Università degli Studi di Modena e Reggio Emilia
Corso di Laurea Magistrale in Ingegneria Informatica

Applicazione web per la gestione di un'acetaia

Elaborato sul progetto di Sistemi e Applicazioni di Rete.

Tecnologia prevalente: servlet.

A.A. 2013/2014

Xhensila Doda

1 CARATTERISTICHE GENERALI DEL PROGETTO

GestioneAcetaia è un'applicazione web per la gestione di un'acetaia di piccole dimensioni.
Il sistema mantiene informazioni sui barili di aceto, che sono raggruppati in batterie, e consente di effettuare e tenere traccia di operazioni quali:

- Aggiunta di aceto nei barili
- Prelievo di aceto dai barili
- Rabbocco, che consiste nel prelevare aceto da un barile per versarlo in un altro
- Misurazione di alcuni parametri specifici del prodotto contenuto in ogni barile
- Memorizzazione della scansione di una scheda di degustazione
- Ricerca delle operazioni eseguite in passato, in base a uno o più dei seguenti parametri: barile e/o batteria coinvolti, periodo in cui è stata effettuata, tipo di intervento compiuto

Il progetto è stato realizzato usando Eclipse come ambiente di sviluppo, Apache Tomcat versione 7 come Web Server e MySql Workbench come database.
Per la parte di grafica è stato usato il *framework* Bootstrap come base, modificando poi tramite HTML e CSS i *template* di layout per adattarli meglio ai nostri scopi e necessità.
È stato richiesto di sviluppare l'applicazione mediante tecnologia Servlet tuttavia, al fine di rimanere più fedeli possibile al paradigma MVC (*Model-View-Controller*) abbiamo scelto di realizzare la parte di visualizzazione dell'output tramite JSP da poter dividere in tal modo la parte di rappresentazione grafica da quella logica che elabora i dati.

2 STRUTTURA DEL DATABASE

Nel database sono presenti otto tabelle: una per tener traccia di username e relative password, una per le informazioni sui barili e le altre per le possibili operazioni.

Admin
In questa tabella si tiene traccia dei vari username (*primary key*) relativi agli utenti autorizzati ad accedere all'applicazione con le corrispettive password di accesso.

Barile
Questa tabella tiene traccia di tutti i dati riguardanti ogni barile presente nell'acetaia. In particolare vengono indicati:

- idbarile, identificatore univoco del barile e *primary key* della tabella
- capacità massima di riempimento e legno in cui è realizzata la botte
- batteria a cui afferisce il barile
- livello massimo di riempimento (al fine di lasciare nel barile dello spazio per l'aria in modo che possano avvenire le reazioni aerobiche necessarie alla formazione dell'aceto) e livello effettivo di riempimento
- titolazione dell'acidità e misura della densità del prodotto contenuto nel barile

Storico
Tabella contenente tutte le operazioni effettuate. I parametri barile, batteria data e operazione indicano l'identificatore del barile e della batteria a cui appartiene il barile sul quale è avvenuta l'operazione, il tipo di intervento che è stato fatto (aggiunta, prelievo, rabbocco, misurazione o degustazione) e la data. L'identificatore univoco dell'operazione è *idOp,* un intero crescente a partire da 1. Esso viene usato, insieme all'identificatore del barile, come *primary key* della tabella, a cui fanno riferimento le tabelle dei singoli interventi.

Aggiunta
Tutti i dettagli relativi all'operazione di aggiunta vengono memorizzati in questa tabella. Ogni record è identificato dalla *primary key* idOperazione, *foreign key* riferita all'attributo idOp della tabella storico. Vengono poi riportati il barile sul quale viene eseguito l'intervento, il tipo di prodotto aggiunto (mosto o aceto) e la quantità.

Prelievo
Tutti i dettagli relativi all'operazione di prelievo vengono memorizzati in questa tabella. Ogni record è identificato dalla *primary key* idOperazione, *foreign key* riferita all'attributo idOp della tabella storico. Vengono poi riportati il barile sul quale viene eseguito l'intervento e la quantità prelevata.

Rabbocco
Tutti i dettagli relativi all'operazione di rabbocco vengono memorizzati in questa tabella. Ogni record è identificato dalla *primary key* idOperazione, *foreign key* riferita all'attributo idOp della tabella storico. Vengono poi riportati il barile dal quale viene prelevato il prodotto (src), quello in cui viene versato (dest) e la quantità.

