DOCUMENTO DI PROGETTO

Capogrosso Luigi VR408776
Chiarani Fabio VR408541

Gestionale per Magazzini Sportivi

A.A. 2017 / 2018
Università degli studi di Verona
Elaborato Ingegneria del Software

Indice

1. Specifica Del Progetto

Si vuole progettare un sistema informatico per gestire il magazzino di una catena di negozi di articoli sportivi.

Il negozio vende articoli di diversa tipologia, raggruppati per sport. Per ogni tipo articolo si registra: un nome univoco, una descrizione, lo sport, e i materiali utilizzati per produrlo. Il sistema registra tutti gli articoli in magazzino memorizzando per ogni articolo: il tipo di articolo, un codice univoco, il prezzo e la data di produzione.

Gli articoli in magazzino vengono gestiti dal sistema che registra per ogni ingresso in magazzino: un codice interno univoco, la data e tutti articoli entrati e le loro posizioni in magazzino. Per ogni uscita il sistema registra: la data e il numero di bolla (*univoco*), tutti gli articoli usciti, il negozio che li ha ordinati e lo spedizioniere che li ritira. Per ogni negozio della catena il sistema registra: il codice fiscale, il nome, l'indirizzo e la città.

Il sistema memorizza inoltre gli ordini dei negozi registrando: il negozio che ha effettuato l'ordine, un codice ordine univoco, la data dell'ordine, i tipi di articolo ordinati e per ogni tipo di articolo la quantità` ordinata e il prezzo totale. Quando un ordine viene evaso si registra un'uscita dal magazzino che viene collegata all'ordine al quale si riferisce. Si suppone che per ogni ordine evaso si abbia una sola uscita dal magazzino.
Per ogni tipo di articolo il sistema memorizza esplicitamente alla fine di ogni mese dell'anno la quantità` di articoli ricevuti in magazzino e la quantità` di articoli usciti.

Il sistema deve permettere ai magazzinieri di inserire le informazioni relative ai movimenti di ingresso e uscita dal magazzino. I magazzinieri, inoltre, possono spostare un articolo da una posizione ad un'altra del magazzino, al fine di ottimizzare l'occupazione del magazzino.

La segreteria amministrativa della catena di negozi è responsabile dell'inserimento dei tipi di articolo. Essa può` accedere al sistema e visualizzare i movimenti di magazzino rispetto agli ordini dei vari negozi. Tutti gli utenti sono opportunamente autenticati dal sistema, prima che possano accedere alle funzionalità specifiche. I responsabili dei negozi possono accedere al sistema per effettuare gli ordini e per avere un riassunto degli ordini passati.

2. Introduzione

Il progetto è stato progettato e sviluppato con l'*IDE* di *Netbeans 8.2* con la versione di Java 8. Per il controllo di versione del codice sorgente abbiamo utilizzato i client per GIT *Gitkraken* e *CLI*, inoltre per il database ci siamo appoggiati ai client *MySqlWorkbench* e *CLI*.

Abbiamo presupposto che i dati all'interno del database relativi ai negozi, alcuni contenuti nel magazzino e gli utenti, siano già predisposti dall'amministratore del programma.

Abbiamo predisposto tre tipologie di utenti per poter provare i tre diversi tipi di utenti rivolti al nostro sistema, poiché compiono operazioni diverse:

Utente per la segreteria amministrativa:
- ***Username*:** *fabio*
- ***Password*:** *chiarani*

Utente per i magazzinieri:
- ***Username*:** *luigi*
- ***Password*:** *capogrosso*

Utente responsabile dei negozi:
- ***Username*:** *univr*
- ***Password*:** *univr*

3. Scopo Di Questo Documento

Lo scopo che si prefigge questo documento è quello di spiegare cosa il progetto dovrà offrire, in particolare, la finalità è quella di esplicare i requisiti utilizzati durante la fase di realizzazione.

3.1 Requisiti

I requisiti presentati nella specifica sono tutti di tipo funzionale.

3.2 Utenti

Dall'analisi del progetto sono state individuate tre diverse tipologie di utenti, ognuna delle quali può usufruire di determinate funzionalità del sistema. Nello specifico, gli attori del sistema sono:

- Magazzinieri: utenti con specifiche credenziali ad esso assegnate dalla ditta responsabile del magazzino.
  I magazzinieri hanno la possibilità di inserire le informazioni relative ai movimenti di ingresso e uscita dal magazzino, inoltre, possono spostare un articolo da una posizione ad un'altra dal magazzino, al fine di ottimizzare l'occupazione di quest'ultimo.

- Segreteria amministrativa: utenti con credenziali ad esso assegnate dalla ditta responsabile del magazzino.
  La segreteria amministrativa della catena dei negozi è responsabile dell'inserimento dei tipi di articolo, inoltre essa può anche accedere al sistema e visualizzare i movimenti di magazzino rispetto agli ordini dei vari negozi.

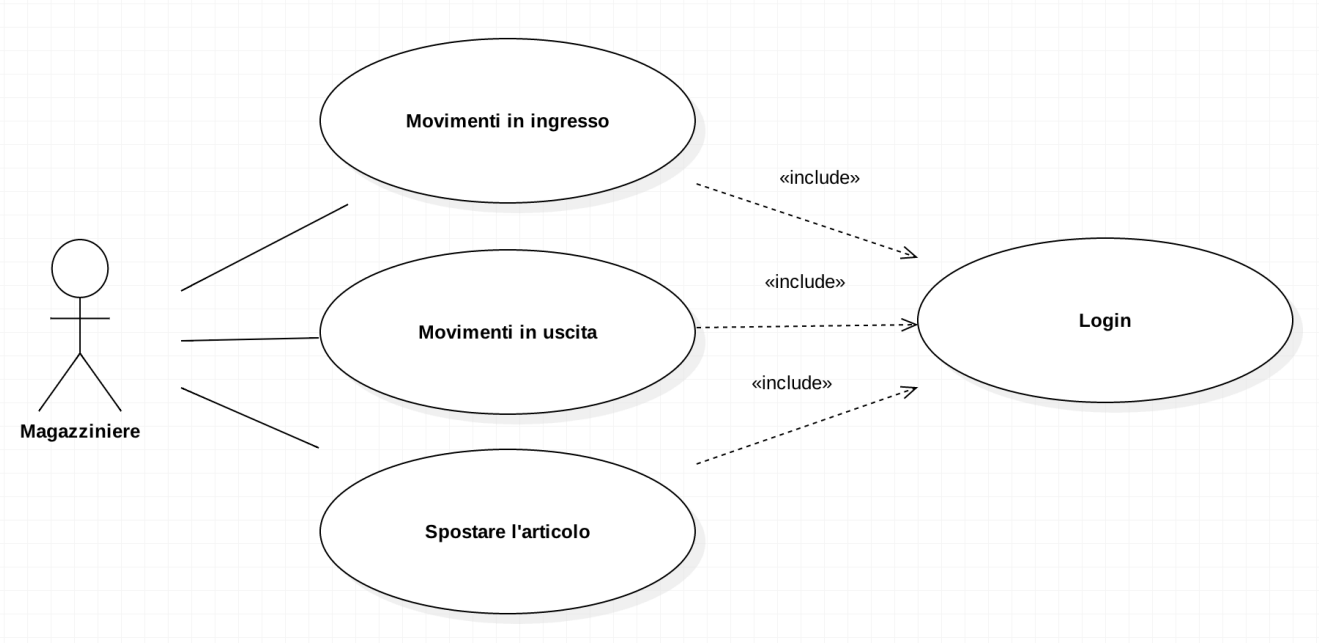
- Responsabili dei negozi: utenti con credenziali ad esso assegnate dalla ditta responsabile del magazzino.
  Quest'ultimi possono accedere al sistema per effettuare gli ordini e per avere un riassunto degli ordini passati.

4 Diagrammi Use Case

Si tratta di, essenzialmente, di una tecnica utilizzata per scoprire, chiarire e concordare i requisiti di un sistema. Utilizzare i casi d'uso significa:

1. Individuare **chi** (*persone, o altri sistemi "esterni"*) dovrà utilizzare il sistema;
2. Chiedersi quali sono gli **obiettivi** che si intendono conseguire usando il sistema;
3. Approfondire, in termini di scenari concreti, ciascuna **modalità d'uso**.

4.1 Caso D'Uso: Magazziniere



Caso d'uso: Movimenti in Ingresso
ID: UC-1
Attori: Magazziniere
Precondizioni: Programma avviato ed utente già registrato e loggato

Sequenza degli eventi:
1. Il caso d'uso inizia quando il magazziniere clicca su "*Insert*";
2. Il sistema controlla i dati inseriti;
  1. Se sono corretti vengono memorizzate le informazioni relative ai movimenti in ingresso;
  2. Se sono errati, dato che il sistema si aspetta delle informazioni corrette, nessun dato verrà memorizzato.

