Travel Dream

Matteo Camagni , Fabio Dellea

Matricole: 820203 817045

Anno: 2013 / 2014

Consegna: 29 / 11 / 2013

INDICE

Design Document

1 – Design dei Dati

Modello ER

All'interno del progetto sono state individuate quattro entità i cui dati è bene siano registrati in un database.

La prima è il *cliente*, i cui dati devono essere necessariamente registrati in modo da poter effettuare il login al sistema.

La seconda è l'*impiegato*, altro attore fondamentale del sistema, che ha anche lui necessità di login.

Pensiamo sia significativo poi tener traccia delle *prenotazioni* effettuate (non sviluppato, da solo conferma della prenotazione), in modo da avere uno storico delle vendite nel tempo e anche della *wishing list* di ciascun cliente in modo che sia sempre accessibile a ogni login.

L'ultima, fondamentale, entità del nostro modello sono i *prodotti*, i cui dati devono essere in memoria, in modo da permetterne la visualizzazione. I prodotti sono poi stati specializzati in base al tipo e divisi nelle categorie *volo, soggiorno, escursione* e *pacchetto*, quest'ultima costruita

ricorsivamente da tre prodotti (il pacchetto è un prodotto che ha tutti valori nulli tranne che per idVolo, idEscursione, idSoggiorno, prezzo e descrizione). Abbiamo fatto ciò per poter cogliere le singole peculiarità di ogni prodotto in maniera adeguata.

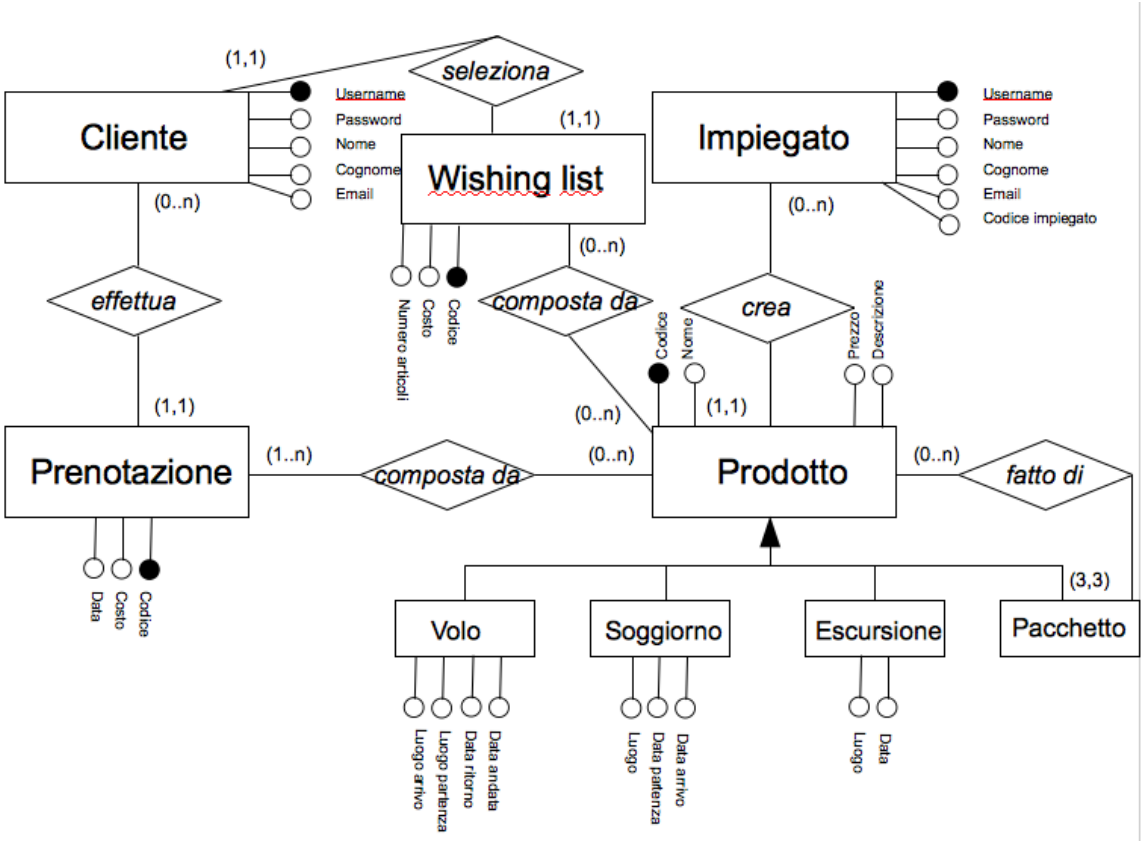
Le relazioni significative individuate tra le entità sono state cinque.