Misurazione
Tutti i dettagli relativi all'operazione di misurazione vengono memorizzati in questa tabella. Ogni record è identificato dalla *primary key* idOperazione, *foreign key* riferita all'attributo idOp della tabella storico. Vengono poi riportati il barile sul quale viene eseguito l'intervento, il parametro misurato (livello di prodotto nel barile, titolazione acidità o densità) e il valore rilevato.

Degustazione
Tutti i dettagli relativi all'operazione di inserimento della scansione della scheda di degustazione vengono memorizzati in questa tabella. Ogni record è identificato dalla *primary key* idOperazione, *foreign key* riferita all'attributo idOp della tabella storico. Vengono poi riportati il barile il cui prodotto è stato testato e il percorso in memoria in cui è stata salvata la scheda.

Di seguito è riportata un'immagine che mostra lo schema del database.

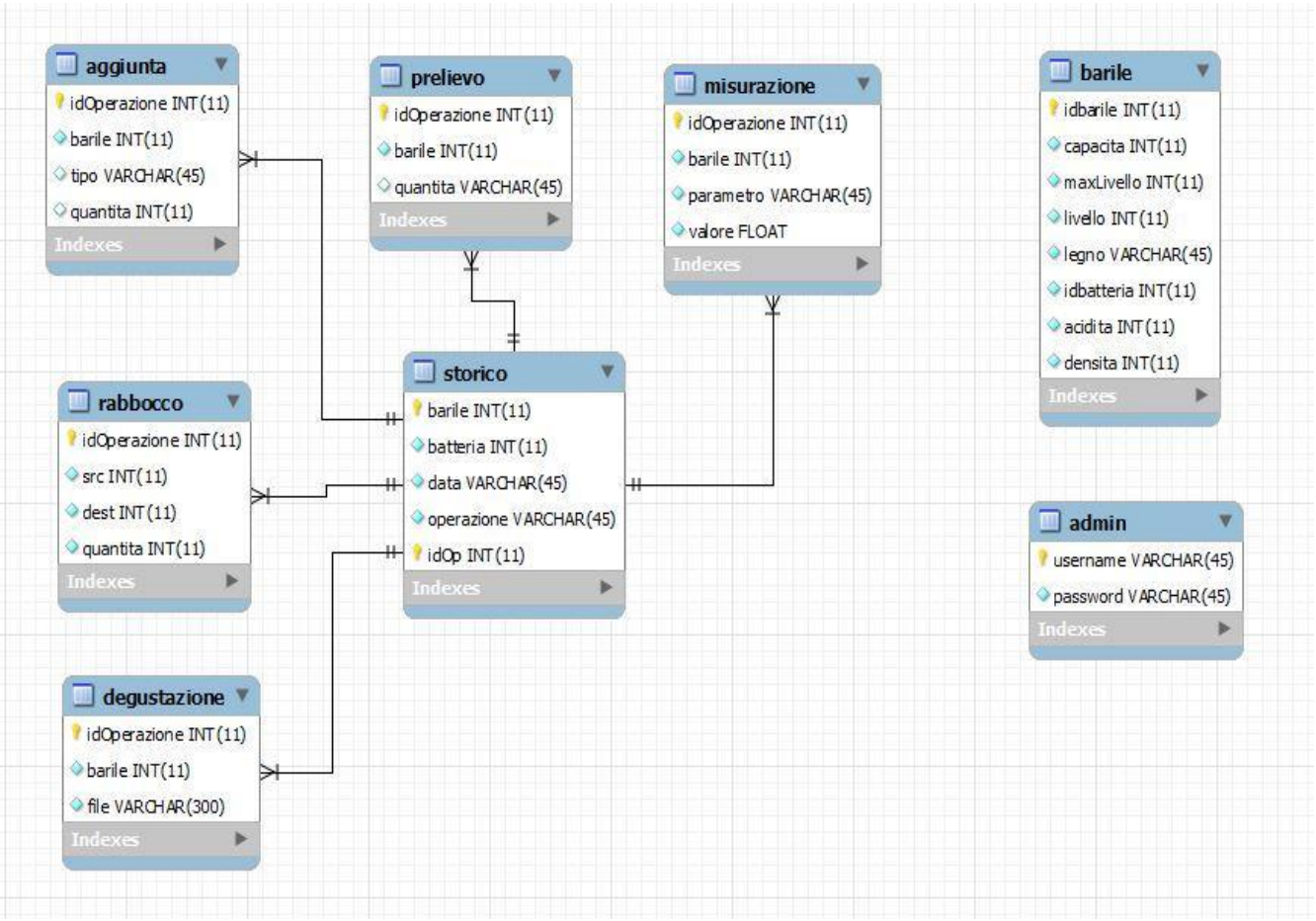

Figura 1.

