Applicazioni Internet 2016-17

Esercitazione 1

Un'applicazione web permette l'acquisto online dei biglietti di un sistema di trasporto metropolitano. Esistono diverse tipologie di biglietto (corsa singola urbana, corsa singola suburbana, biglietto giornaliero, abbonamento settimanale, abbonamento mensile, …), ciascuna caratterizzata da un proprio prezzo, ed è possibile comprare più titoli per ciascuna categoria.

Interagendo con l'applicazione, l'utente può esaminare le tipologie di biglietto, aggiungere uno o più titoli al proprio carrello della spesa, ispezionare il contenuto del carrello, modificare il contenuto del carrello, accedere alla piattaforma con le proprie credenziali (login), uscire dalla piattaforma rimuovendo tutte le proprie informazioni (logout), procedere al pagamento (check out), confermare il pagamento. Tutte le operazioni, escluse le ultime due, possono essere effettuate in modo anonimo, mentre per procedere al pagamento occorre essere autenticati. L'autenticazione può, in ogni caso, essere eseguita anche prima di procedere al pagamento, senza che questo precluda il funzionamento del sistema.

Si implementi tale applicazione utilizzando la tecnologia JavaEE, seguendo la traccia seguente (può essere utile, in caso di difficoltà, procedere in modo iterativo: considerare inizialmente un sotto-insieme del problema complessivo ed eseguire i punti di seguito esposti, fino ad avere un sistema minimale ma funzionante; poi ripeterne lo svolgimento considerando tutti i vincoli dello scenario proposto).

1. Partendo dalla descrizione precedente, si provino ad abbozzare su carta una serie di schermate corrispondenti ai diversi stati in cui l'applicazione si può trovare.
   (Ciascuna di queste schermate dovrà, in seguito, essere trasformata in una pagina JSP che – ricavando i dati concreti da attributi presenti nella richiesta o nella sessione – provvederà a generare la vista definitiva inviata al browser).

2. Ad ogni possibile azione dell'utente, si associ una URL opportuna, avendo cura di associare alle azioni da eseguire previa autenticazione un prefisso ("/private/") che le renda distinguibili dalle altre.
   Si distinguano le azioni "safe", implementabili tramite metodo GET, e le azioni "unsafe", da implementarsi tramite metodo POST.
   Si decida come chiamare gli eventuali parametri associati a ogni azione.
   Si colleghino graficamente tra loro le varie schermate identificate al punto 1, etichettando ciascun collegamento con la relativa azione.

3. Dall'analisi del testo, emergono tre tipologie di funzionalità: la gestione del login, la gestione del carrello, la gestione del pagamento. Si dichiarino tre interfacce (LoginService, CartService, PaymentService) che racchiudano la definizione di opportuni metodi che riassumano la logica di business descritta (es.: login(…), logout(), getUserName() / add(…), remove(…), getItems(), getTotal(),…).

4. Si creino tre classi concrete che implementano tali interfacce, mantenendo il proprio stato nelle proprie variabili-istanza (questo perché al momento non abbiamo ancora visto come contattare una base dati per rendere persistenti tali informazioni). Tali

oggetti dovranno essere memorizzati all'interno della sessione di ciascun utente, associati ad un'apposita chiave.

5. Si crei un oggetto di tipo WebListener che implementi le interfacce ServletContextListener e HttpSessionListener. Tale oggetto ha due compiti: all'atto della creazione di un nuovo contesto, dovrà memorizzare all'interno di quest'ultimo un array di oggetti che descrivono i diversi tipi di biglietti disponibili (queste informazioni infatti sono statiche e possono essere condivise tra tutti gli utenti della piattaforma); ogni volta che viene creato un nuovo oggetto di tipo HttpSession, dovrà inserire all'interno di esso tre nuove istanze delle classi che implementano i diversi servizi associandole alla chiave scelta (lo stato di questi oggetti rispecchia l'insieme delle azioni via via svolte da ciascun utente, ed è perciò necessario averne tante istanze quante sono le sessioni attive).

6. Si crei, per ogni URL scelta al punto 2, un servlet che gestisca tale operazione. Tale servlet cercherà, all'interno dell'oggetto sessione e/o del contesto applicativo, l'istanza del servizio di cui ha necessità per implementare l'azione richiesta. Dopo aver verificato che gli eventuali parametri richiesti siano presenti e validi, il servlet delegherà all'oggetto di servizio l'esecuzione dell'operazione richiesta e poi inoltrerà la richiesta ricevuta ad una pagina JSP, tra quelle identificate al passo 1, responsabile di visualizzare la rappresentazione dello stato dell'applicazione.
   Non si prenda in considerazione, in questo passo, il vincolo di autenticazione.

7. Si implementi un filtro, da mappare su tutte le URL che iniziano con "/private/", che verifichi se l'oggetto LoginService presente nella sessione indica o meno che c'è uno userName attivo. In caso affermativo, lascia passare la richiesta verso il proprio naturale destinatario, altrimenti reindirizza la richiesta verso la URL di login che presenterà una schermata per l'accesso al sistema. A fronte di tale pagina, l'utente potrà inserire le proprie credenziali (cablate nell'oggetto LoginService) ed inviarle all'applicazione. In caso di validazione positiva, l'utente sarà reindirizzato alla pagina che aveva chiesto in precedenza (come fa a ricordarsela?), altrimenti resterà nella pagina del login.