Abbiamo poi la relazione *effettua*, che indica da che cliente è stata effettuata la prenotazione (non viene tenuta traccia dell'effettiva prenotazione).

Ogni prenotazione e ogni whishing list sono *composte da* uno o più prodotti, che sono *creati* da un'impiegato.

Anche se di primo acchito la relazione *crea* che collega l'impiegato al prodotto da lui inserito in database può sembrare forzata, abbiamo pensato che possa essere utile. In caso di errore di un dipendente nell'inserimento di un prodotto, il responsabile d'ufficio potrebbe infatti scoprire rapidamente e facilmente di chi è la colpa, senza dover accedere ai file di log del database, la cui consultazione è ben più macchinosa.

Abbiamo inoltre individuato i principali attributi delle varie entità. A questo riguardo vorremmo fare un rapido commento solo rispetto alle chiavi primarie scelte: nel caso di cliente e impiegato abbiamo selezionato come chiave primaria l'userid (o username), basandoci sul fatto che evidentemente non possano essere registrati nel sistema due utenti con il medesimo nome utente. Sarebbe forse stato più comodo usare anche in questo caso un identificativo progressivo automaticamente inserito dal sistema, però questa scelta conferisce più semantica al database, permettendoci inoltre di delegare al server della base di dati il controllo sulla non esistenza di due identici username, che avremmo altrimenti dovuto effettuare esplicitamente noi nel sistema.



Note: il modello è stato rivisto durante la progettazione. Non c'è più una gerarchia in prodotto, ma una sola entità prodotto per: volo, escursione, soggiorno e pacchetto. Di fatto sono tutti prodotti che hanno tutti gli attributi di tutti (esempio volo è un prodotto che ha gli attributi oraPartenza =NULL). La relazione "fatto di" non esiste più in quanto il pacchetto è anchesso un prodotto con codSoggiorno, CodEstensione,CodVolo relativi ai prodotti che lo compongono.

Cliente ed impiegato sono una solo entità user (per comodità) si distinguono su un valore che indica se admin o cliente.

La prenotazione non è più entita in quanto non è stata implementata per essere memorizzata.

Modello Fisico

Nella traduzione da modello ER a modello fisico, abbiamo pensato di effettuare alcune ottimizzazioni, evitando tabelle con dati evidentemente ridondanti e non utili per i fini del nostro sistema.

Il modello fisico sarà perciò il seguente:

User (userId, password, nome, cognome, email)

Wishing list (codice, idProduct, user_cliente)

Prodotto(ID, aereopAndata, aereopRitorno, codEscursione, codSoggiorno, codVolo, dataArrivo, dataPartenza, descrizione, luogo, nome, prezzo, tipo)

UserGroup (userId, groupId)

Groups(groupId)

La traduzione delle entità user è stata fatta in maniera automatica, senza modifica alcuna dal modello ER.

Per quanto riguarda l'entità prodotto abbiamo deciso di effettuare la seguente scelta: avremo una tabella sola che identifica o un volo, o un soggiorno, o un'escursione e o un pacchetto, in cui verranno salvati tutti i dati del prodotto. Avremo poi una tabella di corrispondenza che abbinerà a ogni codice prodotto delle tabelle specializzate un univoco identificativo prodotto generico.

Tale scelta è stata dettata dalla necessità di avere un identificativo di prodotto che fosse unico per tutti i vari prodotti, in modo da poter facilmente creare le tabelle di relazione che coinvolgono l'entità prodotto. Un'unica tabella con dentro tutti i prodotti, con distinzione di tipo in base all'attributo tipologia che può essere "volo", "hotel", "escursione", "pacchetto"0, anche se ogni prodotto ha caratteristiche e attributi leggermente diversi.

La relazione *composta da* è tradotta nelle tabelle composizione prenotazione e composizione wishing list, che sono due semplici tabelle di corrispondenza tra le prenotazioni/wishing list e i prodotti che le compongono, con doppia chiave primaria essendo una relazione molti in molti.

Identico discorso vale infine per la tabella creazione prodotti, che traduce la relazione *crea* sussistente tra impiegato e prodotto.