Postcondizioni: Il contenuto del magazzino è stato aggiornato.
Sequenza alternativa: Nessuna

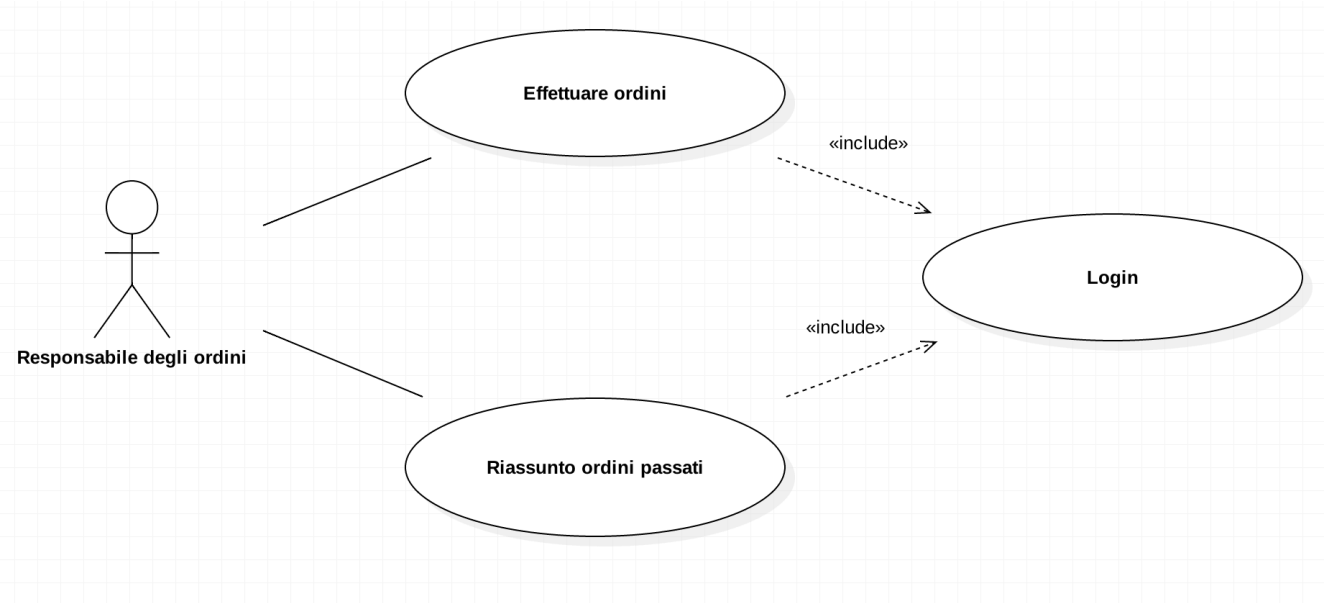
Caso d'uso: Movimenti in Uscita
ID: UC-2
Attori: Magazziniere
Precondizioni: Programma avviato ed utente già registrato e loggato

Sequenza degli eventi:
1. Il caso d'uso inizia quando il magazziniere clicca su "*Remove*";
2. Il sistema controlla i dati inseriti;
  1. Se sono corretti vengono aggiornate le informazioni relative ai movimenti in uscita;
  2. Se sono errati, dato che il sistema si aspetta delle informazioni corrette, nessun dato verrà memorizzato.

Postcondizioni: Il contenuto del magazzino è stato aggiornato.
Sequenza alternativa: Nessuna

Caso d'uso: Spostare l'Articolo
ID: UC-3
Attori: Magazziniere
Precondizioni: Programma avviato ed utente già registrato e loggato
Sequenza degli eventi: 1. Il caso d'uso inizia quando il magazziniere clicca su "*update*"; 2. Il sistema controlla i dati inseriti; 3. Se sono corretti vengono aggiornate le informazioni relative alla posizione degli elementi nei magazzini; 4. Se sono errati, dato che il sistema si aspetta delle informazioni corrette, nessun dato verrà memorizzato.
Postcondizioni: La posizione nel magazzino è stata aggiornata.
Sequenza alternativa: Nessuna

4.2 Caso D'Uso: Responsabile Degli Ordini



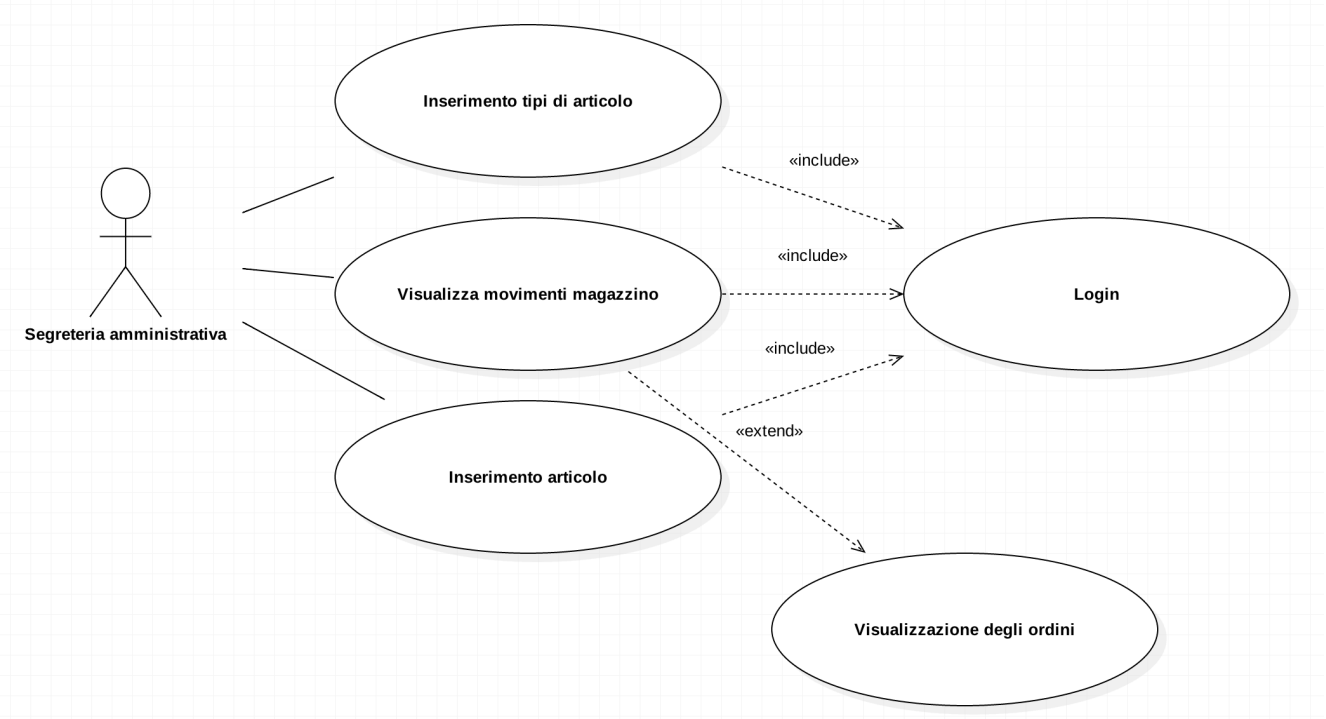
Caso d'uso: Effettuare Ordini
ID: UC-4
Attori: Responsabile degli ordini
Precondizioni: Programma avviato ed utente già registrato e loggato

Sequenza degli eventi:
1. Il caso d'uso inizia quando il responsabile degli ordini clicca su "Order now";
2. Il sistema controlla i dati inseriti;
  1. Se sono corretti vengono aggiornate le informazioni relative agli ordini;
  2. Se sono errati, dato che il sistema si aspetta delle informazioni corrette, nessun dato verrà memorizzato.

