Università degli Studi di Salerno
Corso di Ingegneria del Software

E-store Libri&Musica

System Design Document

SDD
Versione 0.4

Data ultima modifica: 15/02/2017

Partecipanti:

Nome	Matricola
Fimiani Luca	0512103180

Scritto da: Fimiani Luca

Revision History

Data	Versione	Descrizione	Autore
20/11/2016	0.1	Inizio stesura documento	Fimiani Luca
21/11/2016	0.2	Inserimento punti 2.1,2.2,2.3,2.4,2.5	Fimiani Luca
22/11/2016	0.3	Compleatamento punto 2.2 Inserimento punti 2.6,2.7	Fimiani Luca
23/11/2016	0.4	Inserimento Subsystem services	Fimiani Luca
24/11/2016	0.5	Aggiunta glossario e revisione altri punti	Fimiani Luca
15/02/2017	0.6	Revisione	Fimiani Luca

INDICE DEGLI ARGOMENTI

1.Introduzione

Questo documento descrive gli obiettivi di design del progetto, illustrando la decomposizione del sistema in sottosistemi e le strategie adottate per lo sviluppo.

1.1. Scopo del sistema

L'obiettivo del progetto è quello di sviluppare un sito web che permetta l'acquisto di prodotti online. Come molti siti presenti in rete oggigiorno, il nostro offre una serie di funzionalità sviluppate per rendere semplice e piacevole la navigazione e l'acquisto di prodotti nel negozio virtuale.
Il presente documento propone lo sviluppo di un sito web in grado di rendere semplice e intuitiva la vendita dei prodotti al cliente. Il sito offre due tipi di prodotti , come si evince dal titolo del progetto: libri e cd musicali. Il cliente può accedere al sito attraverso un form di login. Una volta effettuato l'accesso il cliente può facilmente scegliere i prodotti da aggiungere al carrello virtuale prima di procedere successivamente all'acquisto. L'homepage del sito è progettata per consigliare al cliente prodotti che potrebbero essere di suo interesse. Il cliente, inoltre, può scegliere se ricercare un prodotto attraverso la barra di ricerca oppure utilizzando come criterio di selezione la categoria (libri o musica). Il sito è gestito da un amministratore attraverso il proprio account. L'account amministratore ha funzionalità diverse rispetto a quello del cliente. L'admin può inserire , modificare o eliminare prodotti a seconda delle necessità; inoltre può tener sotto controllo gli ordini di tutti i clienti e può visualizzare le informazioni riguardanti gli utenti registrati al sito. Il cliente, infatti, può facilmente registrarsi attraverso una procedura di registrazione che gli consente l'accesso al sito e dunque la possibilità di effettuare ordini.

1.2 Design goals

Gli obiettivi di design sono definiti basandosi sui requisiti funzionali esposti. Il sito presenta un'interfaccia semplice ed intuitiva. La pagina web è stata progettata per essere utilizzabile da qualsiasi tipo di utente. Essa, infatti, è costituita da varie componenti, tra cui una barra di navigazione, all'interno della quale l'utente potrà rilevare tutte le voci relative alla visualizzazione e alla ricerca dei prodotti. All'interno di tale barra, inoltre, sarà presente anche l'icona del carrello virtuale, che, se cliccata, reindirizzerà l'utente all'interno del carrello stesso. Nella sezione sottostante alla barra di ricerca l'utente potrà visualizzare le info inerenti alla sezione in cui egli si trova in quel preciso momento. Eventuali contatti social, recapiti e informazioni supplementari saranno visibili all'interno del "footer", in basso nella pagina. Il sito "E-store Libri & Musica " ha come struttura principale una banca dati che viene continuamente aggiornata ad ogni interazione da parte dell'utente.

Si pone dunque l'obiettivo di rispettare i seguenti criteri di design :

1.2.1 DG_0 Dependability Criteria

Il sito sarà in grado di garantire il corretto svolgimento delle proprie funzioni, gestendo la maggior parte degli errori logici che comunemente vengono causati da una errata interazione da parte dell'utente. Il sistema si propone di rispettare i seguenti requisiti di affidabilità :

- **DG_01 Robustezza :** non verranno compromessi i dati contenuti nel database. Nel caso in cui l'utente sottometta dati errati durante l'interazione con il sito, questo notificherà l'errata compilazione e non altererà nessun dato all'interno del database relazionale.
- **DG_02 Affidabilità :** l'e-store dovrà fare in modo che tutte le funzionalità vengano svolte in maniera corretta, gestendo tutte le situazioni, proteggendo i dati e producendo l'output atteso.
- **DG_03 Disponibilità :** il sito sarà sempre disponibile, salvo nei periodi di manutenzione.
- **DG_04 Fault tolerance :** il sistema garantirà una tolleranza media agli errori. In caso di errori legati all'inserimento di dati in forma non corretta il sito provvederà a notificare l'errore all'utente.
- **DG_05 Sicurezza :** Il sistema farà in modo che , in base alla tipologia di utente che interagisce con esso, verranno presentate funzionalità diverse. Il cliente del sito, per esempio, non potrà usufruire delle funzionalità messe a disposizione per l'admin.

