Tecnologie Web T
14 Luglio 2017 – Compito

Tempo a disposizione: 3 ore

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

Comitiva.zip	file zip contenente il sorgente java/class e pagine Web per punto 1
Uppercase.zip	file zip contenente il sorgente java/class e pagine Web per punto 2
Hospital.zip	file zip contenente il sorgente java/class, file XML e txt 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, principalmente basata su tecnologie JSP e servlet, per ***l'acquisto di biglietti per Disneyland da parte di un gruppo di turisti*** (acquisto in comitiva).
L'applicazione deve avere una pagina iniziale **start.html** presso cui sia possibile autenticarsi tramite username e password; dopo l'autenticazione, ogni utente sarà riconosciuto come appartenente a un gruppo; quindi, per semplicità, si supponga già nota server-side la lista degli utenti registrati, i loro username-password, e i loro gruppi di appartenenza. Dopo l'autenticazione, l'utente potrà aggiungere il numero di biglietti che vuole acquistare al suo carrello personale e, tramite un pulsante, chiedere di completare l'acquisto.
Nota bene: visto che i biglietti saranno usati per una *visita di comitiva dell'intero gruppo*, la vera finalizzazione dell'acquisto avverrà solo DOPO che tutti gli utenti del gruppo avranno chiesto di completare il loro acquisto. Nel caso in cui anche un singolo utente di un gruppo non richieda il completamento del suo acquisto prima della terminazione della sua sessione, l'intero acquisto di gruppo deve essere annullato e tutti gli utenti del gruppo devono ricominciare la loro operazione d'acquisto dall'inizio.
Infine, l'amministratore di sistema (username=admin; password=admin) deve avere la possibilità in ogni istante di forzare il completamento o l'annullamento di qualunque acquisto di comitiva.

Tecnologie Web T
14 Luglio 2017 – Compito

ESERCIZIO 2 (11 punti)

Si realizzi un'applicazione Web, principalmente basata su tecnologia Javascript, per la trasformazione ***di caratteri minuscoli in maiuscoli*** in un testo inviato dall'utente.

L'applicazione deve avere una pagina iniziale **start.html** tramite la quale viene richiesto all'utente di inserire in una casella un input testuale costituito da un numero di caratteri compreso fra 1000 e 2000; inoltre, la pagina **start.html** richiederà l'inserimento di una stringa che sarà il nome del file server-side su cui verrà salvato il risultato del processamento da parte dell'applicazione Web.

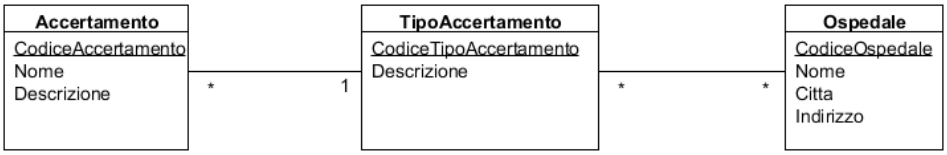
Una volta inseriti i due dati di input descritti sopra, l'utente potrà premere un pulsante per l'invio della richiesta al server. Lato server, il processamento deve essere eseguito in modo concorrente da due servlet: la prima servlet deve trasformare i caratteri minuscoli in maiuscoli sulla prima metà del testo di input; la seconda servlet deve svolgere lo stesso compito in concorrenza sulla seconda metà; al termine dovrà esistere un unico file, col nome specificato dall'utente, che conterrà l'intero risultato della trasformazione.

Inoltre, dovrà essere restituito all'utente, tramite JSON, il numero complessivo di caratteri trasformati da minuscoli in maiuscoli.

Si giustifichi nel codice sorgente tramite commento la scelta effettuata in termini di modello concorrente di esecuzione (ad esempio, quanti thread? Quanti oggetti? Possibili problemi di accesso concorrente a che cosa e quali soluzioni adottate? Con quali overhead?).

ESERCIZIO 3 (11 punti)

Partendo dalla realtà illustrata nel **diagramma UML** di seguito riportato, si fornisca una soluzione alla gestione della persistenza basata su tecnologia **Hibernate** in grado di "mappare" efficientemente, e con uso di ID surrogati, 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 derivati dal diagramma UML), implementato i **JavaBean**, definiti 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 due o più tuple nelle tabelle di interesse;
- restituire **i)** per l'accertamento di codice "*C010*", l'elenco degli ospedali (modellati come triple "*Nome, Citta, Indirizzo*") in cui è possibile eseguire tale accertamento; **ii)** per l'ospedale "*San Camillo*" di *Bologna*, il numero di accertamenti erogabili per ogni tipologia di accertamento supportata dalla struttura;
- produrre una stampa opportunamente formattata dei risultati delle query al punto precedente sul file **Hospital.txt**;

il tutto, mediante opportuna gestione delle **transazioni**.

N.B. **La soluzione deve sfruttare i mapping 1-N e M-N specificati nel diagramma UML**. Ogni ulteriore scelta fatta dallo studente deve essere opportunamente giustificata mediante commenti nel codice.