Postcondizioni: Il contenuto del magazzino è stato aggiornato.
Sequenza alternativa: Nessuna

Caso d'uso: Riassunto Ordini Passati
ID: UC-5
Attori: Responsabile degli ordini
Precondizioni: Programma avviato ed utente già registrato e loggato
Sequenza degli eventi: 1. Il caso d'uso inizia quando il responsabile degli ordini clicca su "Order history"; 2. Il sistema recupera i dati dal database; 3. Vengono mostrate le informazioni acquisite.
Postcondizioni: Nessuna
Sequenza alternativa: Nessuna

4.3 Caso D'Uso: Segreteria Amministrativa



Caso d'uso: Inserimento Tipi di Articolo
ID: UC-6
Attori: Segreteria amministrativa
Precondizioni: Programma avviato ed utente già registrato e loggato
Sequenza degli eventi: 1. Il caso d'uso inizia quando l'utente della segreteria amministrativa clicca su "Insert" nel pannello adibito all'inserimento della tipologia dell'articolo; 2. Il sistema controlla i dati inseriti; 1. Se sono corretti vengono aggiornate le informazioni relative ai tipi di articoli; 2. Se sono errati, dato che il sistema si aspetta delle informazioni corrette, nessun dato verrà memorizzato.
Postcondizioni: Il contenuto del magazzino è stato aggiornato.
Sequenza alternativa: Nessuna

Caso d'uso: Visualizza Movimenti Magazzini
ID: UC-7
Attori: Segreteria amministrativa
Precondizioni: Programma avviato ed utente già registrato e loggato
Sequenza degli eventi: 1. Il caso d'uso inizia quando il responsabile degli ordini clicca su "Movimenti Magazino"; 2. Il sistema recupera i dati dal database; 3. Vengono mostrate le informazioni acquisite.
Postcondizioni: Nessuna
Sequenza alternativa: Nessuna

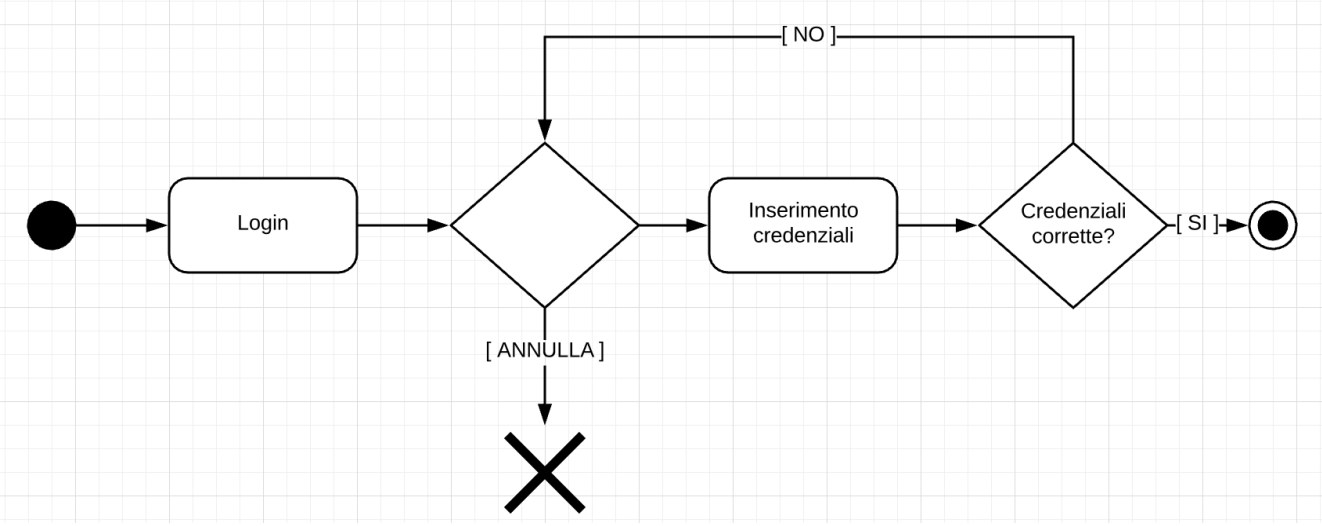
Caso d'uso: Visualizzazione degli ordini
ID: UC-7
Attori: Segreteria amministrativa
Precondizioni: Programma avviato ed utente già registrato e loggato
Sequenza degli eventi: 1. Il caso d'uso inizia quando il responsabile degli ordini clicca su "Movimenti Magazzino"; 2. Il sistema recupera i dati dal database; 3. Vengono mostrate le informazioni acquisite.
Postcondizioni: Nessuna
Sequenza alternativa: Nessuna

Caso d'uso: Inserimento Articolo
ID: UC-8
Attori: Segreteria amministrativa
Precondizioni: Programma avviato ed utente già registrato e loggato
Sequenza degli eventi: 1. Il caso d'uso inizia quando l'utente della segreteria amministrativa clicca su "Insert" nel pannello adibito all'inserimento dell'articolo; 2. Il sistema controlla i dati inseriti; 1. Se sono corretti vengono aggiornate le informazioni relative agli articoli; 2. Se sono errati, dato che il sistema si aspetta delle informazioni corrette, nessun dato verrà memorizzato.
Postcondizioni: Il contenuto del magazzino è stato aggiornato.
Sequenza alternativa: Nessuna

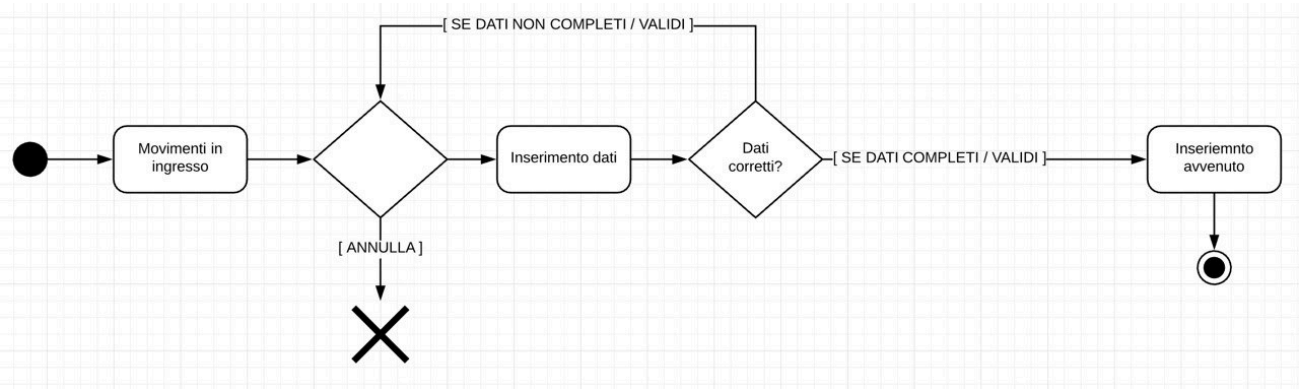
5. Diagrammi Delle Attività

Per entrare più nel dettaglio mostriamo i diagrammi di attività, che descrivono un comportamento più specifico del software come flusso di azioni.

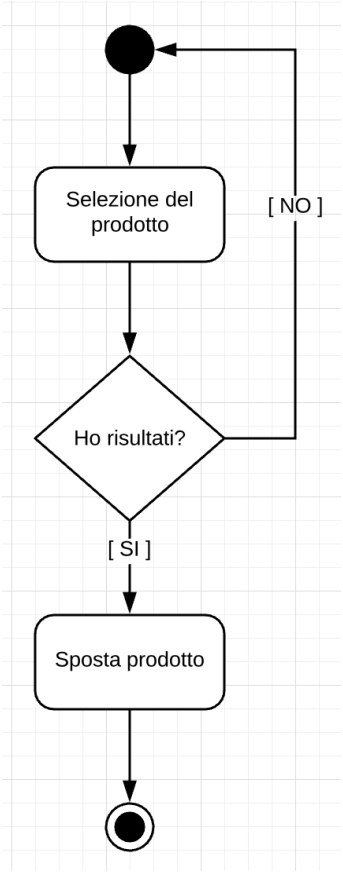
5.1 Diagramma Attività: Login



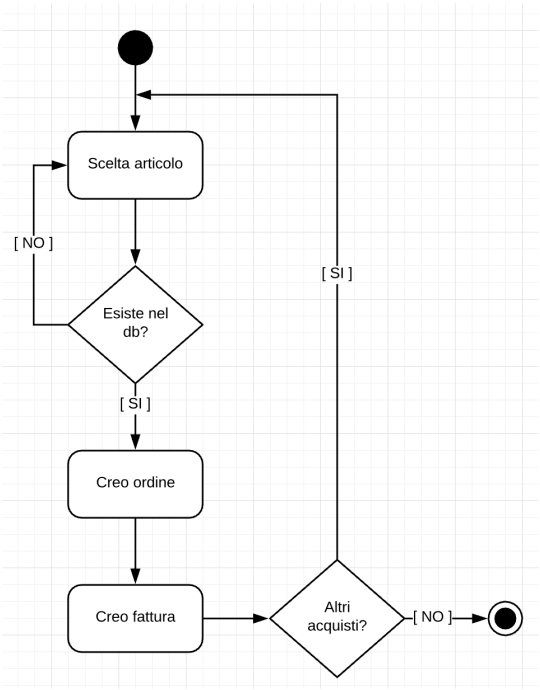
5.2 Diagramma Attività: Movimento In Ingresso