1.2.2 DG_1 Performance Criteria

Il sito sarà in grado di garantire performance medio/alte. Si cercherà di rendere il sistema più leggero possibile limitando la dimensione delle pagine web e facendo in modo che le operazioni effettuate abbiano un tempo di risposta rispettabile (inferiore ai 12 secondi).
Tale e-commerce si propone dunque di rispettare i seguenti requisiti relativi alla qualità delle prestazioni :

- **DG_1.1 Tempo di risposta :** le operazioni basilari ,come ad esempio la visualizzazione del carrello o dei prodotti, avranno un tempo di risposta breve, a seconda del livello di carico del sistema. Le operazioni che devono effettuare l'accesso e la manipolazione dei dati richiederanno leggermente più tempo.

- **DG_1.2 Throughtput :** il sistema dovrà completare il maggior numero di operazioni nel minore tempo possibile, in modo tale da garantire una maggiore interattività con la varietà degli utenti, nel caso vi sia più di un utente che lo sta usando.

- **DG_1.3 Memoria :** il sistema necessita di una quantità di memoria dipendente dal numero di dati che verrà memorizzato. Il dispendio sarà conteso tra la memorizzazione dei prodotti che verranno inseriti nello store e gli utenti che si registreranno al sito.

1.2.3 DG_2 Maintenance Criteria

Il sito dovrà garantire una discreta manutenibilità. L'e-store dovrà dunque rispettare i seguenti requisiti :

- **DG_2.1 Estensibilità :** il sistema dovrà essere sviluppato in maniera tale da garantire l'aggiunta di nuove funzionalità con modalità semplificata. Esso deve fare in modo che tali aggiunte non comportino modifiche di altre funzionalità.
- **DG_2.2 Modificabilità:** il sistema dovrà poter garantire la modifica di funzionalità già presenti, senza dover compromettere funzionalità non interessate alla modifica che si sta effettuando.
- **DG_2.3 Tracciabilità dei requisiti :** sarà facile rilevare i requisiti funzionali all'interno del codice grazie ad un'accurata documentazione.

1.2.4 DG_3 End-User Criteria

Per quanto riguarda la vista dell'utente il sito dovrà garantire i seguenti requisiti:

- **DG_3.1 Utilità :** grazie ai requisiti funzionali raccolti in fase di analisi, il sistema sarà in grado di supportare in maniera coerente quelle che sono le esigenze degli utenti.
- **DG_3.2 Usabilità :** il sistema dovrà essere semplice ,intuitivo e abbastanza efficiente. E' opportuno , inoltre, che i requisiti non funzionali, descritti nel documento RAD, vengano rispettati come prefissato.

1.2.5 Trade-offs

1.2.5.1 TO_0 Spazio vs efficienza

L'intento è quello di ottimizzare il tempo di risposta del sistema. Per fare ciò si è deciso di concedere più spazio , sia in termine di caching che di ridondanza dei dati.

1.2.5.2 TO_1 Tempo di rilascio vs qualità

L'intento è quello di finire il sistema il prima possibile in modo tale da rispettare la data di rilascio prefissata. Per fare ciò sarà concessa molta flessibilità nel gestire eventuali bug rilevati in extremis e si farà in modo che essi vengano corretti solamente dopo il rilascio del sistema.

1.2.5.3 TO_3 Efficienza vs riusabilità

Si cercherà di fare il possibile per fare in modo che il sistema sia efficiente e al tempo stesso garantisca la riusabilità.