3 STRUTTURA DEL SOFTWARE

L'applicazione web rispetta il paradigma MVC (*Model-View-Controller*).
La parte di *view* è stata realizzata tramite Java Server Pages (JSP): una pagina JSP per visualizzare la pagina di login, una per la *welcome page*, due per le pagine dei risultati delle operazioni (errore o successo), una per l'output della ricerca e le altre per un interfaccia in cui l'utente inserisce i dati necessari all'esecuzione di ogni operazione.
Per la parte di *model* sono presenti cinque classi, come mostrato nel seguente *Class Diagram*.

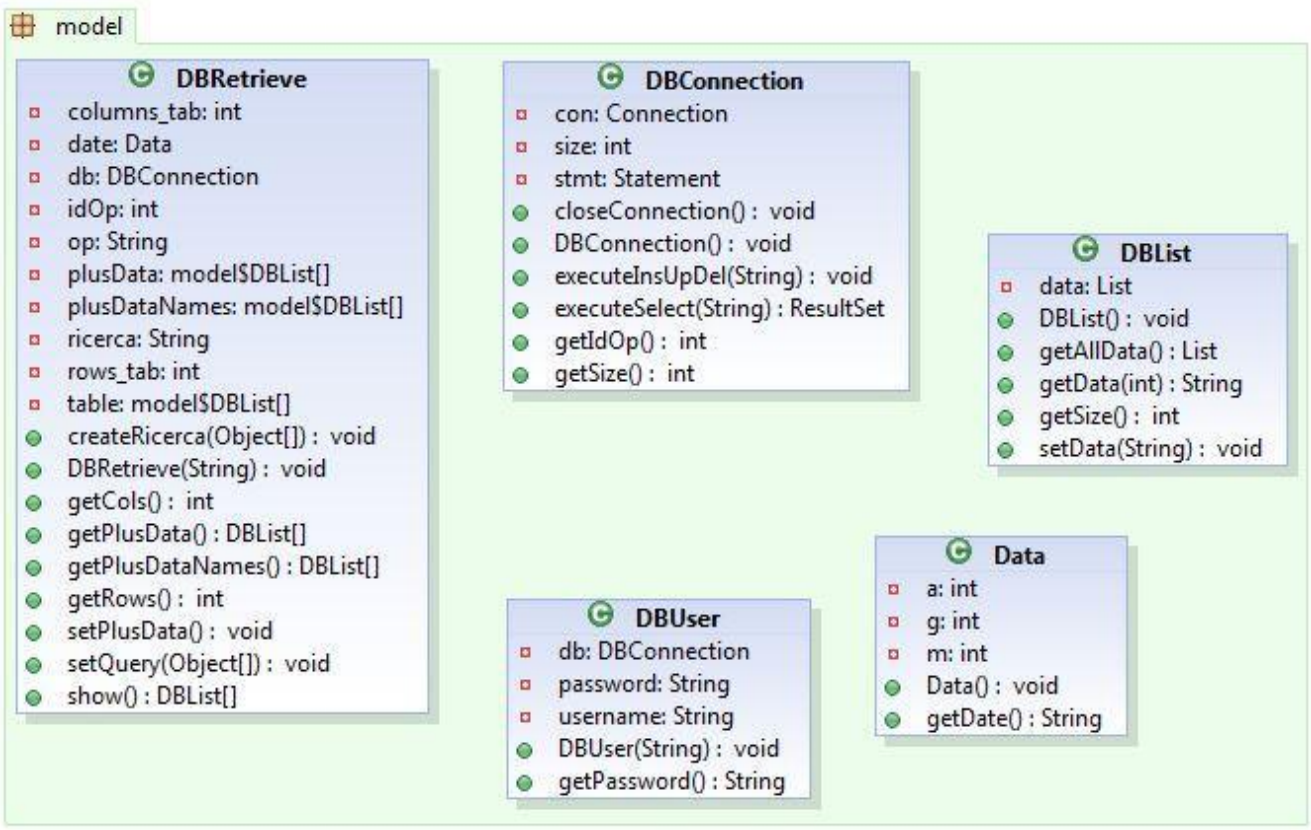

Figura 2.

