Tecnologie Web T
02 Luglio 2018 – Compito B

Tempo a disposizione: 3 ore

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

Conteggio.zip	file zip contenente il sorgente java/class e pagine Web per punto 1
Integrale.zip	file zip contenente il sorgente java/class e pagine Web per punto 2
Spesa.zip	file zip contenente il sorgente java/class e file xml/xsd 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, Java servlet, Javascript e AJAX, per il conteggio di caratteri maiuscoli in file mantenuti server-side.
L'applicazione deve consentire a utenti NON autenticati (non richiesta fase di autenticazione tramite username e password) di accedere a una pagina **home.jsp** che determinerà dinamicamente la lista dei file contenuti nella cartella *dir* lato servitore (*dir* è un parametro di inizializzazione della applicazione Web definito tramite file XML di deployment). Richieste successive dello stesso utente NON devono produrre la ri-determinazione dinamica della lista, che può essere considerata invariabile all'interno di una sessione di interazione.
L'utente deve poi poter selezionare 3 file dalla lista; la selezione del terzo file deve produrre automaticamente l'invocazione del conteggio dei caratteri maiuscoli nei 3 file. Il conteggio deve essere effettuato in concorrenza da due "contatori" differenti, in competizione l'uno con l'altro: i) una servlet e ii) un EJB session bean. Ciascuno dei due contatori deve misurare quanto tempo impiega ad effettuare il conteggio; la risposta restituita all'utente deve includere sia il risultato del conteggio che il tempo impiegato.
Ci possono essere operazioni concorrenti che possono portare a inconsistenze? In quali casi? Nel caso di risposta positiva, descrivere quali sono esattamente i rischi di inconsistenza e in quali condizioni si possono verificare, come commento nel file sorgente.

ESERCIZIO 2 (11 punti)

Si realizzi un'applicazione Web, basata su tecnologie Javascript, AJAX, JSON e Java servlet per la realizzazione di un servizio altamente concorrente di calcolo integrale.
L'applicazione deve consentire a utenti NON autenticati (non richiesta fase di autenticazione tramite username e password) di accedere a una pagina **home.html** in cui è possibile specificare gli estremi di integrazione (x, y) di interesse.
Poi deve invocare concorrentemente 4 servlet che si occuperanno in parallelo di calcolare l'integrale della funzione *f* (per semplicità si assuma che la funzione sia fissata e predefinita, direttamente embedded nel codice sorgente), la prima servlet sul primo quarto dell'intervallo di integrazione (x, y), la seconda sul secondo quarto, e così via. Sia gli estremi di integrazione passati alle servlet sia il risultato parziale di calcolo integrale restituito come risposta devono essere trasferiti in formato JSON. Inoltre, si preveda che, lato cliente, venga calcolato il valore finale complessivo dell'integrale, dato ovviamente dalla somma dei 4 risultati parziali.
Infine, si consideri che, per non sovraccaricare il sistema, ogni utente abbia la possibilità di invocare NON più di 5 calcoli integrali all'interno di una sessione di interazione; la sessione di interazione avrà durata massima di 60 minuti e dovrà esserne forzata la terminazione dopo 10 minuti di inattività utente.

ESERCIZIO 3 (11 punti)

Si progetti una grammatica **XML Schema**, e un suo **documento XML** di esempio, per la modellazione delle informazioni relative a "**Ordine spesa Supermercato24**", nel rispetto delle seguenti specifiche:

- La catena Supermercato24 comprende i migliori supermercati di **spesa on-line**, quali, ad esempio, *Esselunga*, *Carrefour*, *Conad*, etc., presso cui l'utente può scegliere di eseguire **un ordine di prodotti alimentari freschi** (si assuma la cardinalità della lista di supermercati aderenti uguale a *N*).
  L'utente può inoltre specificare **l'orario di consegna** della spesa in *fasce orarie di un'ora* e scegliere **come pagare** (pagamento anticipato *on-line* o *in contanti* alla consegna).
- Ogni supermercato della lista sopra descritta espone i propri prodotti alimentari freschi divisi per **categoria**; quattro tipologie nello specifico: "frutta e verdura" (*FV*), "pane e pasta" (*PP*), "latticini" (*L*) e "carne e pesce" (*CP*).
- Ogni prodotto alimentare di una certa tipologia è caratterizzato da una **marca** (es. *Granarolo*), da una **descrizione** (es. *Latte intero*) e dal relativo **prezzo** (es. *€ 1,50*).

Si realizzi quindi l'**applicazione Java "Supermercato24"** che, facendo uso dei parser **SAX** e **DOM e del documento XML di esempio**, esponga i metodi `getCostoTotaleSpesa()` e `getCountCategoria()`, unitamente a un `main` di prova, in grado di calcolare:

1. il **costo totale** da pagare relativo all'ordine eseguito e
2. il **numero di prodotti per ogni categoria** di cui è costituito l'ordine.