1.3 Riferimenti

- E-store Libri&Musica_RAD
- B. Bruegge, A.H. Dutoit, Object Oriented Software Engineering – Using UML, Patterns and Java, Prentice Hall

1.4 Overview

Il documento è strutturato come segue:

La prima parte è dedicata all'introduzione ed il suo obiettivo è fornire una panoramica dell'SDD, definendo i criteri di affidabilità, performance, manutenibilità e quelli per l'utente finale.
Nella seconda parte viene descritta l'architettura del sistema proposto, in cui viene definita la decomposizione in sottosistemi, il mapping SW/HW, la gestione dei dati persistenti, il controllo degli accessi, il flusso di controllo globale e le condizioni limite.
Nella terza parte vengono descritti i sottosistemi in cui il sito è decomposto e i servizi offerti da essi.
L'ultima parte comprende un glossario contenente i termini di rilievo.

2 Architettura del sistema proposto

2.1 Overview

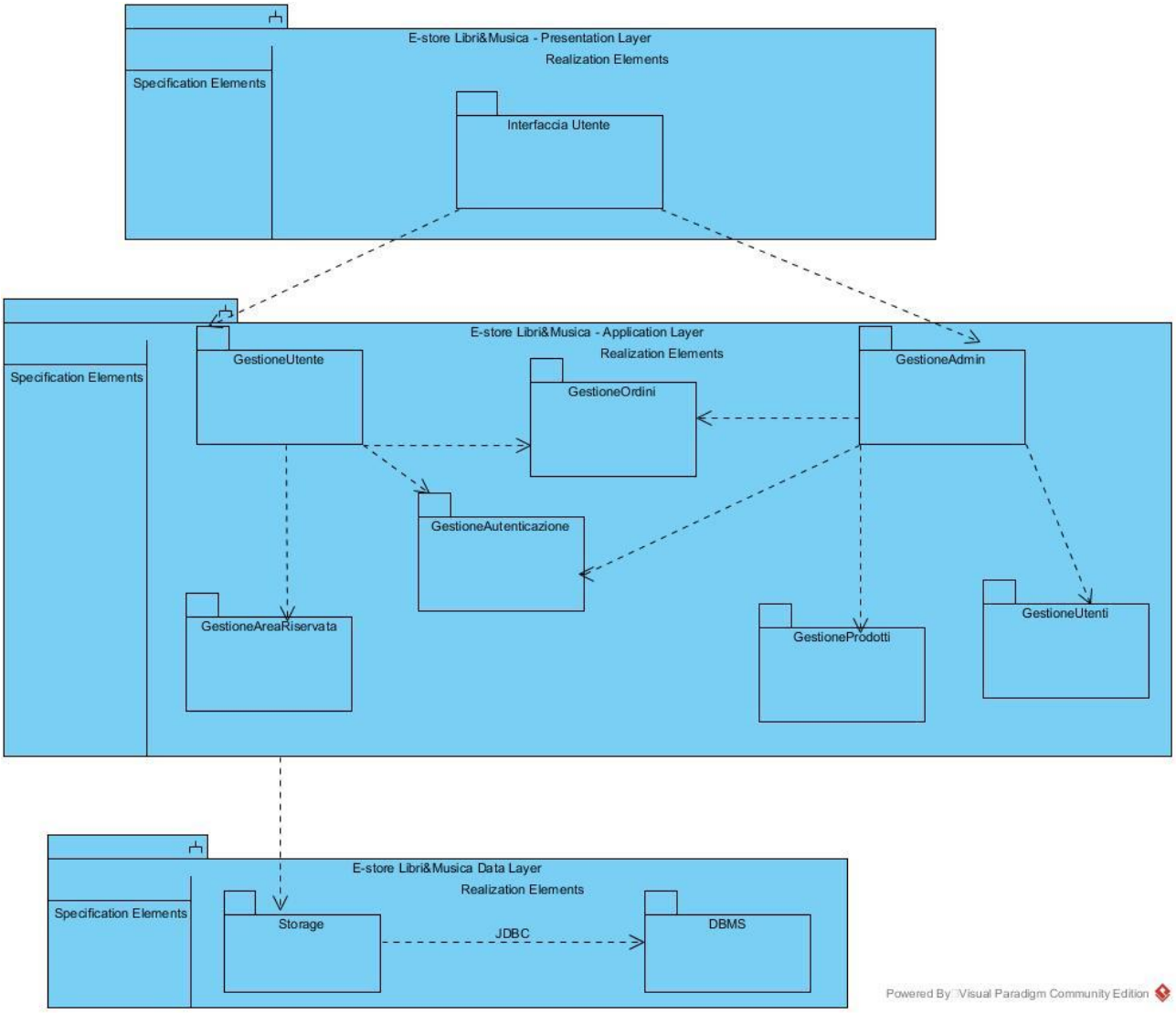
Il modello architetturale che verrà utilizzato sarà il multi-tier con tre strati logici principali. La scelta di tale modello è motivata dalla separazione tra sviluppo ed interfaccia e tra processi funzionali e accesso ai dati che esso offre. Tale modello è comunemente utilizzato nello sviluppo di applicazioni web ed è particolarmente adattabile all'utilizzo di Servlets e Jsp.
I 3 livelli logici definiti sono :

