

**Tecnologie Web T**  
**23 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>):

<b>ShopB.zip</b>	file zip contenente il sorgente java/class e pagine Web per punto 1
<b>ConteggioB.zip</b>	file zip contenente il sorgente java/class e pagine Web per punto 2
<b>OrdineB.zip</b>	file zip contenente il sorgente java/class 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 Java servlet, per la realizzazione di un semplice negozio online.

L'applicazione deve consentire a utenti NON autenticati di accedere a una pagina iniziale generata da una servlet **catalogo** che lista tutti i prodotti disponibili alla vendita online. Ogni prodotto è contraddistinto da codice identificativo univoco, descrizione testuale, prezzo e numero di unità ancora disponibili alla vendita; questi dati devono essere prelevati da memoria persistente (database esistente) e inviati lato cliente in formato JSON. Questa pagina iniziale deve inoltre dare la possibilità all'utente di passare a una pagina di carrello.

La pagina di carrello è semplicemente costituita da un form in cui l'utente può inserire codice identificativo di un prodotto e numero di unità da acquistare; alla pressione del bottone "Metti in carrello" deve essere invocata una seconda servlet **carrello** che popolerà conseguentemente il carrello, segnalando possibili fallimenti (ad esempio nel caso in cui il codice identificativo sia errato).

Infine, si preveda che alla ricezione di una richiesta a **carrello** dopo  $x$  minuti dall'inizio della sessione utente (dove  $x$  è parametro di configurazione dell'applicazione a deployment time) l'applicazione renda definitivo l'acquisto contenuto nel carrello, segnalando anche in questo caso possibili fallimenti. Quali possibili situazioni di fallimento e perché? Si commenti direttamente nel file sorgente.

**ESERCIZIO 2 (11 punti)**

Si realizzi un'applicazione Web, principalmente basata su tecnologie JSP, Java servlet, JavaScript e AJAX, per il conteggio di caratteri di file server-side appartenenti a un range specificato dinamicamente.

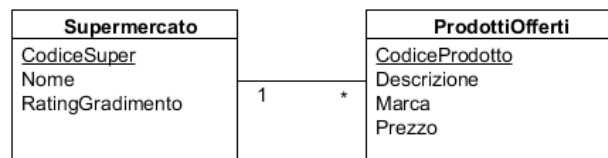
L'applicazione deve consentire a utenti NON autenticati (non richiesta fase di autenticazione tramite username e password) di accedere a una pagina **home.jsp** tramite la quale specificare il range di caratteri da contare (come estremi di un intervallo, ad esempio da "a" a "s"). Richieste successive dello stesso utente NON devono chiedere nuovamente di specificare tale range: l'intervallo di caratteri da contare deve essere considerato invariabile all'interno di una sessione di interazione.

L'utente deve poi poter selezionare un file presente nella cartella *dir* indicata come parametro di configurazione a deployment time dell'applicazione; la selezione del file deve produrre automaticamente l'invocazione del conteggio dei caratteri appartenenti all'intervallo prescelto. 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, trasmessi in formato XML.

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 3 (11 punti)**

Partendo dalla realtà illustrata nel **diagramma UML** di seguito riportato, si fornisca una soluzione alla gestione della persistenza basata su metodologia **Forza Bruta** in grado di **"mappare"** il modello di dominio rappresentato dai **JavaBean** del diagramma UML con le corrispondenti **tabelle relazionali** derivata dalla **progettazione logica** del diagramma stesso.



**N.B.** Relativamente allo schema UML: Nome è **chiave** per la tabella Supermercato;

Nel dettaglio, dopo aver creato da applicazione Java lo schema delle tabelle nel proprio schema nel database **TW\_STUD** di **DB2** (esplicitando tutti i vincoli derivati dal diagramma UML) e implementato **JavaBean e metodi necessari per la realizzazione delle operazioni CRUD**, si richiede la realizzazione dei **metodi**:

- (i) boolean ProdottoOfferto (String NomeSuper, String Descrizione, String Marca) che, dato il nome di un supermercato, la descrizione di un prodotto e la relativa marca, verifica se tale prodotto è offerto dal supermercato o meno.
- (ii) String NomeSuper(String Descrizione, String Marca) che, data la descrizione di un prodotto e la relativa marca, restituisce il nome del supermercato in cui tale prodotto costa meno.

Si richiede quindi di realizzare una **classe di prova** in grado di:

- inserire alcuni supermercati e alcuni prodotti offerti dagli stessi nelle tabelle corrispondenti;
- utilizzare correttamente i metodi ProdottoOfferto() e NomeSuper() producendo una stampa completa sul file **Ordine.txt** dei risultati prodotti rispetto all'istanza creata al punto precedente.