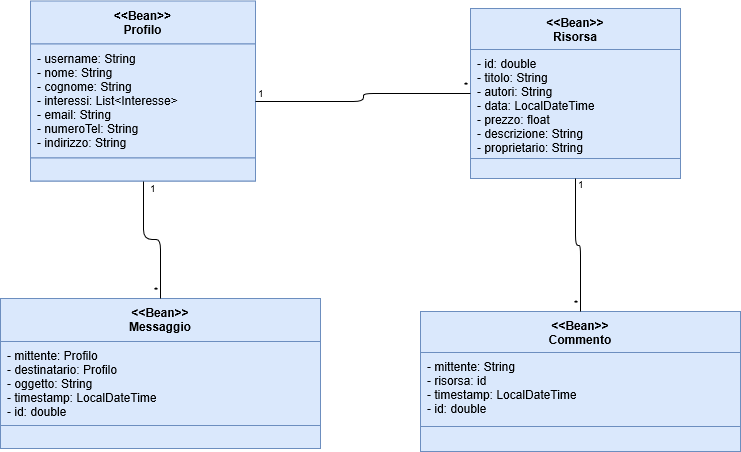
# Progettazione del dettaglio

Saranno riportati di seguito i diagrammi riguardanti il sistema

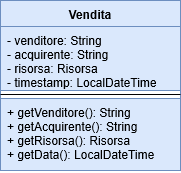
## Struttura

*Diagramma di Dettaglio: Dominio Risorse*

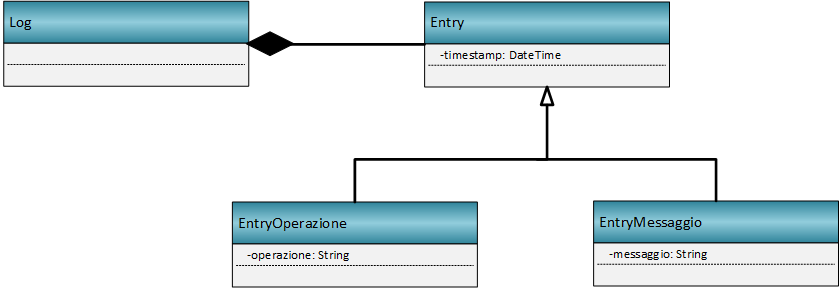


I metodi delle classi sono i tipici utilizzati per i Bean, ovvero getter e setter per ogni attributo mostrato nel diagramma. Non si è ancora deciso come il modello interagirà con il DBMS, questa scelta verrà lasciata alla fase implementativa. In ogni caso queste classi sono da pensare in relazione all’utilizzo di Framework similari ad Hybernate o Pattern DAO. È stata fatta questa scelta in rispetto dello standard de facto in vigore per le applicazioni Web che prevede un model composto da classi JavaBean. Sono state rimosse le entità *CasellaMessaggi* e *Libreria* in quanto semplici collezioni rispettivamente di Risorse e Messaggio.

*Diagramma di dettaglio: Dominio Vendita*



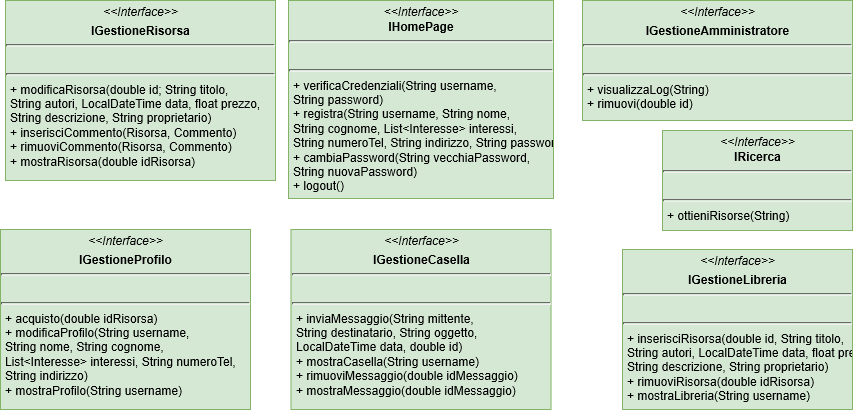
Anche questa entità è un tipico JavaBean

*Diagramma di dettaglio: Dominio Log*

Rimane identica all’Analisi del problema per due motivi.