1. **Presentation Layer :** layer che comprende *boundary objects*, *forms*, *web pages* e altri strumenti utili per l'interazione tra il sistema e l'utente.
2. **Application Layer :** layer che serve per la gestione dei processi logici. Gli strumenti utilizzati garantiranno il controllo e la gestione del sito. Tali strumenti sono i *control objects*.
3. **Data Layer :** si occupa della gestione dei dati persistenti appartenenti al sistema. Offre servizi ai moduli sovrastanti che richiedono i dati rendendo invisibili le operazioni di accesso al database relazionale.

2.2 Decomposizione in sottosistemi

La decomposizione serve a rendere il sistema più facile da progettare, riducendo la complessità della progettazione delle funzionalità. Per decomporre il sistema in sottosistemi vi è la necessità

di minimizzare la dipendenza fra i vari sottosistemi e massimizzare la coesione all'interno degli stessi.



Dal diagramma si vede la separazione del sistema in tre layers :

- E-store Libri&Musica Presentation Layer
- E-store Libri&Musica Application Layer
- E-store Libri&Musica Data Layer

L'architettura utilizzata per il flusso dei dati è Client-Server. Gli utenti utilizzano la componente Web Server per interagire con il *Presentation Layer*, il quale rappresenta l'interfaccia di sistema. Questo modulo interagisce con l'*Application Layer*, che a sua volta esegue le operazioni richieste interfacciandosi con il *Data Layer*, al fine di portare a termine le operazioni che necessitano

l'accesso ai dati persistenti.
Il *Data Layer* utilizza le componenti JDBC per interfacciarsi con i servizi del DBMS.

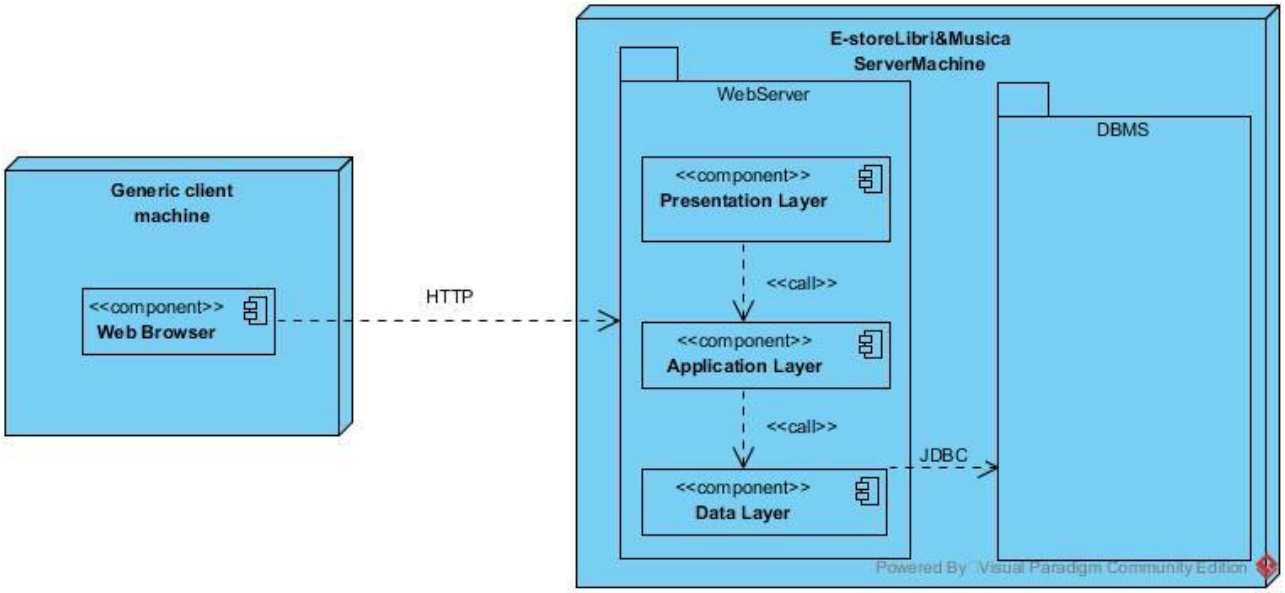
Di seguito sono descritti i moduli definiti dalla decomposizione.