2 – Design dell'Applicazione

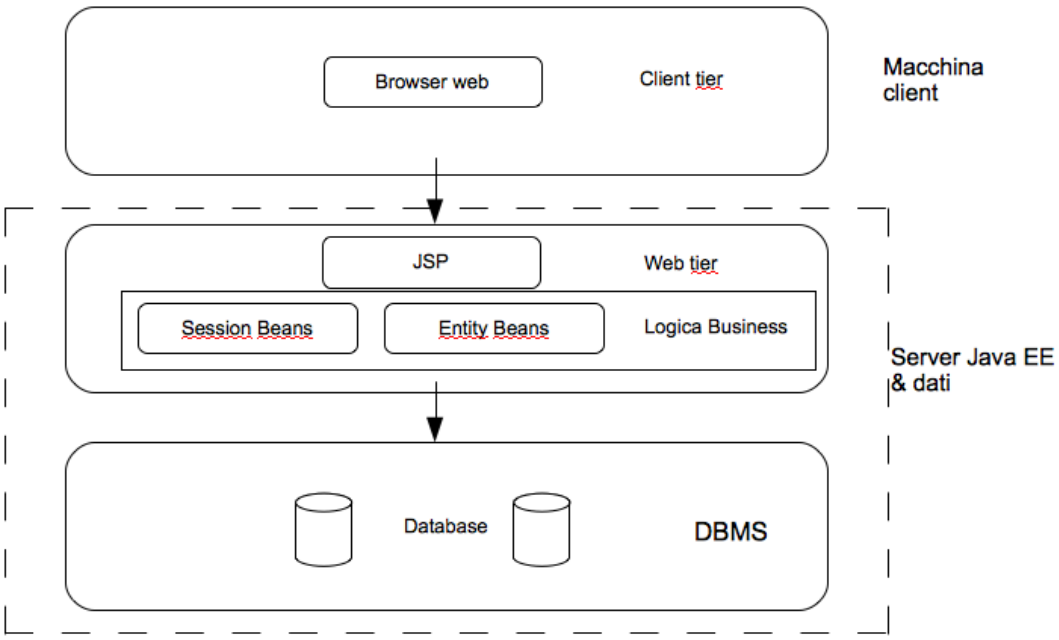
Modello Architetturale

Per realizzare il sistema verrà utilizzato Java EE come infrastruttura di supporto.

Nello specifico utilizzeremo GlassFish, un application server open source che supporta Enterprise Java Beans per la logica business e il collegamento al database, RMI per chiamate remote a procedure, JSP per la creazione dinamica di pagine web e un derivato di Apache Tomcat come servlet container.

Sulla macchina client non servirà l'installazione di una applicazione dedicata, ma si potrà accedere al sistema tramite browser.

Il database sarà infine gestito da un server MySQL, localizzabile sia sulla stessa macchina del web server che, in caso di necessità di scalare i dati su una maggior quantità di memoria, su una macchina dedicata.



Modello di Navigazione

Un utente non registrato può visualizzare inviti ricevuti (che riguardano prodotti o pacchetti) oppure può utilizzare la funzionalità di registrazione e diventare a tutti gli effetti un possibile cliente.

Un utente registrato (cliente) per poter utilizzare le proprie funzionalità (quali per esempio la ricerca di prodotti/pacchetti, l'acquisto…) deve necessariamente effettuare il login al sistema.

Una volta effettuato il login il Cliente avrà a disposizione una pagina personale ("Personal Page Cliente") da cui potrà svolgere le seguenti operazioni:

- Gestione Account : Il cliente può gestire il proprio profilo Account e in particolare modificare i propri dati personali.
- Visualizzare la Wishing-List: ovvero visualizzare i prodotti precedentemente aggiunti dal Cliente stesso .
- Effettuare una ricerca: se questa va a buon fine il Cliente potrà decidere, se il prodotto è di suo interesse, di:
    - aggiungerlo alla wishing-list. A questo punto viene mostrata al Cliente la lista dei prodotti-pacchetti presenti fino a quel momento nella Wishing-List e da qui potrà confermare una prenotazione o eliminare alcuni prodotti dalla Wishing List (concludendo o meno una Prenotazione).
    - Inviare un invito ad un suo amico del prodotto selezionato (il codice deve essere inviato manualmente dal cliente all'amico stesso)