Queste classi hanno il compito di stabilire una connessione col database, recuperare quando necessario i parametri mancanti ma richiesti dalle interrogazioni, formare ed eseguire le *query* specifiche per le diverse operazioni.
Inoltre viene fatto il controllo su *username* e *password* forniti in fase di autenticazione.

Per la parte di *controller* sono presenti nove classi di cui tre raccolte nel *package session*. Quest'ultime, due *servlet* e una classe JAVA con il compito di filtro, si occupano delle fasi di login e logout e verificano che un utente non autenticato non possa accedere alle pagine dell'applicazione. Le altre *servlet*, invece, gestiscono ognuna una diversa operazione e si occupano di caricare i dati presenti nel database che sono necessari alla relativa pagina JSP, prendere in input i parametri forniti dall'utente ed eseguire le modifiche richieste nel database. Si ricorda che sia il recupero sia la modifica e l'inserimento di nuovi valori nel database è effettuato da specifiche classi del *model* che vengono invocate dalle *servlet*.

Di seguito è mostrato il *Class Diagram* del *controller*.

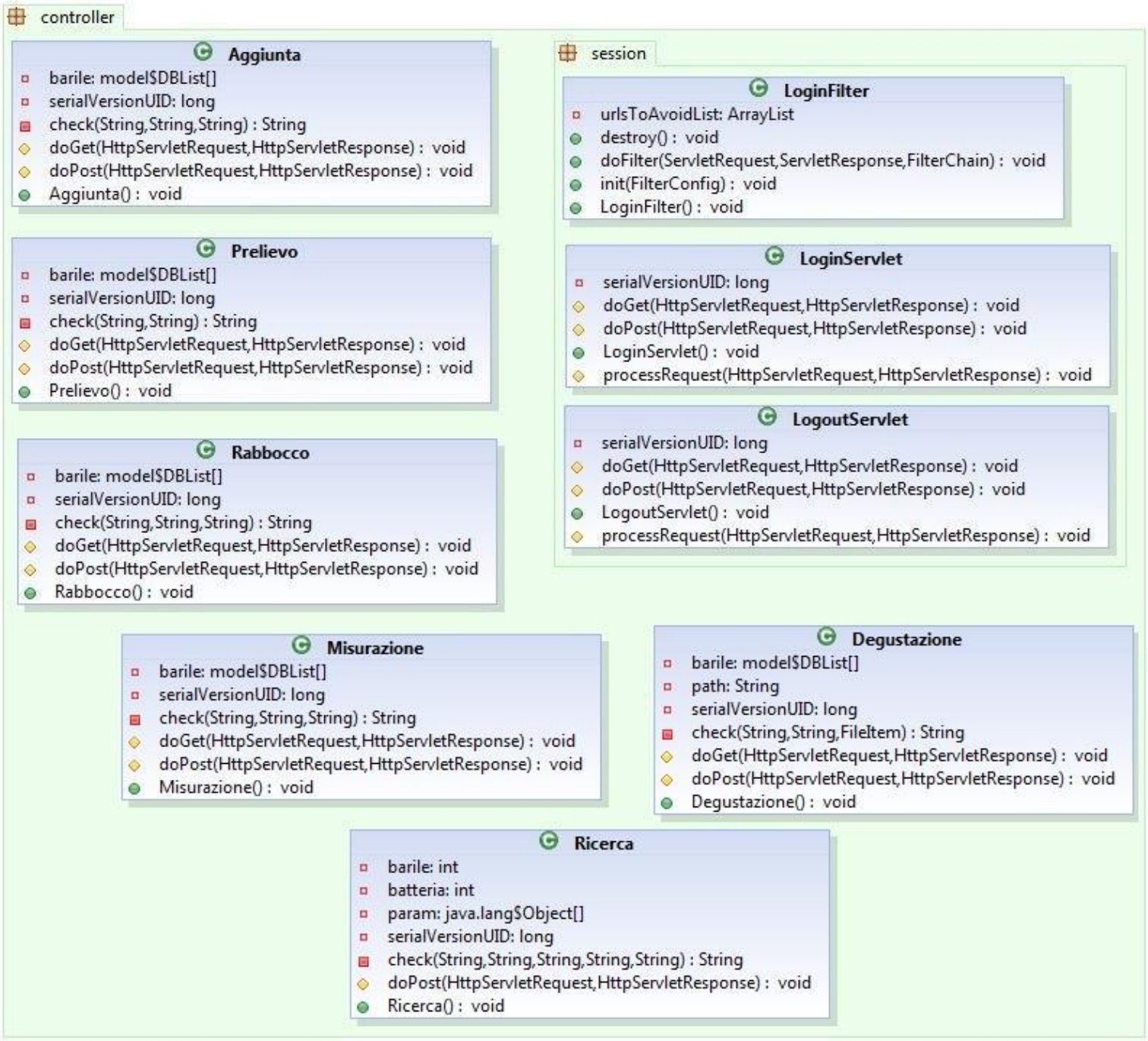

Figura 3.