Presentation Layer	
Interfaccia Utente	Modulo che rappresenta l'interfaccia che l'utente utilizza per interagire con il sistema. L'utente mediante il suo utilizzo inoltra richieste al sistema ed attende eventuali risposte.
Application Layer	
GestioneUtente	E' una decomposizione funzionale del sistema. Possiamo infatti decomporre il sistema in GestioneUtente e GestioneAdmin.
GestioneAreaRiservata	E' un modulo che si occupa di gestire l'area riservata dell'utente. Tutte le fuzionalità inerenti alla gestione dell'area riservata sono descritte in tale modulo.
GestioneAutenticazione	E' un modulo che si occupa di controllare gli accessi da parte degli utenti (users & admins), le autorizzazioni e respingere le operazioni non consentite. Raccoglie le informazioni necessarie per permettere di modificare i contenuti da visualizzare e l'interfaccia in base all'utente che ne fa uso.
GestioneOrdini	E' un modulo che si occupa della gestione degli ordini che l'utente andrà a formulare mediante la procedura di inserimento prodotti nel carrello. Tale modulo farà in modo che l'utente riesca a completare l'ordine nel modo corretto
GestioneAdmin	E' una decomposizione funzionale del sistema. Possiamo infatti decomporre il sistema in GestioneUtente e GestioneAdmin.
GestioneAutenticazione	E' un modulo che si occupa di controllare gli accessi da parte degli utenti (users & admins), le autorizzazioni e respingere le operazioni non consentite. Raccoglie le informazioni necessarie per permettere di modificare i contenuti da visualizzare e l'interfaccia in base all'utente che ne fa uso.
GestioneOrdini	E' un modulo che si occupa della gestione degli ordini da parte dell'admin. L'admin, infatti, mediante tale modulo, potrà visualizzare gli ordini di tutti gli utenti e all'occorrenza rimuoverli.
GestioneProdotti	E' un modulo che si occupa di controllare le operazioni che un admin può effettuare con i prodotti. L'admin, infatti, può inserire, modificare ed eliminare i vari prodotti nello store.
GestioneUtenti	E' un modulo che permette all'admin di controllare la gestione degli utenti del sito. Egli mediante tale modulo potrà visualizzare la lista degli utenti registrati al sito e all'occorrenza rimuovere uno o più di essi.

Data Layer	
Storage (ControllerDB)	Modulo (JDBC) che permette di evitare l'accoppiamento fra il modulo del database (gestito in un sistema esterno) e quello di controllo (Application Layer) . Offre un'interfaccia di funzionalità costruita "ad-hoc"per i moduli di controllo dello strato sovrastante.
DBMS	Modulo che interagisce con il database manipolando i dati presenti nel database .

2.3 Hardware / Software Mapping

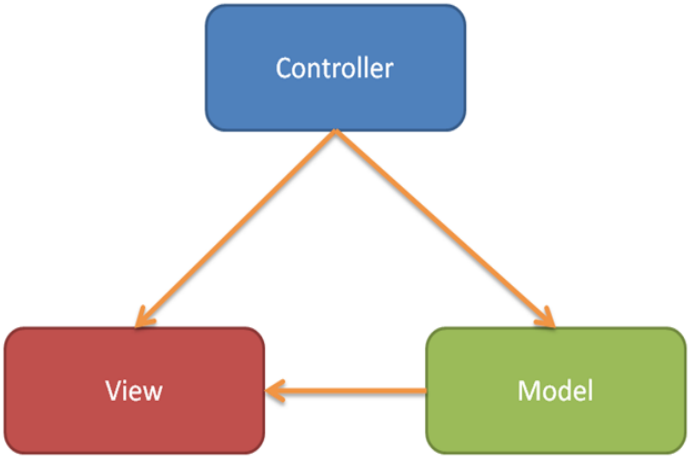
L'architettura può essere scomposta in 2 livelli : un livello *Client* ed un livello *Server*. Di seguito è riportata la distribuzione delle componenti HW e SW sui due nodi.