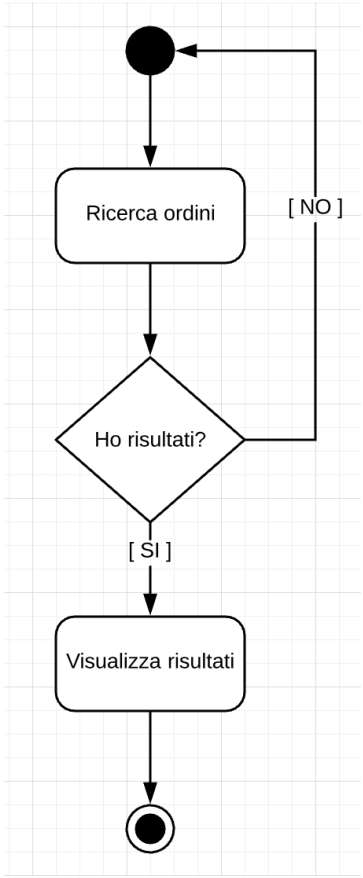
5.3 Diagramma Attività: Spostamento Del Prodotto



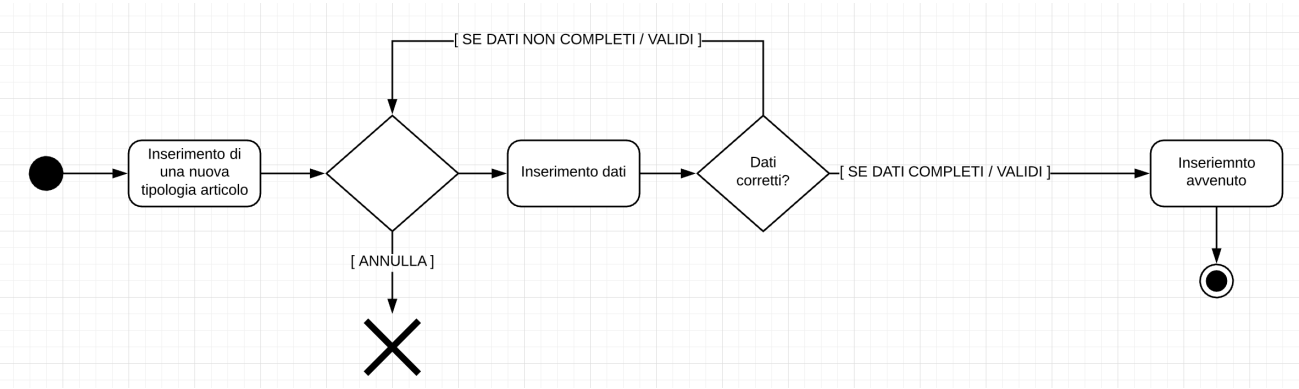
5.4 Diagramma Attività: Creazione Dell'ordine



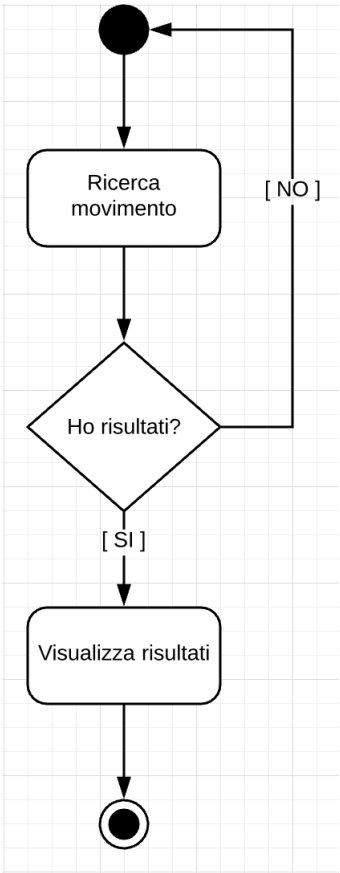
5.5 Diagramma Attività: Riassunto Ordini Passati



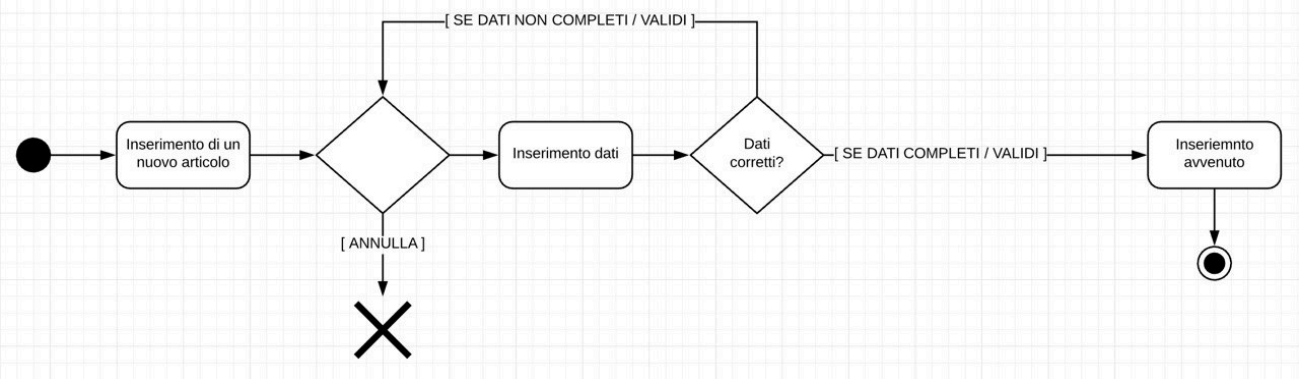
5.6 Diagramma Attività: Inserimento Tipologia Articolo



5.7 Diagramma Attività: Visualizzazione Movimenti Magazzino



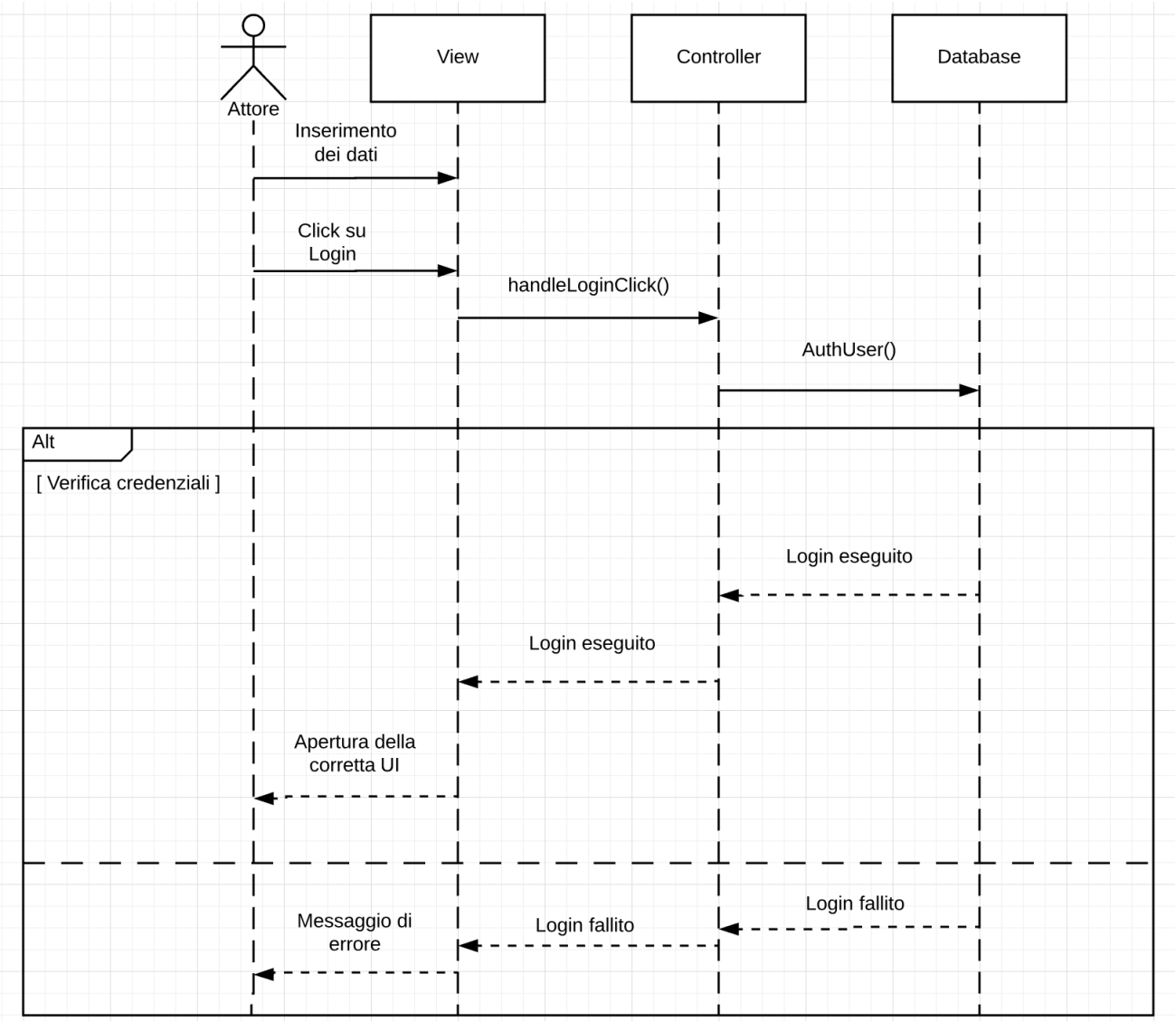
5.8 Diagramma Attività: Inserimento Articolo



