Progetto di esame di "Sistemi

e

Applicazione di rete"

My Market

Caratteristiche generali del progetto

My market è un sito di aste che si pone prima di tutto di soddisfare i seguenti servizi a un utente:

- Registrazione e login al sito.
- Inserimento di prodotti con opportuna descrizione e successiva messa in vendita di esso indicando un prezzo base e una data di scadenza d'asta.
- Visualizzazione di tutte le aste aperte con possibilità di fare un'offerta.
- Monitoraggio di tutte le aste create dall'utente stesso e di tutte le aste a cui un utente abbia partecipato (anche quelle concluse).

Tecnologie usate:

- HTML, CSS
- Servlet e JavaServer Pages (JSP)
- Java Database Connectivity (JDBC)

Tools di sviluppo:

- Eclipse Java EE IDE + Tomcat 6.0 server
- MySQL WorkBench

n.b.: è stato richiesto che il progetto fosse realizzato usando principalmente JSP, tuttavia per rimanere fedeli il più possibile al modello MVC si è scelto di usare una Servlet come Controller.

Struttura del software

In figura 1, alla pagina successiva, è presente il Class Diagram dove sono rappresentate le classi Java con i loro attributi e i loro metodi.

User

Mette a disposizione metodi che accettano parametri come username e password per le operazioni di login e registrazione di un utente, ma anche metodi invocati dalla Servlet di controllo per validare l'input inerente a un utente.

Il metodo più utilizzato sia da altri metodi, che dalla Servlet, ma anche dalle pagine JSP è senza dubbio User.getUsername():string che restituisce l'username dell'utente loggato.

Prodotto

Questa classe java serve per qualsiasi operazione riguardante un prodotto.

In generale utile per reperire il nome e la descrizione di uno specifico prodotto salvati in vettori di stringhe. Le principali operazioni su un prodotto fornite dai metodi di questa classe sono: inserimento, modifica, cancellazione, vendita.

Inoltre alcuni suoi metodi permettono di verificare errori di inserimento da parte dell'utente nella maggior parte delle operazioni appena citate.

Asta

I dati con cui questa classe lavora maggiormente sono: idasta, prezzo base, scadenza dell'asta. Il suo metodo più importante serve a creare una nuova Asta (inserimento di una nuova tupla nella tabella relativa alle aste). Gli altri metodi riguardano prevalentemente il reperimento di informazioni (sopracitate) riguardante una o più aste (accesso alla tabella "asta").

Offerta

Offetta.java possiede metodi che acquisiscono (in primis dal DB) informazioni riguardo un'offerta fatta da un utente e/o in particolare l'offerta massima (ovvero l'ultima fatta). Inoltre è possibile reperire il nome dell'offerente.

In particolare, questa classe possiede un metodo fondamentale: la creazione di una nuova offerta. A livello di database l'operazione appena citata comporta un'inserimento di una nuova tupla nella tabella "offert".

User url: String dbName: String a driver: String userNameMySQL: String p passwordMySQL: String

- username: String a errors: Hashtable<String,String>
- User() User(String)
- setUsername(String):void
- getUsername():String
- checkReg(String,String,String,String,String):boolean
- getErrorMsg(String):String
- regUser(String,String,String,String,String):void
- logUser(String):boolean
- existingUser(String):boolean

Prodotto

- url: String
- dbName: String
- driver: String
- userNameMySQL: String passwordMySQL: String
- prodid: Vector<String>
- prodName: Vector<String>
- prodDesc: Vector<String>
- errors: Hashtable<String,String>
- Prodotto()
- setProdld(String):void
- getProdld(int):String
- setProdName(String):void
- getProdName(int):String
- setProdDesc(String):void
- getProdDesc(int):String
- getCurrentProdName():String
- getCurrentProdDesc():String
- countProd():int
- insProd(String,String,String):void
- viewOneProd(String):void
- viewProd(String):void
- sellProd(String):void
- updateProd(String,String,String):void
- delProd(String):void
- getErrorMsg(String):String
- checkUpProd(String,String):boolean
- checkInsProd(String,String,String):boolean
- existingProd(String,String,String):boolean
- checkSellProd(String,String,String,String,String):boolean
- existingSell(String,String):boolean

Asta

- url: String
 bdbName: String
- a driver: String
- userNameMySQL: String
- passwordMySQL: String
- astald: Vector<String>
- astaScad: Vector<String>
- astaPrezzo: Vector<Integer>
- prodld: Vector<String>
- setAstald(String):void
- getAstald(int):String
- setAstaScad(String):void
- getAstaScad(int):String
- setAstaPrezzo(Integer):void
- getAstaPrezzo(int):Integer
- setProdld(String):void
- getProdld(int):String
- countAsta():int
- newAsta(String,String,String,Timestamp):void
- prodtoAsta(String):String
- astatoProd(String):String
- viewOtherAsta(String):void
- viewAsta():void
- viewAstaCreate(String):void

Offerta

- url: String
- dbName: String
- a driver: String
- userNameMySQL: String
- passwordMySQL: String
- astaldOfferente: Vector<String>
- p offerta: Vector<Integer>
- offerente: Vector<String>
- astaScad: Vector<String>
- errors: Hashtable<String,String>
- of Offerta()
- setAstaldOfferente(String):void
- getAstaldOfferente(int):String
- setOfferta(Integer):void
- getOfferta(int):Integer
- setOfferente(String):void
- getOfferente(int):String
- countAsteOfferente():int
- setAstaScad(String):void
- getAstaScad(int):String
- newOfferta(String,String,Integer):void
- viewOfferta(String):void
- viewAsteOfferente(String):void
- getErrorMsg(String):String
- checkOffer(String,String,Integer):boolean
- existingOff(String,String,String):boolean
- failOffer(String,String,Integer):boolean

Struttura del database

Sì è deciso di strutture il database in quattro tabelle (di cui è possibile osservare lo schema E/R in figura 2 alla pagina successiva).

user

Questa tabella contiene dati VARCHAR sull'utente. I più importanti sono password (a livello di applicazione per il login dell'utente) e la <u>username</u> che è anche la KEY di questa tabella.