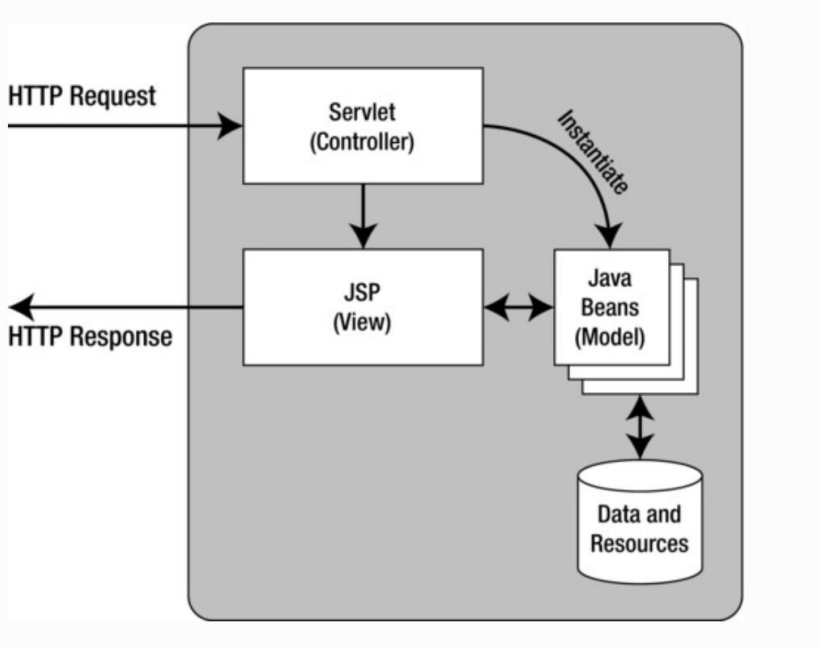
Web Server	Il WebServer utilizzato è apache tomcat.
Presentation Layer	Strato di interfacciamento con l'utente del sito. Esso è implementato nel WebServer, posizionato nella ServerMachine e i contenuti, realizzati con le JSP, saranno visualizzati mediante la componente WebBrowser, collocata nella client machine.
Application Layer	Strato che si occupa della logica di controllo del sito. Il suo scopo principale è quello di mediare fra utente, che interagisce solo con l'interfaccia e il livello di deposito dati, ovvero lo storage.
Data Layer	E' rappresentato da un insieme di funzioni disegnate sulla base delle necessità della logica di business realizzata nei moduli specificati nello strato sovrastante. Questa scelta ha lo scopo di generare una completa coesione fra il database fisico e le funzionalità che necessitano l'accesso ai dati persistenti. La tecnologia di interfacciamento con il database scelta è JDBC(Java DataBase Connectivity).
DBMS	Serve per realizzare e manipolare il database relazionale.

Il nostro sito si basa sull'architettura MVC(Model Control View) . Tale architettura preme per una separazione netta fra logica di business e logica di presentazione.

Paradigma MCV



Architettura MVC 2-tier



• Il Model rappresenta il livello dei dati, incluse operazioni per accesso e modifica. Esso deve notificare le View associate quando viene modificato.

• La View si occupa del rendering dei contenuti del model. Accede ai dati tramite il model e specifica come essi debbano essere presentati.

• Il Controller definisce il comportamento dell'applicazione.

In applicazioni Web conformi al modello 2-tier, le richieste del browser client vengono passate al Controller (implementato dalle *Servlets*). Il Controller si occupa di eseguire logica di business necessaria per ottenere il contenuto da mostrare. Il Controller mette il contenuto nel Model (implementato con *JavaBean* o Plain Old Java Object - *POJO*) in un messaggio e decide a quale View (implementata da *JSP*) passare la richiesta. La View si occupa della visualizzazione del contenuto (ad es. stampa dei valori contenuti in struttura dati o bean, ma anche operazioni più complesse come invocazione metodi per ottenere dati).

2.4 Gestione dati persistenti

La gestione dei dati avviene mediante un database relazionale. I dati sono memorizzati in esso e gestiti mediante un DBMS (DataBase Management System). Le entità in esso rese persistenti raffigurano gli elementi principali del nostro e-commerce. Il sito, per interfacciarsi con il DB, utilizza un modulo JDBC (Java DataBase Connectivity) che permette di eseguire Query direttamente dal codice Java. La gestione del DB è resa dunque semplice ed efficiente in termini di sviluppo e al tempo stesso la persistenza ci viene garantita dall'utilizzo del DBMS che serve per manipolare i dati ed evitare perdite . Il DBMS scelto per la gestione dei dati persistenti è MySQL , il quale utilizza SQL come linguaggio per la manipolazione dei dati.

2.5 Controllo degli accessi e sicurezza

L'e-store sarà realizzato in modo da essere utilizzato da 2 tipi di utenti ,ovvero amministratori e utenti . L'utente è suddiviso in utente esterno e cliente. L'utente esterno è una persona che si deve ancora registrare al sito e dunque non può usufruire dei servizi che esso offre. Il cliente, invece, essendo già registrato, dopo aver effettuato il login, può utilizzare le funzionalità allo stesso dedicate.

