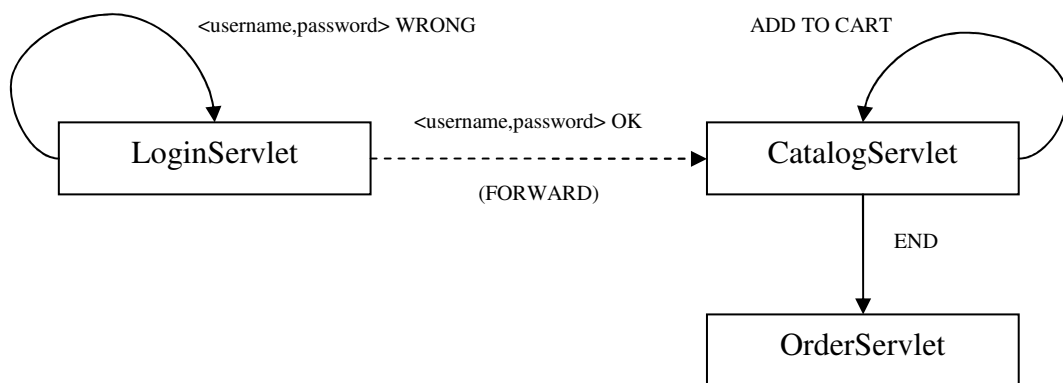


## Laboratorio 4. Servlet (9/11/2007)

Questo è l'inizio di una serie di laboratori dedicati alle tecnologie più importanti della piattaforma J2EE (*Java 2 Enterprise Edition*), uno standard per lo sviluppo di applicazioni Web multilivello. Nel corso di questi laboratori verrà ipotizzata la conoscenza dei concetti teorici relativi a Web Application (*Servlet e JSP*), Enterprise Application ed Enterprise Java Beans (*EJB*) nonché relativi all'accesso ad un database sfruttando l'interfaccia *JDBC*.

Con questo laboratorio svilupperemo una semplice Web Application che implementa, utilizzando le Servlet, la gestione di un login utente e di un banale carrello della spesa.



Al primo accesso dovrà essere presentata una form di login (con username e password). In caso di login con successo sarà mostrata una lista di prodotti (con i campi codice, descrizione, prezzo) dalla quale, tramite un link per ogni elemento della lista, sarà possibile aggiungere un prodotto al carrello. Nella stessa pagina saranno visualizzati nome e cognome dell'utente che ha effettuato il login, il numero dei prodotti nel carrello e un link per il termine della selezione dei prodotti che avrà l'effetto di visualizzare una pagina riassuntiva con i codici dei prodotti scelti e le rispettive quantità. Poiché sarà possibile aggiungere lo stesso prodotto più di una volta al carrello, questo dovrà tener conto della quantità.

In caso di login fallito verrà presentata la stessa pagina di login con un messaggio di errore.

Il login e il recupero della lista dei prodotti verranno effettuati accedendo al database tramite la libreria di supporto (*dbmanager.jar*). Questa libreria contiene alcune classi (package *it.unipi.ing.sari.db*) descritte di seguito.

- **User**: bean contenente nome, cognome, username e password di un utente.
- **CatalogItem**: bean contenente codice, descrizione e prezzo di un prodotto.
- **DBManager**: classe con la seguente interfaccia:
  - `DBManager(String dbDriverName, String dbUrl, String username, String password) throws DBInitException;`  
Costruttore della classe. I parametri indicano i dati per aprire una connessione al database.
  - `void DBRelease() throws SQLException;`  
Rilascia la connessione al database.
  - `User verifyUser(String username, String password) throws SQLException;`

Verifica l'esistenza nel database della coppia username, password. Se esiste ritorna una istanza della classe *User* altrimenti null.

- o *CartItem[] getCatalog() throws SQLException;*

Preleva il catalogo da database. Ritorna un array di *CartItem*. L'array ha lunghezza zero se non ci sono prodotti nel catalogo.

## Strumenti di lavoro

Per questo laboratorio sono necessari un ambiente di sviluppo Java, un Web Server che contenga un Servlet Container ed un DBMS con supporto JDBC. L'ambiente di sviluppo Java può essere sia il semplice compilatore fornito con la distribuzione J2SE SDK (*Java 2 Standard Edition SDK*) sia un ambiente IDE come NetBeans. Per quanto riguarda il Servlet Container, ne esistono molti sia gratuiti che commerciali ma è sufficiente l'uso di Apache Tomcat. Come DBMS server è sufficiente la versione di PointBase fornita con il J2EE SDK (*Java 2 Enterprise Edition SDK*).

**NOTA:** sembra che con la configurazione attuale del software utilizzato, il database PointBase possa dare dei problemi, legati al fatto che la versione di PointBase in SJSAS 8.1 non è più supportata dalla Sun. Si consiglia di provare prima a svolgere l'esercizio in locale come da dispensa e, nel caso di problemi di configurazione, di utilizzare un altro database (es. MySQL).

## Struttura e Configurazione

La Web Application sarà costituita da tre Servlet: *LoginServlet*, *CatalogServlet* e *OrderServlet*.

Il carrello della spesa sarà mantenuto in un attributo di sessione e sarà implementato con la classe *java.util.Hashtable* utilizzando come chiave il codice prodotto (*Integer*) e come valore la quantità associata a quel prodotto (*Integer*). Il carrello verrà creato vuoto dopo l'operazione di login.

I dati dell'utente che ha effettuato il login verranno mantenuti in un attributo di sessione con la memorizzazione di un bean di tipo *it.unipi.ing.sari.db.User*. Il numero totale dei prodotti verrà mantenuto anch'esso in un attributo di sessione con un oggetto di tipo *Integer*.

La *LoginServlet* sarà la prima ad essere invocata e si occuperà della visualizzazione della pagina di login e della verifica dei dati inseriti. Inoltre, in caso di login con successo, predisporrà le strutture dati per la gestione del carrello della spesa ed effettuerà il forward alla *CatalogServlet*.

La *CatalogServlet* si occuperà della visualizzazione della lista dei prodotti e dell'aggiunta di prodotti al carrello. Ogni volta che l'utente aggiungerà un prodotto al carrello questa Servlet si preoccuperà di aggiornare le strutture dati presenti in sessione.

La *OrderServlet* avrà lo scopo di visualizzare la lista dei prodotti presenti nel carrello.

## Suggerimenti

1. Si consiglia di provare ad utilizzare NetBeans per lo sviluppo dell'applicazione e di prendere confidenza con gli strumenti di debugging. La versione finale da distribuire può essere generata con il comando Build, che crea un file WAR nella directory dist.
2. I parametri di connessione al DB possono essere inseriti come context parameters in *web.xml*. In NetBeans, questa azione può essere effettuata direttamente dalla scheda General dell'interfaccia grafica del file di configurazione.

3. La libreria di supporto `dbmanager.jar` e la libreria dei driver JDBC (`pbclient.jar` o il MySQL Connector-J, a seconda del DB utilizzato) devono essere poste nella directory `WEB-INF/lib` in modo da essere visibili dalla Web Application.
4. Per rispettare l'atomicità delle connessioni al DB, le servlet devono implementare le interfacce `SingleThreadModel`.
5. L'esercizio comprende molti aspetti di programmazione servlet: reindirizzamento, persistenza a livello di sessione, ecc. Si consiglia di consultare le slide.
6. Si consiglia di utilizzare una `HashTable` con chiave codice prodotto e valore quantità del prodotto per mantenersi in sessione lo stato attuale dell'ordine.