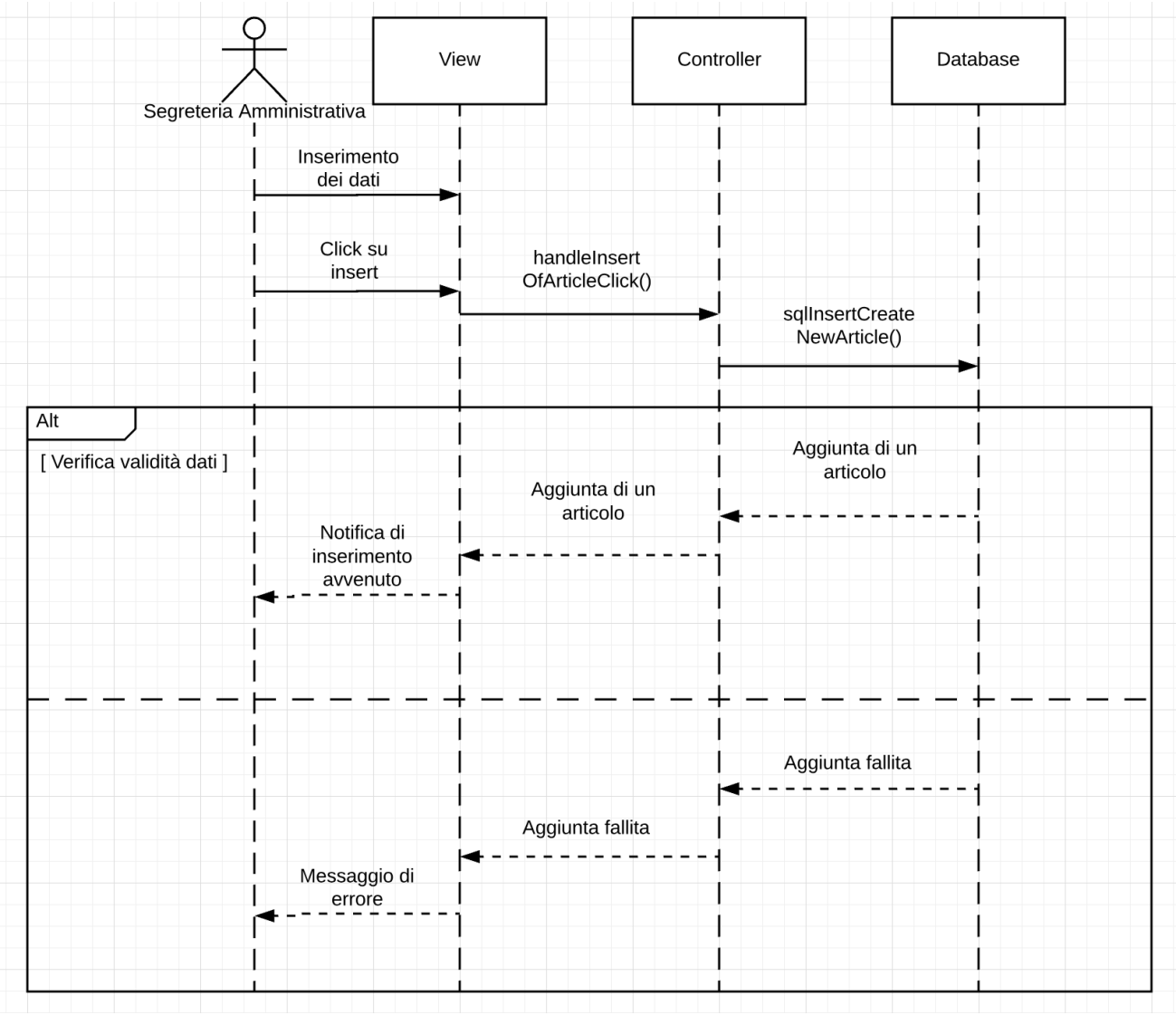
6. Diagrammi Di Sequenza

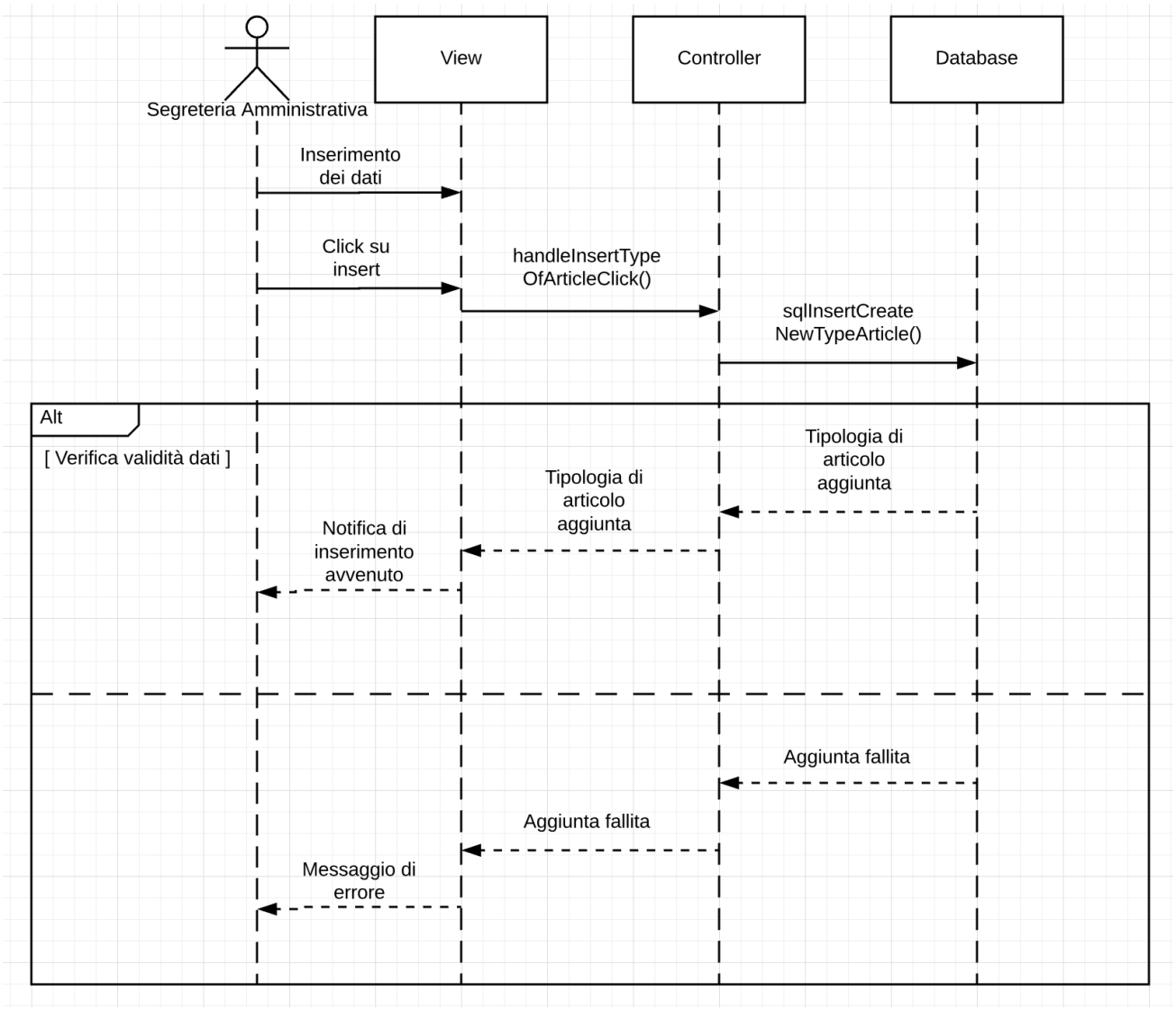
Di seguito, attraverso i digrammi di sequenza, mostriamo come un certo comportamento viene realizzato attraverso la collaborazione delle entità in gioco.