- L'amministratore potrà :
    - Effettuare il login e il logout
    - Visualizzare e rimuovere gli utenti registrati.
    - Visualizzare e rimuovere gli ordini di tutti gli utenti
    - Inserire , modificare e rimuovere prodotti dallo store.
- L'utente esterno potrà :
    - Navigare senza usufruire delle funzionalità del sito, dedicate esclusivamente al cliente.
    - Registrarsi al sito.
- Il cliente potrà :
    - Effettuare il login e il logout
    - Modificare l'account.
    - Visualizzare e ricercare i prodotti.
    - Inserire i prodotti nel carrello.
    - Effettuare l'ordine.
    - Visualizzare e cancellare ordini.

2.5.1 Sicurezza

La sicurezza del sistema è garantita dall'obbligo di accesso al sito da parte di tutti gli utenti. Ogni utente, prima di accedere alle funzionalità relative al tipo di account con cui effettua l'accesso, deve compilare il form di login con i propri dati d'accesso. Tali credenziali dovranno essere inserite ogni qualvolta si desideri utilizzare il sito, facendo si che abbia inizio una nuova "sessione" di navigazione.
Tale sessione terminerà nel momento in cui l'utente richiederà esplicitamente di effettuare il logout.

Nel caso l'accesso al sito non abbia successo, l'e-commerce notificherà l'errato inserimento evidenziando in rosso i campi non corretti. L'utente potrà effettuare un nuovo tentativo correggendo tali campi.

Poiché le funzionalità offerte variano in base alla tipologia di utente, il sistema proposto mostrerà diverse viste dello stesso. Dopo l'autenticazione, infatti, sarà visualizzata dall'utente una schermata diversa in base a quelle che sono le funzionalità dedicate a quella tipologia di user.

2.6 Flusso di controllo globale

2.6.1 Flusso di controllo esterno

Il controllo del flusso del software è regolato da classi Java che fungono da ricevitori di eventi. Esse rispondono alle attivazioni del client.

1. Le richieste vengono generate e inviate da un client.
2. La classe preposta a gestire l'evento associato alla richiesta, prendendo degli input, si preoccupa di inizializzare le richieste ed inoltrarle alle classi per lo svolgimento della operazione.
3. La classe gestore, una volta ottenuto il risultato, si preoccupa di inoltrarlo al client che aveva generato la richiesta.

Il sistema , come prestabilito nella descrizione dell'architettura MVC, fa uso di:

- JSP e Servlets per processare e gestire le richieste
- Entità "model" che, utilizzando entità "bean" (raffigurazioni degli oggetti persistenti), effettuano le operazioni sul database e restituiscono un risultato. Tale risultato verrà inviato al client facendo in modo che le informazioni siano rese visibili mediante l'uso di JSP.

2.6.2 Controllo della concorrenza

Il sistema deve permettere a più utenti di accedere al sito contemporaneamente. La concorrenza deve essere gestita dal DBMS del database server. Tale DBMS deve fare in modo di facilitare la manutenzione del sito e gestire efficacemente la coordinazione della concorrenza degli accessi al DB.

2.7 Boundary Conditions

Di seguito sono riportate alcune condizioni limite a cui il sistema può andare incontro.

2.7.1 Configuration e Start-up del sistema

All'avvio del sistema i vari sottosistemi non hanno necessità di accedere ai dati. Il sistema viene utilizzato nel momento in cui un utente interagirà con esso.

2.7.2 Failure del sistema

Nel caso in cui si verifichi un crash del sistema , dovuto ad un errore hardware o software, nel client o nel server, si tenta un ripristino della sessione precedente al crash forzando il riavvio del sistema.
Visto che i dati vengono gestiti dal DBMS non c'è rischio di perdita : le transazioni effettuate con il DBMS sono infatti atomiche . Tuttavia, in caso di guasti al supporto di memorizzazione dei dati nel database server, si va in contro ad una perdita di essi.

Per minimizzare tale rischio bisogna eseguire periodicamente dei backup del database del sistema. Inoltre tale problema viene segnalato affinchè l'amministratore possa provvedere alla risoluzione dello esso.

Per prevenire tali problemi è consigliabile effettuare dei controlli periodici sull'hardware.