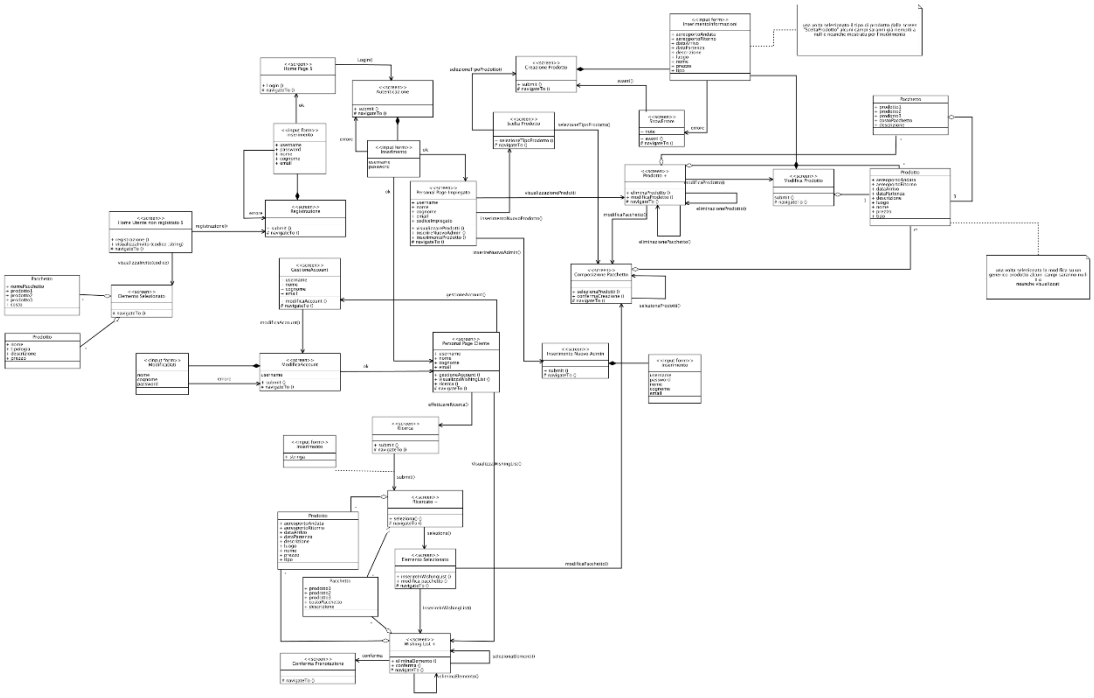
Un utente registrato può essere un Impiegato.

Similarmente a quanto detto per il cliente, anche all'Impiegato verrà mostrata la sua Personal Page da cui potrà, oltre che gestire il proprio Account, gestire i Prodotti o i Pacchetti.

In particolare la gestione di essi prevede:

- Inserimento di nuovi, qualora non ci siano errori nell'inserimento (prodotto con lo stesso codice già presente)
- Modifica e Cancellazione, le cui funzionalità prevedono una ricerca di essi Note:

1. Le ricerche (per prodotto o per pacchetto) sono valide anche se inseriamo solo un attributo lasciando vuoti gli altri.
2. le ricerche per prodotto o pacchetto produco risultati diversi in quanto **si assume** che la ricerca di un pacchetto, sia che venga effettuata per codice o per nome del Pacchetto, fornisce sempre un risultato univoco, mentre questo può non essere vero per il prodotto.