4 MODALITÀ DI INTERAZIONE CLIENT-SERVER

Per avere una migliore organizzazione strutturale, il progetto è stato realizzato secondo la logica MVC anticipata all'inizio della documentazione.
Com'è solito nelle applicazioni web, anche in questo progetto il client è rappresentato dal browser. Tramite quest'ultimo, l'utente interagisce con l'applicazione inviando richieste di tipo *http* che vengono recuperate dalle *servlet*, le quali hanno il ruolo di *controller.* Esse, dopo aver effettuato una verifica sulla correttezza dei parametri passati in input, in base ad essi invocano metodi delle classi JAVA del *model*. Queste ultime si occupano della connessione al database e dell'elaborazione dei dati, basandosi sui parametri passati dalle *servlet*. Una volta che i dati sono pronti, vengono mandati alle *servlet* che questa volta, invece, inviano i dati all'utente tramite risposta *http,* richiamando le apposite pagine JSP della *view* responsabili della loro visualizzazione.
Per motivi di sicurezza, l'applicazione è provvista di una pagina di login per l'autenticazione dell'utente. Una volta autenticato, si può accedere alle diverse funzionalità di cui dispone l'applicazione web. Ogni richiesta da parte dell'utente richiama la logica descritta sopra secondo il modello MVC, di cui è fornita la rappresentazione seguente.

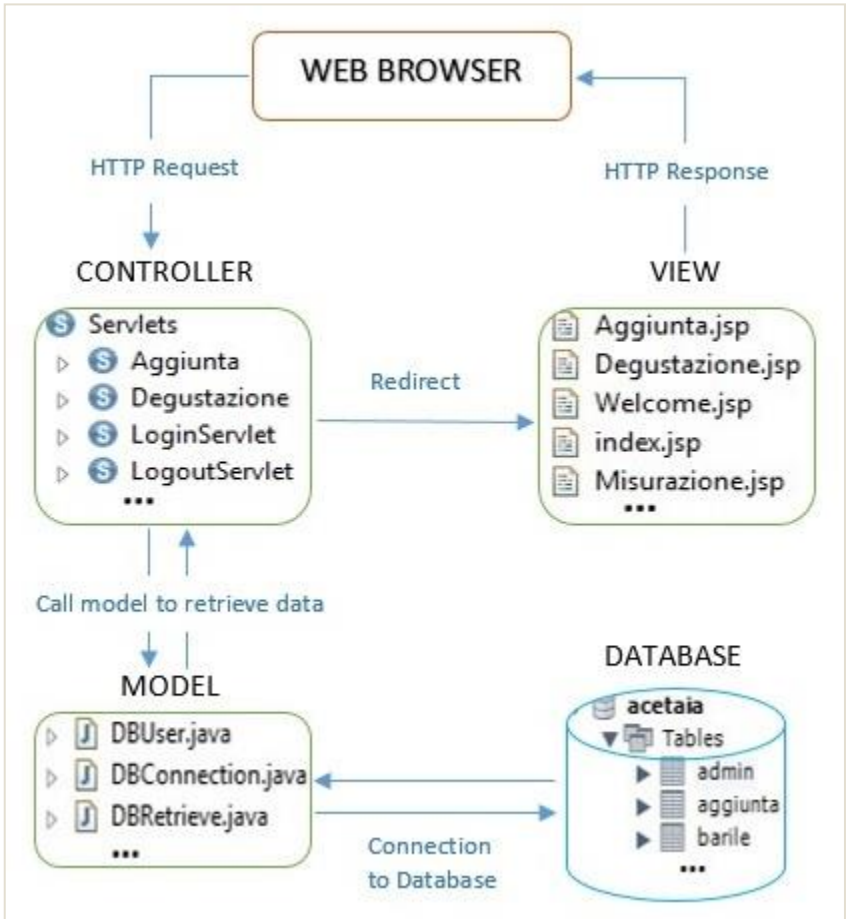

Figura 4.