In caso di crash dovuto ad un bug nel codice del sistema si distinguono 3 casi :

1. Il crash è avvenuto nel client. Il sistema potrà continuare a funzionare regolarmente salvo che non venga riutilizzata la funzionalità che ha causato il crash del sito.
2. Il crash si è verificato nell'application server. Il sistema risulta dunque inutilizzabile poiché non è possibile comunicare con il database.
3. Il crash è avvenuto nel database server. Il sistema risulta inutilizzabile in quanto non è possibile accedere ai dati.

2.7.3 Terminazione del sistema

La terminazione del sistema avviene solo nel caso in cui tutti i sottosistemi siano disattivati : nel caso in cui uno o più sottosistemi siano ancora attivati , il sistema rimane attivo. Per non incorrere in problemi è consigliabile disattivare le sessioni attive prima di disattivare l'application server.

3. Servizi sottosistemi

GestioneUtente

3.1. SS_0 Sottosistema per RF_0 GestioneAutenticazione

Application Layer	Presentation Layer	Data Layer
Login-*searchUser*	Home.jsp->LoginForm,LoginButton ->AreaUtente.jsp	//
Logout	LogoutButton->Home.jsp	//

3.2. SS_1 Sottosistema per RF_1 GestioneAreaRiservata

Application Layer	Presentation Layer	Data Layer
Registation-*insertUser*	Home.jsp->RegistrationButton ->Registration.jsp- >RegistrationForm	//
Modifiy- *redirectForUpdate aggiornaUtente*	Home.jsp -> ImpostazioniUtente.jsp ->ModificaForm	//

3.3. SS_2 Sottosistema per RF_2 GestioneOrdini

Application Layer	Presentation Layer	Data Layer
View books-*getAllLibri*	AreaUtente.jsp -> BookButton -> book.jsp	//
View cds-*getAllCd*	AreaUtente.jsp -> MusicButton ->music.jsp	//
Search product - *searchProdottoDaTitolo*	AreaUtente.jsp -> SearchButton -> ProdottoRicerca.jsp	//
product in cart - *addToCart* *deleteFromCart* *emptyCart*	ProdottiRicerca.jsp || book.jsp || music.jsp	//
View products in cart	AreaUtente.jsp -> CartButton -> cart.jsp	//
Make order - *insertOrder*	AreaUtente -> CartButton -> cart.jsp -> ContinueButton -> Continue.jsp -> OrderButton	//
Delete order - *deleteOrder*	AreaUtente -> OrdiniButton -> Ordini.jsp -> RemoveButton	//

GestioneAdmin

3.1. SS_4 Sottosistema per RF_3 GestioneAutenticazione

Application Layer	Presentation Layer	Data Layer
Login-*searchUser*	Home.jsp->LoginForm,LoginButton ->AreaUtente.jsp	//
Logout	LogoutButton->Home.jsp	//

3.5. SS_4 Sottosistema per RF_4 GestioneOrdini

Application Layer	Presentation Layer	Data Layer
View orders - *getMyOrders*	AreaAdmin.jsp -> VisualizzaOrdiniButton -> getMyOrders	//
Delete orders - *deleteButton*	AreaAdmin.jsp -> VisualizzaOrdiniButton ->getMyOrders -> deleteButton	//

3.6. SS_5 Sottosistema per RF_5 GestioneProdotti

Application Layer	Presentation Layer	Data Layer
Insert product - *insertProduct*	AreaAdmin.jsp -> InsertProductButton -> AggiungiProdotto.jsp -> AggiungiProdottoForm	//
Modify product - redirectForUpdate *aggiorna*	AreaAdmin.jsp -> VisualizzaProdottiForm -> ListaProdotti.jsp -> ModifyButton -> ModificaProdotto.jsp -> MoficaProdottoForm	//
Remove product - *deleteProduct*	AreaAdmin.jsp -> VisualizzaProdottiForm -> ListaProdotti.jsp->deleteButton	//

3.7. SS_6 Sottosistema per RF_6 GestioneUtenti

Application Layer	Presentation Layer	Data Layer
View users - *listaUtenti*	AreaAdmin.jsp -> VisualizzaUtentiButton -> ListaUtenti.jsp	//
Delete users - *deleteButton*	AreaAdmin.jsp -> VisualizzaUtentiButton ->ListaUtenti.jsp -> deleteButton	//

Glossario

Acronimo	Descrizione
RAD	Requirements Analysis Document
DBMS	Database Management System
SDD	System Design Document
HW	Hardware
SW	Software
JDBC	Java DataBase Connectivity
DB	DataBase
SQL	Structured Query Language