Componenti

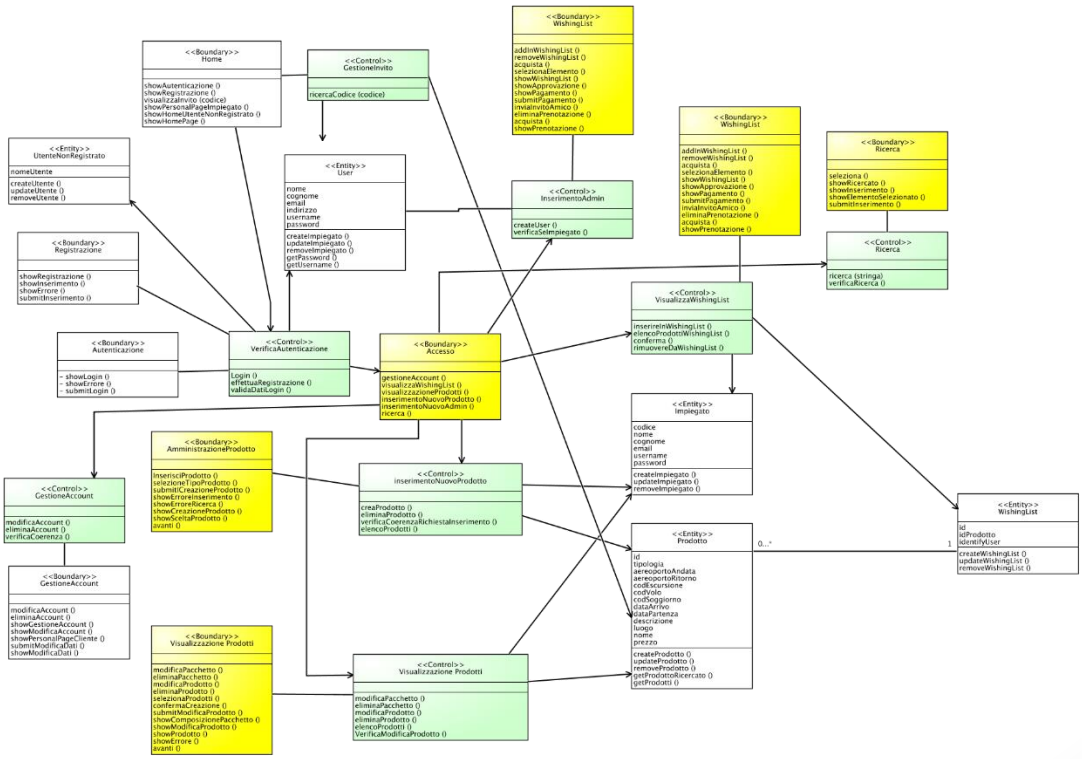
In questa fase l'obbiettivo è quello di separare i tre elementi di presentazione, di controllo e di interazione con dati o risorse esterne. Perciò ci si avvale del Boundary – Control – Entity (o BCE) che consente di rappresentare il sistema in diagrammi di analisi che consentono di separare la logica applicativa dalla presentazione dei dati. In particolare, avremo tre tipologie di classi definite come segue:

<< boundary>> indica l'interfaccia esterna del sistema verso l'utente

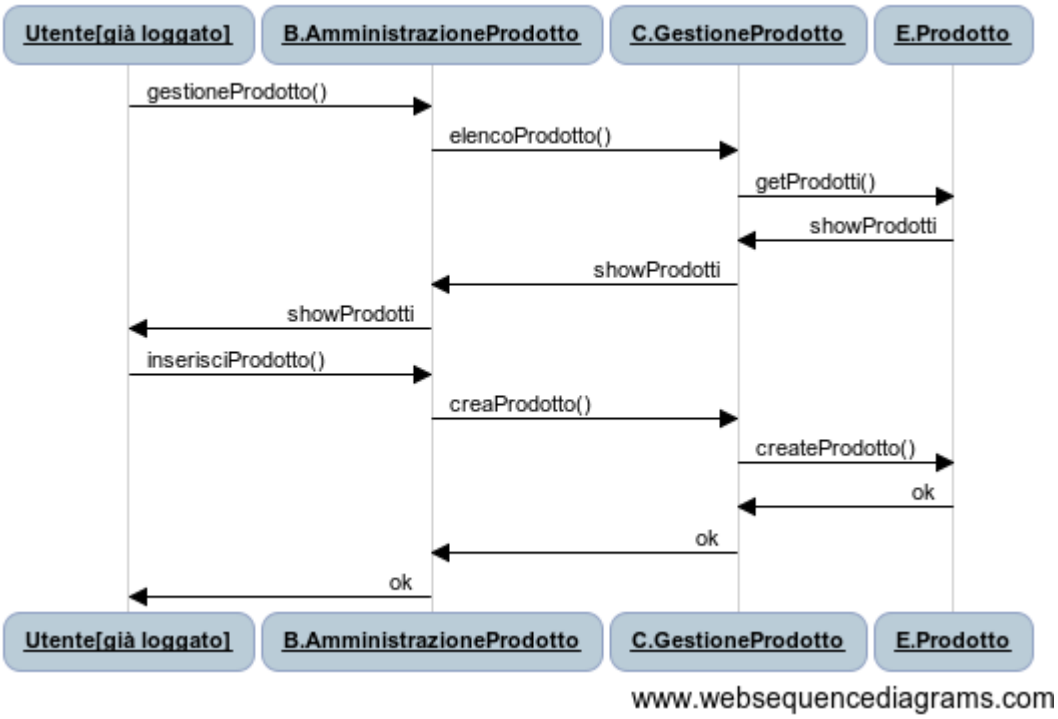
<< control >> rappresenta una classe che realizza la parte della logica applicativa. Queste classi gestiscono lo stato degli oggetti applicativi che sono utilizzati per fornire le funzionalità descritte attraverso i boundary.

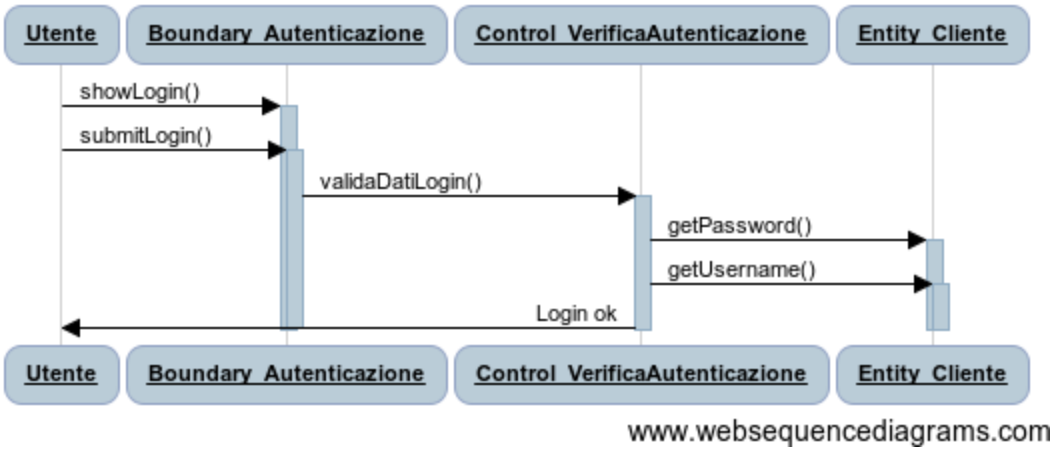
<< entity >> identifica una classe di accesso ai dati di interesse, memorizzati in basi di dati, che sono presenti nel sistema.

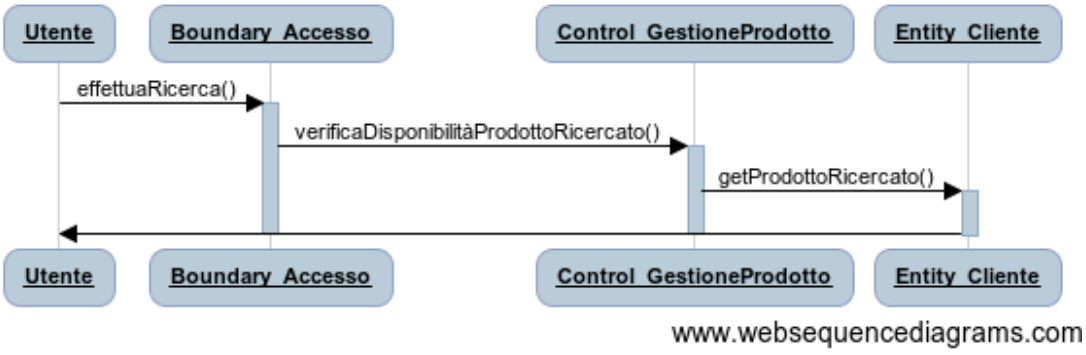
Di seguito viene fornita il linea di massima lo schema.



Diagrammi di sequenza




Utente
Boundary_Autenticazione
Control_VerificaAutenticazione
Entity_Cliente
showLogin()
submitLogin()
validaDatiLogin()
getPassword()
getUsername()
Login ok
Utente
Boundary_Autenticazione
Control_VerificaAutenticazione
Entity_Cliente
www.websequencediagrams.com


Utente
Boundary_Accesso
Control_GestioneProdotto
Entity_Cliente
effettuaRicerca()
verificaDisponibilitàProdottoRicercato()
getProdottoRicercato()
Utente
Boundary_Accesso
Control_GestioneProdotto
Entity_Cliente
www.websequencediagrams.com