Un utente può quindi inserire più prodotti e creare più aste.

prodotto

Le informazioni del prodotto sono prevalentemente dei VARCHAR. Tuttavia sono presenti anche un INT (idprodotto che è anche KEY) e un dato di tipo BIT che si chiama "invendita". Quest'ultimo campo è settato di default a 0 e a livello logico significa che un prodotto non è stato ancora messo in vendita.

asta

Un asta è identifica da un intero (idasta), ma caratterizzato in particolare dal prodotto (idprodotto) che è stato messo in vendita da un utente (username).

Si è usato uno SMALLINT per il prezzo_base perché considerato sufficiente grande come range (offerto nell'ordine delle decine di migliaia di euro) e un campo DATETIME per indicare la scadenza dell'asta.

Un'asta può avere più offerte (relazione 1:N con la tabella offer).

offer

La tabella offer contiene tutte le offerte, ognuna identificata da "idoffert", fatte per un asta ("idasta").

Per ogni tupla sono presenti anche altri due campi molto importanti: "offerta" in pratica uno SMALLINT che rappresenta l'offerta fatta e "offerente" rappresentato in VARCHAR.

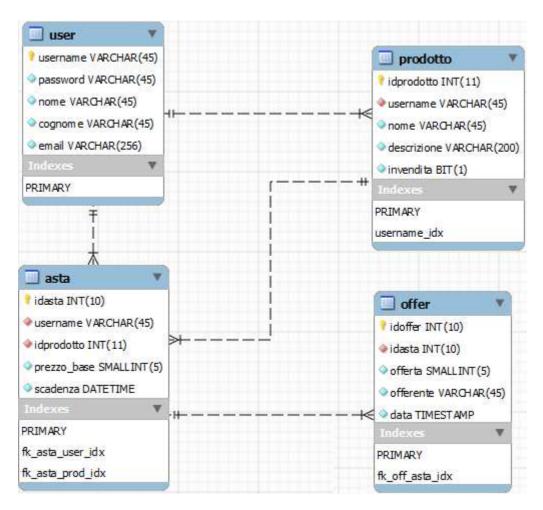


Figura 2: Schema E/R

Le modalità di interazione Client-Server

Trattandosi di un Sito Web, l'applicazione presentata nel progetto ha come client il browser. L'utente interagisce con il sito utilizzando appunto il browser.

Quest'ultimo manda una richiesta a una servlet, (chiamata ControllerServlet.java presente nel package "Controller" del progetto), che ha il ruolo di "contollore" ovvero di validare l'input.

Inoltre il controllore utilizza le <u>classi Java</u>* che modellano i dati (reperiti dal <u>Database</u>** tramite JDBC).

A questo punto deve essere generata una risposta al client: è il Controller che sceglie quale VIEW (in questo progetto si tratta di pagine JSP) selezionare come risposta al Client.

Questa modalità d'interazione è appunto rappresentabile (come in Figura 3) secondo uno schema MVC.

*fanno tutte parte del package "Model" descritte a pagina 3 e 4

^{**}descritto a pagina 5 e 6

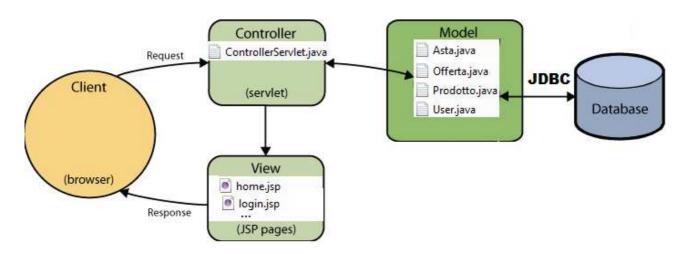


Figura 3: schema MVC

Conclusioni

Il progetto soddisfa i requisiti principali di un sito di asta online con anche una dovuta gestione degli errori in casi come refresh indesiderati dell'utente in fase di offerta o creazione d'asta e invio di post con campi compilati non appropriatamente.

Nello sviluppo del software si può notare una cercata fedeltà al modello MVC che soprattutto a livello logico ha facilitato la modifica/inserimento/cancellazione di metodi di ogni tipo (accesso al database, gestione degli errori, re direzioni a determinate pagine jsp).

Sempre in riferimento a questo progetto si è cercato di mantenere una correlazione logica tra tabelle e classi java che fanno uso di metodi per l'accesso al database. Quest'ultimo concetto lo si può spiegare notando che: la classe "User" fa riferimento alla tabella "user", la classe "Prodotto" alla tabella "prodotto", ...

L'unico aspetto che non si è concretamente affrontato è stato in generale quello prestazionale e in particolar modo se il sito possa essere scalabile.

Tuttavia concludendo a livello di progettazione software si può essere soddisfatti di questo progetto perché realizza le specifiche richieste basandosi su un approccio MVC che ha portato a diversi vantaggi sopracitati.