6.1 Diagramma Di Sequenza: Login

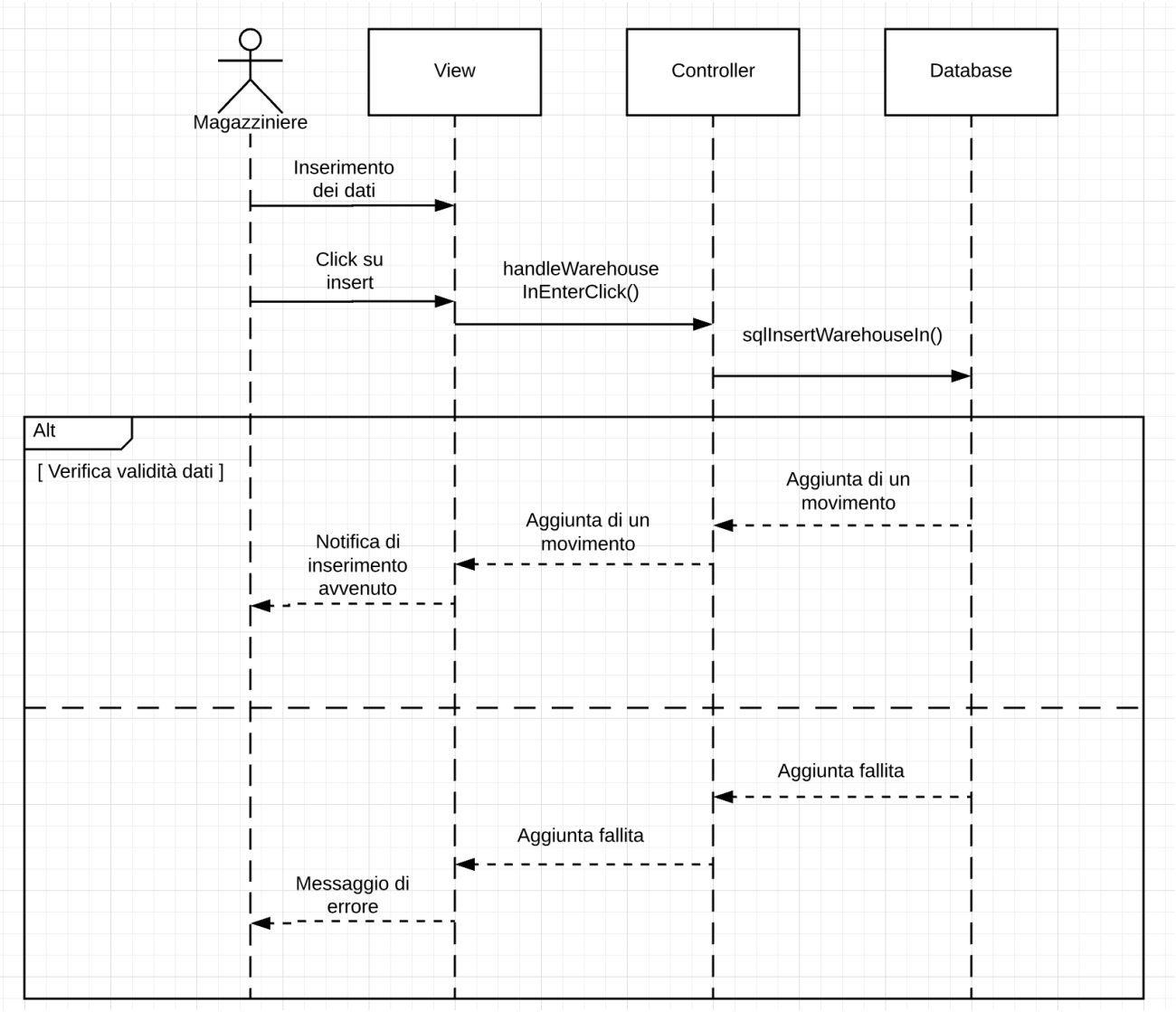


6.2 Diagramma Di Sequenza: Inserimento Articolo e Tipo Articolo

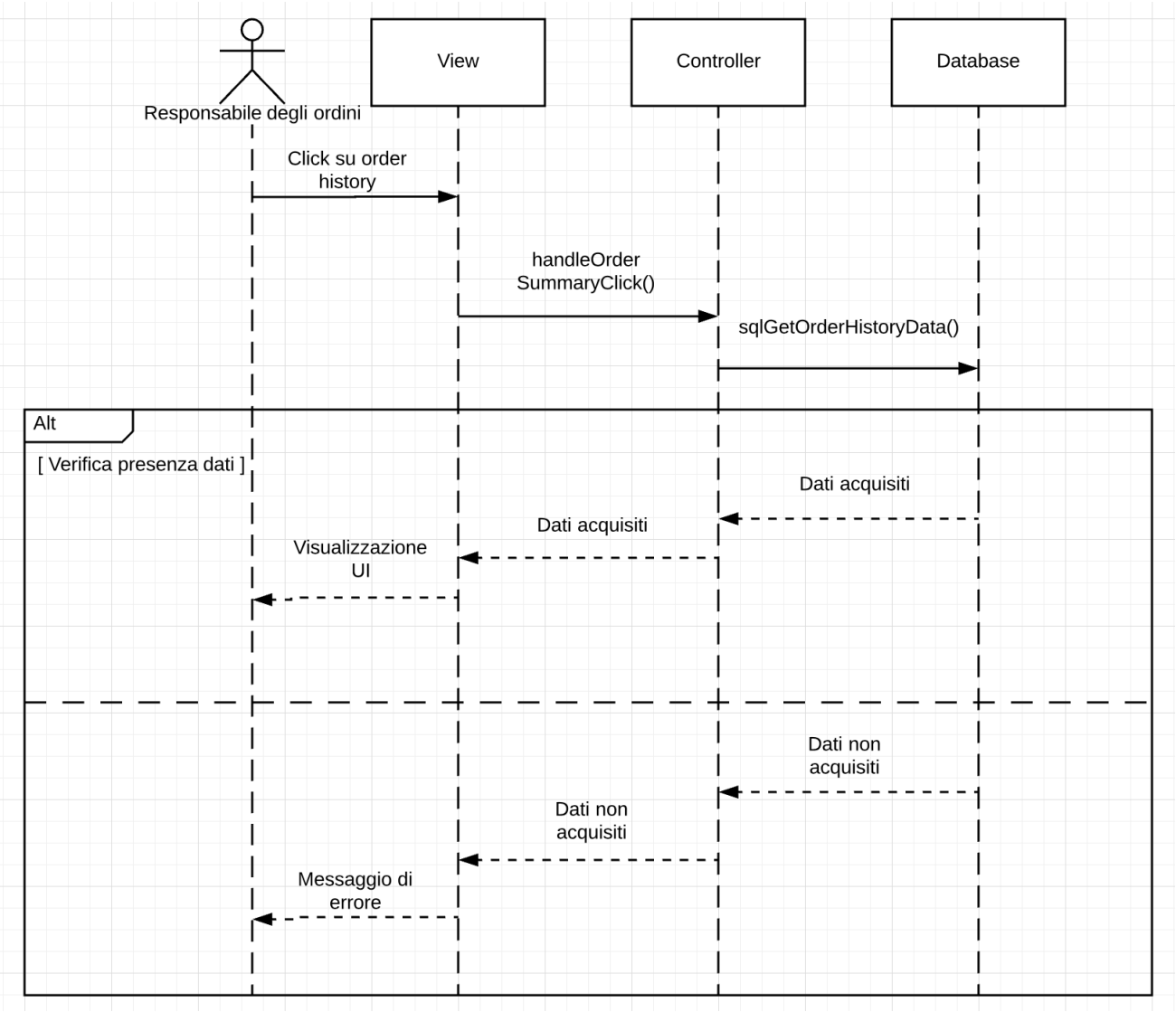



Segreteria Amministrativa
View
Controller
Database
Inserimento dei dati
Click su insert
handleInsertType OfArticleClick()
sqlInsertCreate NewTypeArticle()
Alt
[Verifica validità dati]
Tipologia di articolo aggiunta
Tipologia di articolo aggiunta
Notifica di inserimento avvenuto
Aggiunta fallita
Aggiunta fallita
Messaggio di errore

