriti siano numeri interi (positivi o negativi, ma non nulli); superati i controlli, tali parametri devono essere trasferiti verso una servlet **dispatcher** codificati in JSON. **dispatcher** deve effettuare una estrazione casuale e sulla base del risultato dell'estrazione passare gli operandi a una JSP per il calcolo della somma oppure a una JSP per il calcolo della divisione. Le due pagine JSP devono effettuare il calcolo associato e restituire direttamente all'utente il valore del risultato, in aggiunta alla storia delle interazioni precedenti di tale utente con **entrambe le pagine JSP**; tali dati relativi alle interazioni precedenti devono essere codificati e restituiti dalle pagine JSP in formato JSON.
Inoltre, l'applicazione deve includere una pagina **statistiche.jsp**, accessibile solo da un amministratore (username=admin; password=admin), capace di mostrare quante operazioni di somma e quante operazioni di divisione sono state svolte negli ultimi 30 minuti.

Tecnologie Web T
10 Luglio 2015 – Compito

ESERCIZIO 2 (11 punti)
Si realizzi un'**applicazione Web** che permetta di calcolare in modo ricorsivo la moltiplicazione fra due numeri naturali, basata principalmente su **tecnologie Java servlet, Javascript** e **AJAX**.
In particolare, l'applicazione dovrà essere costituita da una pagina Javascript **home** che controlli che i due operandi inseriti (x e y) siano numeri naturali. Una volta effettuati i controlli, **home** deve invocare in modo concorrente, tramite richieste AJAX parallele, una servlet **add1** con parametri di ingresso x e trunc(y/2) e una servlet **add2** con parametri di ingresso x e y-trunc(y/2). Ad esempio, se x=15 e y=9, add1 riceverà come parametri di ingresso (15, 4) mentre add2 riceverà (15, 5).
Le due servlet **add1** e **add2** si devono occupare di calcolare il risultato della loro moltiplicazione parziale tramite somma e invocazione ricorsiva, sfruttando l'idea algoritmica che mult(a, b) = (a+b) + add(a, b-1). Una volta che l'ultima chiamata ricorsiva ha determinato la risposta, sia **add1** che **add2** devono restituire il loro risultato alla pagina Javascript **home**; **home** dovrà infine visualizzare "In attesa di risultato" fino a che entrambi i risultati parziali non sono ricevuti, poi il risultato complessivo derivante dalla somma dei due risultati parziali.
Il candidato illustri tramite commento nel file sorgente quale modello di esecuzione concorrente delle servlet è il più appropriato per questa applicazione Web, motivi tale scelta e usi tale modello, eventualmente anche se deprecato, nella soluzione proposta.

ESERCIZIO 3 (11 punti)
Partendo dalla realtà illustrata nel **diagramma UML** di seguito riportato, si fornisca una soluzione alla gestione della persistenza basata su **Hibernate** in grado di "mappare" efficientemente il modello di dominio rappresentato dai **JavaBean del diagramma UML** con le corrispondenti **tabelle relazionali derivate dalla progettazione logica del diagramma** stesso.



Nel dettaglio, dopo aver **creato da applicazione Java** gli **schemi delle tabelle** all'interno del proprio schema nel database **TW_STUD** di **DB2** (esplicitando tutti i **vincoli** opportuni), implementato i **JavaBean**, definito i **file XML di mapping** e il **file XML di properties**, si richiede la realizzazione di una classe di prova facente uso delle **API Hibernate** in grado di:

- inserire tre o piu' tuple nelle tabelle d'interesse;
- determinare: (i) numero di candidati che hanno partecipato al concorso nella cui commissione è stato presente il commissario con matricola "X0034" (restituire anche la classe di concorso); (ii) nome e cognome dei candidati che hanno partecipato a più concorsi;
- produrre una stampa del risultato dei punti precedenti sul file **concorso.txt**

il tutto, mediante uso di **ID surrogati** e opportuna **gestione delle transazioni**.

N.B. La soluzione **deve sfruttare i mapping M-N** e **1-N** specificati nel diagramma UML. **Ogni ulteriore scelta da parte dello studente deve essere opportunamente giustificata** con commenti nel codice.