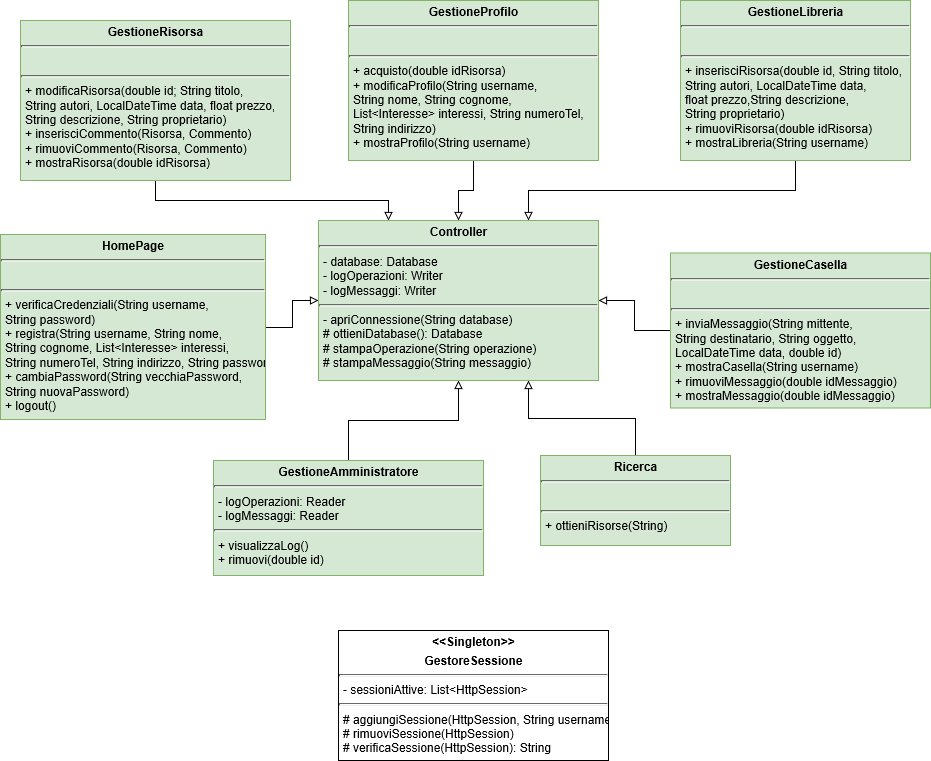
1. Le scritture avvengono su file di testo quindi non è necessario rapportarsi a un DBMS
2. La modellazione era già completa per quanto riguarda il nostro dominio

*Diagramma di dettaglio: Interfacce nel Server*



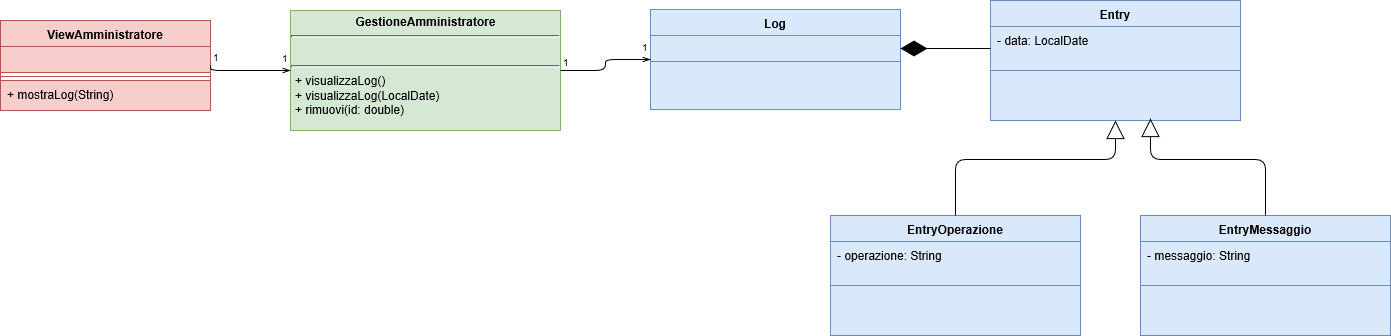
Il metodo *cambiaPassword* si occuperà come prima cosa di cifrare le password in chiaro, successivamente le spedirà e in caso di corrispondenza saranno effettivamente modificate. Il metodo *acquisto* utilizzerà il sistema esterno per validare il pagamento, successivamente si occuperà di rimuovere la risorsa dalla libreria del proprietario. Tutte queste classi sono da intendersi come Servlet i cui metodi prendono i loro parametri all’interno della richiesta get e post. Queste interfacce ci permetteranno di applicare il “The Inversion Design Principle”

*Diagramma del Dettaglio: Server*

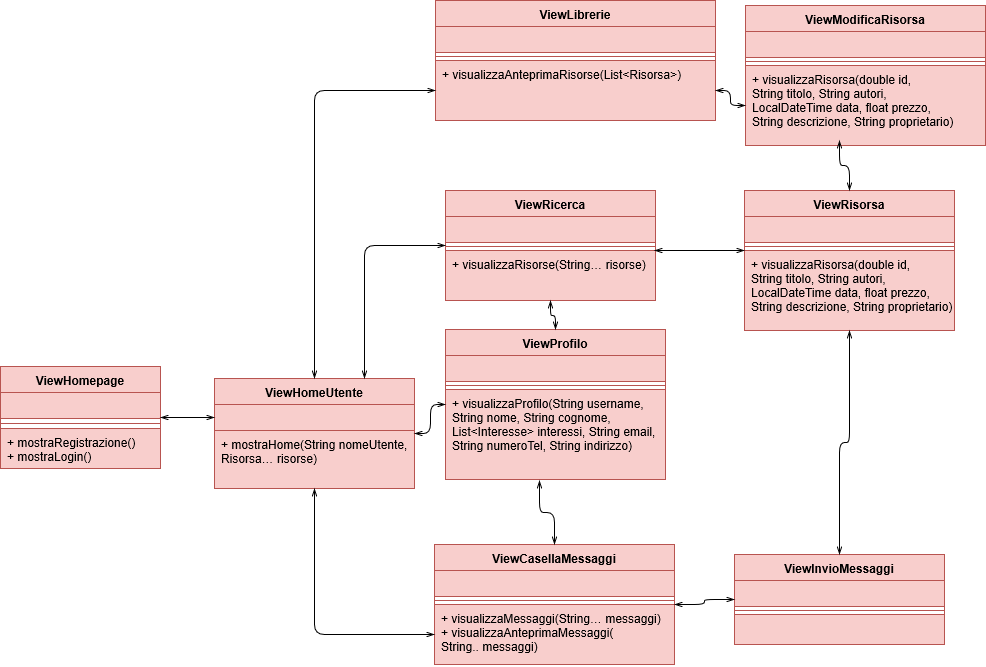
Alcuni appunti:

1. Il singleton GestoreSessione è utilizzato da tutti quanti i controller (tranne GestioneAmministratore) per validare le richieste a loro carico.
2. La classe controller server a fattorizzare tutto il codice che permette l’interazione con il layer della persistenza
3. L’oggetto Database citato non esiste, serve solo a indicare che all’interno di controller ci sarà un oggetto che permetterà l’interazione con il DBMS.

*Diagramma di dettaglio: Amministratore*

 Il programma di amministrazione è del tutto separato dal sito, in concordanza con il vincolo di massima sicurezza. Tutto il programma con username e password dell’amministratore risiedono sulla stessa macchina. Username e password saranno inseriti all’interno di un file cifrato e accessibile solo da programmi con privilegi di root.

*Diagramma di dettaglio: Client Utente*



I metodi delle view non sono propri. Visto il contesto Web, si è scelto di adottare il pattern risolutivo più diffuso utilizzando pagine JSP per le View. I metodi con i rispettivi parametri sono da intendersi come forward alle pagine con relativo inserimento di quei parametri all’interno del HttpRequest