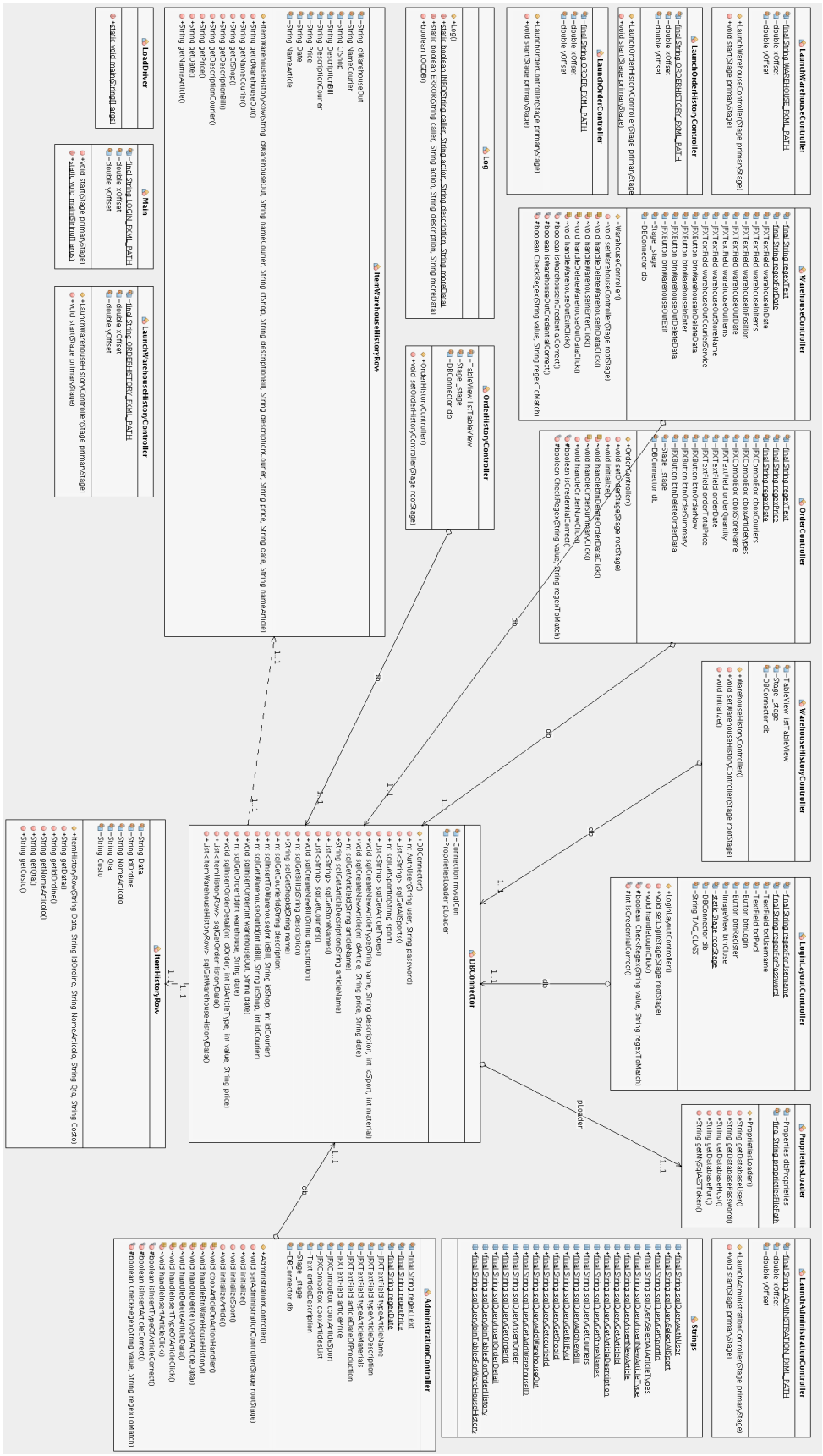
6.3 Diagramma Di Sequenza: Movimento In Ingresso



6.4 Diagramma Di Sequenza: Riassunto Ordini Passati



7. Diagramma Delle Classi



8. Test

In una prima fase sono stati eseguiti test sui singoli componenti ogni qualvolta una nuova parte di codice veniva implementata: sono stati svolti svariati test di inserimento dei dati, ponendo particolare attenzione alla correttezza dei dati limitando al minimo le possibilità` di errore di inserimento da parte dell'utente.

È stato verificato che ogni funzionalità stabilita nei requisiti e implementata nel software sia stata realizzata correttamente secondo le specifiche. Nello specifico è stato verificato:

- Login: dalla form di *Login* è possibile eseguire l'accesso attraverso il proprio username e password (che vengono forniti dalla ditta che gestisce il magazzino) per aprire i diversi pannelli contenenti le operazioni che ogni specifico attore può svolgere.
  Superato: sì.

- Movimenti in ingresso: dalla form abilitata ai soli magazzinieri, deve essere possibile inserire le informazioni relative ai movimenti di ingresso nel magazzino.
  Superato: sì.

- Ottimizzare occupazione magazzino: dalla form abilitata ai soli magazzinieri deve essere possibile spostare un articolo da una posizione ad un'altra nei magazzini.
  Superato: sì.

- Effettuare un ordine: dalla form abilitata ai soli responsabili dei negozi è possibile effettuare un nuovo ordine di un prodotto.
  Superato: sì.

- Riassunto degli ordini: dalla form abilitata ai soli responsabili dei negozi è possibile avere un riassunto degli ordini passati.
  Superato: sì.

- Nuova tipologia di articolo: dalla form abilitata alla sola segreteria amministrativa è possibile aggiungere una nuova tipologia di articolo.
  Superato: sì.

- Movimenti magazzino: dalla form abilitata alla sola segreteria amministrativa deve essere possibile visualizzare i movimenti di magazzino rispetto agli ordini dei vari negozi:
  Superato: sì.

- Nuovo articolo: Dalla form adibita alla sola segreteria amministrativa è possibile aggiungere un nuovo articolo.
  Superato: sì.

In una seconda fase si è deciso di far usare l'applicazione a non informatici (*famigliari, amici e fratelli*) dalla quale sono emerse alcune problematiche che abbiamo risolto, tra le quali:

- Colore della form di *Login* troppo acceso, abbiamo quindi deciso di adottare una tonalità di colore meno intensa.

- Il font all'interno delle form di inserimento dei dati era troppo piccolo

- Ci è stato suggerito di aggiungere i dettagli del negozio durante la visualizzazione dei movimenti degli ordini dalla segreteria amministrativa

9. Junit Test

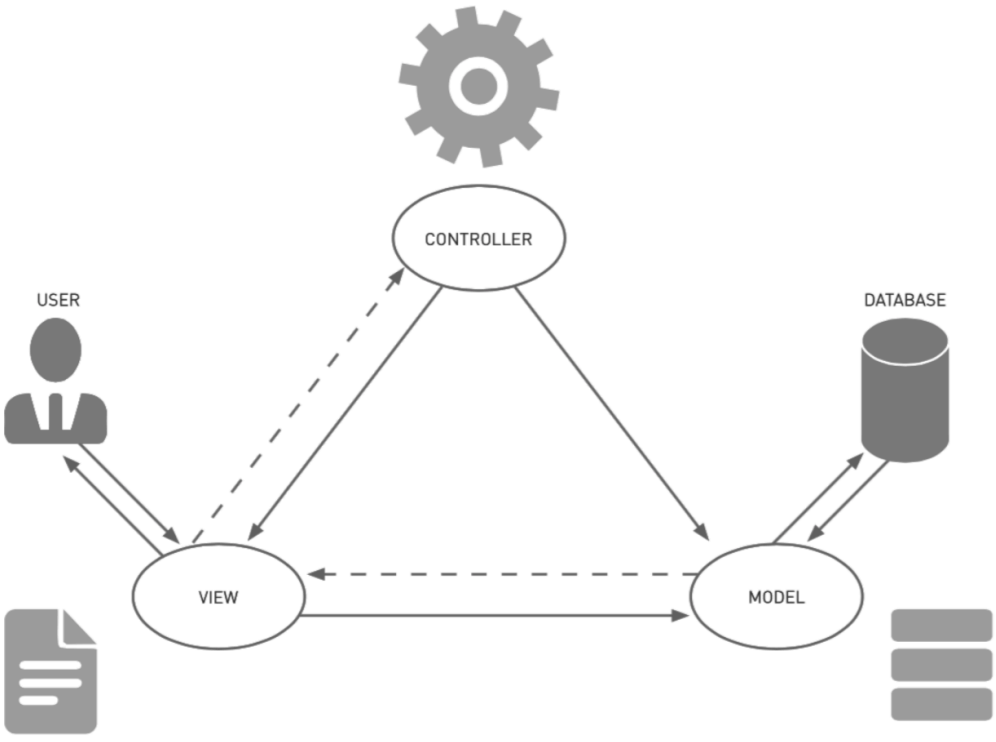
Ci siamo appoggiati alla libreria *JUnit* di java per eseguire dei test software alle funzioni all'interno del nostro codice che abbiamo ritenuto particolarmente critiche, ad esempio:

- È stato controllato che le funzioni di controllo della validità dell'inserimento delle date, ritornassero un valore corrispondente all'*assert* atteso.

10. Pattern Architetturale

I pattern sono schemi utilizzabili nel progetto di un sistema, permettono quindi di non inventare da capo soluzione ai problemi già risolti, ma di utilizzare dei "mattoni" di provata efficacia.

Il seguente progetto è principalmente basato secondo il pattern architetturale ***MVC*** (***M****odel-****V****iew-****C****ontroller*) :



Il progetto può essere visto come una suddivisione in tre principali componenti:

- Il package *Model* si occupa della parte relativa alla memorizzazione dei dati, dove al suo interno sono presenti le classi per poter interagire con il database *MySql*

- Il package *View* si occupa di favorire l'interazione con l'utente, dove al suo interno sono presenti i layout delle varie form.

- Il package *Controller* che si occupa di gestire i comandi degli utenti andando a modificare lo stato delle altre due componenti.

Nel package *Model*, per garantire la creazione di una sola istanza della classe `DBConnector`, è stato utilizzato il pattern creazionale *Singleton*.

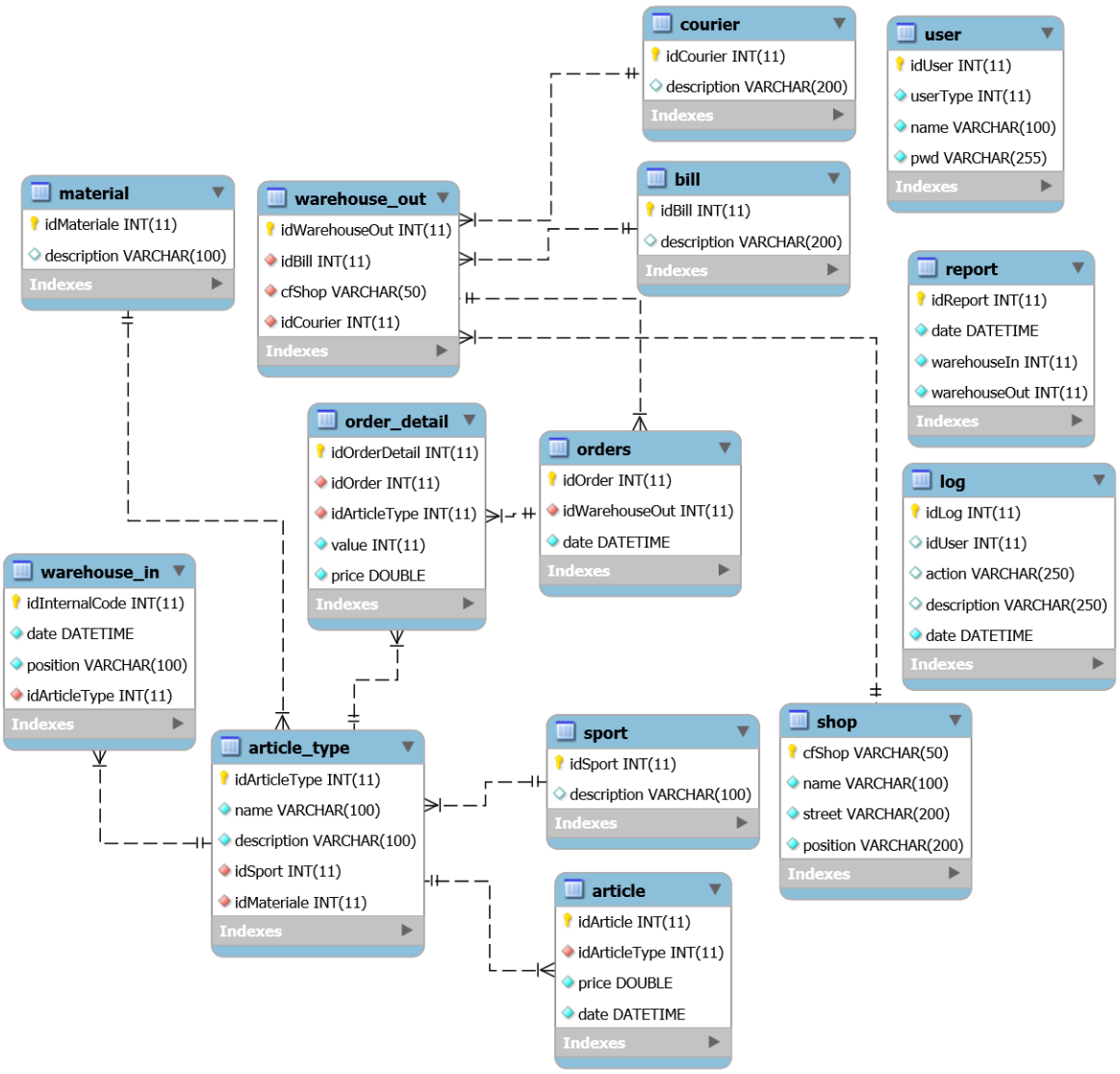
Questa classe serve per creare la connessione con il database: il primo oggetto che richiede l'utilizzo di quest'ultimo crea la connessione vera e propria, la quale verrà passata poi agli altri oggetti che eventualmente ne richiedono l'utilizzo.

11. Database

Il progetto interagisce con un database *MySql*. È stato deciso di adottare l'utilizzo di un database per migliorare l'aspetto della gestione del *Model* all'interno dell'architettura *MVC* da noi adottata. Inoltre, un database ci permetteva di gestire la consistenza, privatezza e affidabilità dei dati. Alcuni vantaggi nell'utilizzo di un database sono stati i seguenti:

- Accesso ai dati tramite un linguaggio universale (*SQL*)
- Accesso efficiente ai dati
- Controllo della ridondanza dei dati
- Atomicità delle operazioni
- Accesso concorrente ai dati
- Privatezza (*suddivisione degli utenti*)
- Affidabilità (*backup dei dati*)

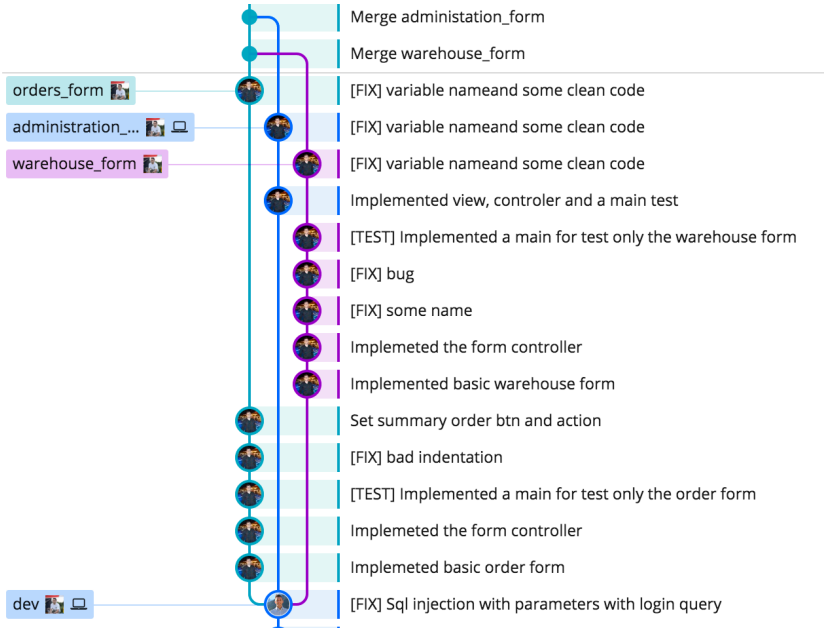
11.1 Diagramma Del Database



12. GIT

Il progetto utilizza lo strumento *GIT*, in particolare *github.com*, per il controllo della versione del codice.
Questo strumento ci ha aiutato nella condivisione del codice e ci ha permesso di strutturarlo su diversi *branch*, e taggati dalle *issues*.

i.e. Creazione dei *branch* per suddividere i compiti delle diverse UI:



i.e. Gestione dei compiti tramite le *issues*: