UNIVERSITA’ DEGLI STUDI DI NAPOLI

***FEDERICO II***

SCUOLA POLITECNICA E DELLE SCIENZE DI BASE

DIPARTIMENTO DI INGEGNERIA ELETTRICA E TECNOLOGIE DELL’INFORMAZIONE



CORSO DI LAUREA IN INFORMATICA

INSEGNAMENTO DI OBJECT ORIENTATION ANNO ACCADEMICO 2019/2020

**Progettazione e sviluppo di un applicativo per la gestione di un negozio di abbigliamento che permetta di gestire vendite e disponibilità di magazzino**

Autori: Docente:

Lorenzo Lama : N86002845 Prof. Sergio di Martino

Aleks Nikolaev Nikolov : N86003002

**Introduzione**

L’applicativo offre varie funzionalità necessarie per la gestione di un negozio di abbigliamento:

* Gestione dell’inventario, tenendo traccia dei diversi articoli in magazzino, con la possibilità di aggiungerne nuovi
* Gestione di vendite e rifornimenti, permettendo di effettuare nuove transazioni ed eliminare quelle vecchie
* Gestione dei fornitori da cui si effettuano i rifornimenti, e la possibilità di aggiungerne nuovi

Per lo sviluppo del progetto è stata utilizzata la seguente repository: <https://github.com/aleks-nikolov/Progetto-OO-BD>

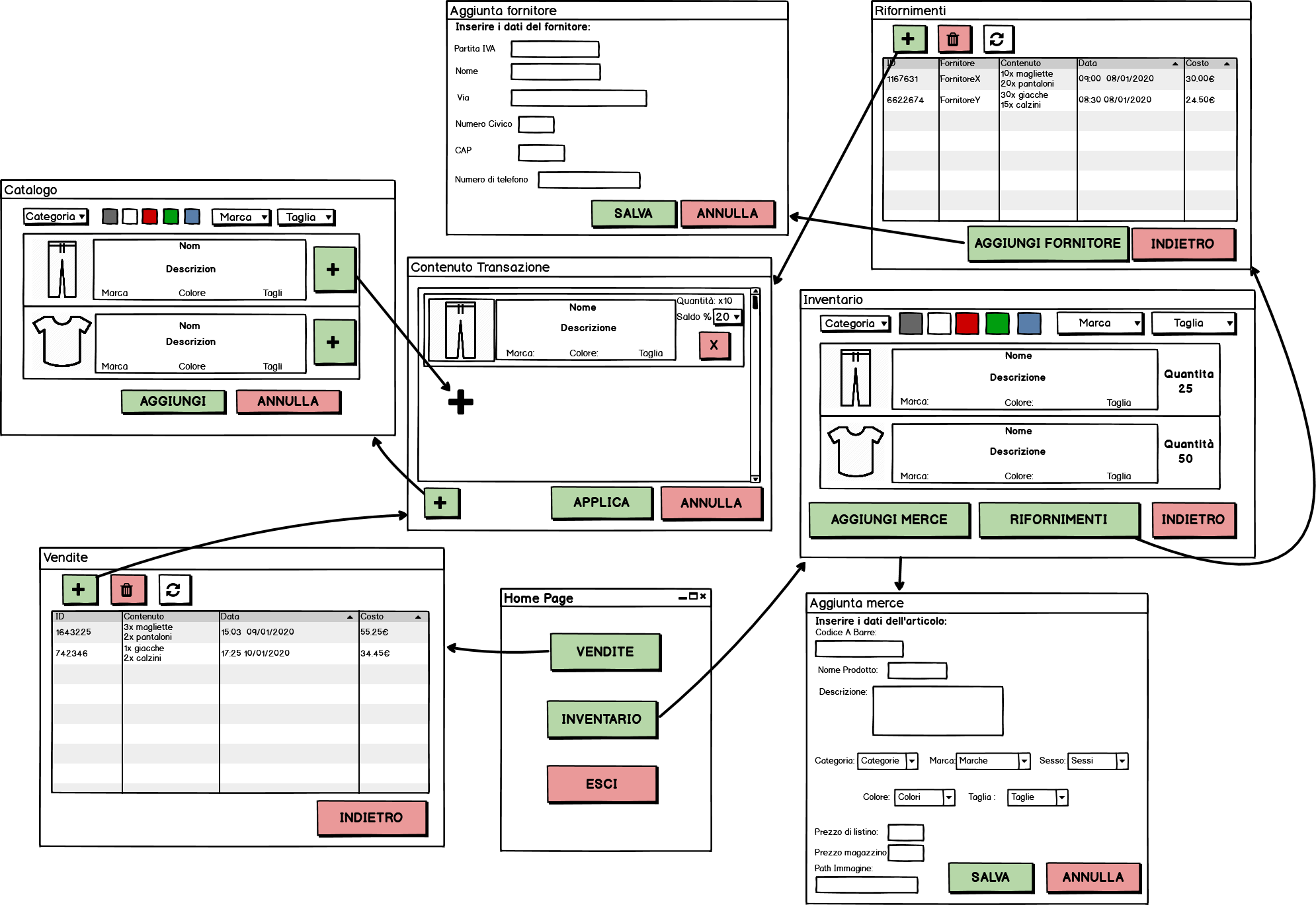
Il software è stato sviluppato con il JDK 8, utilizzando la libreria Java Swing per la creazione dell’interfaccia grafica.

Viene effettuata una connessione con una base di dati implementata con PostgreSQL 12 e hostata online sulla piattaforma ElephantSQL. Il driver che permette la connessione al DBMS si trova nella cartella *lib* del codice sorgente.

L’applicativo fa uso di immagini per la rappresentazione degli articoli. Tali immagini possono essere trovate nella cartella *res* insieme alle icone utilizzate dal software.

**Mockup**

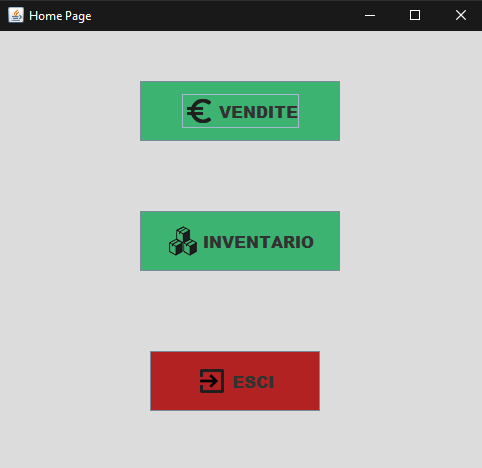
La seguente immagine rappresenta il mockup iniziale, che descrive le interazioni base tra i diversi frame. Un’immagine per intero può essere trovata nella cartella contente il PDF.

  
In seguito verranno brevemente descritte le funzionalità di ogni frame che fa parte del programma.

**Home Page**

È la prima finestra che viene mostrata all’avvio del programma.

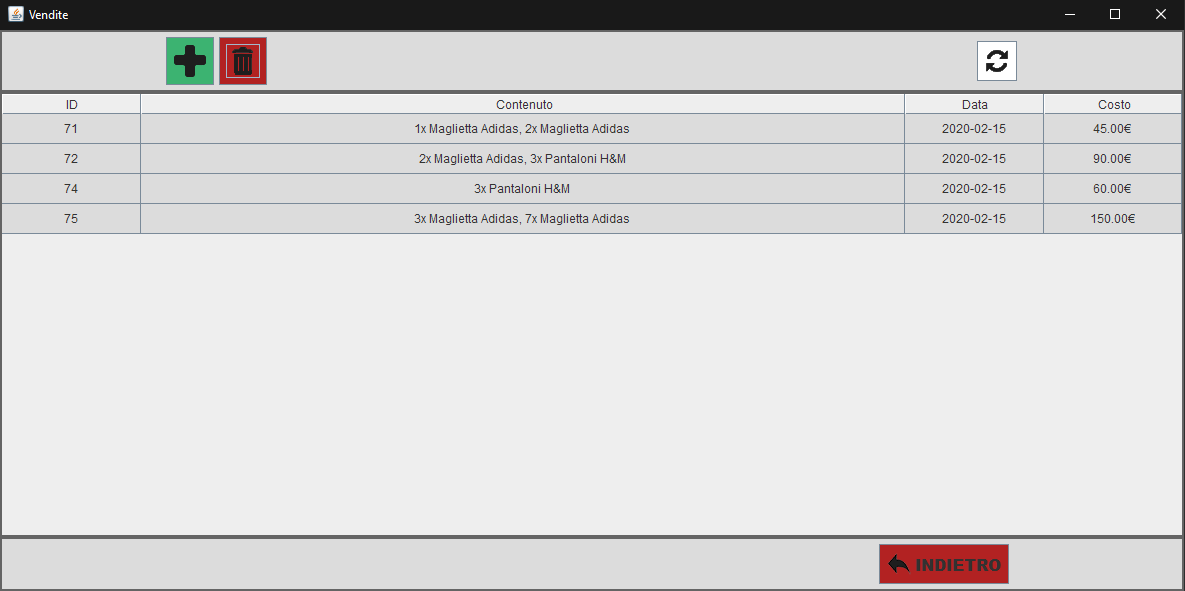
Permette l’accesso a ***finestraVendite*** e ***finestraInventario***.



**FinestraVendite**

Mostra i dati di tutte le vendite effettuate in una JTable.

Permette di aggiungere una nuova vendita, aprendo così ***contenutoVendita.*** Inoltre, consente di eliminare vendite precedentemente effettuate.

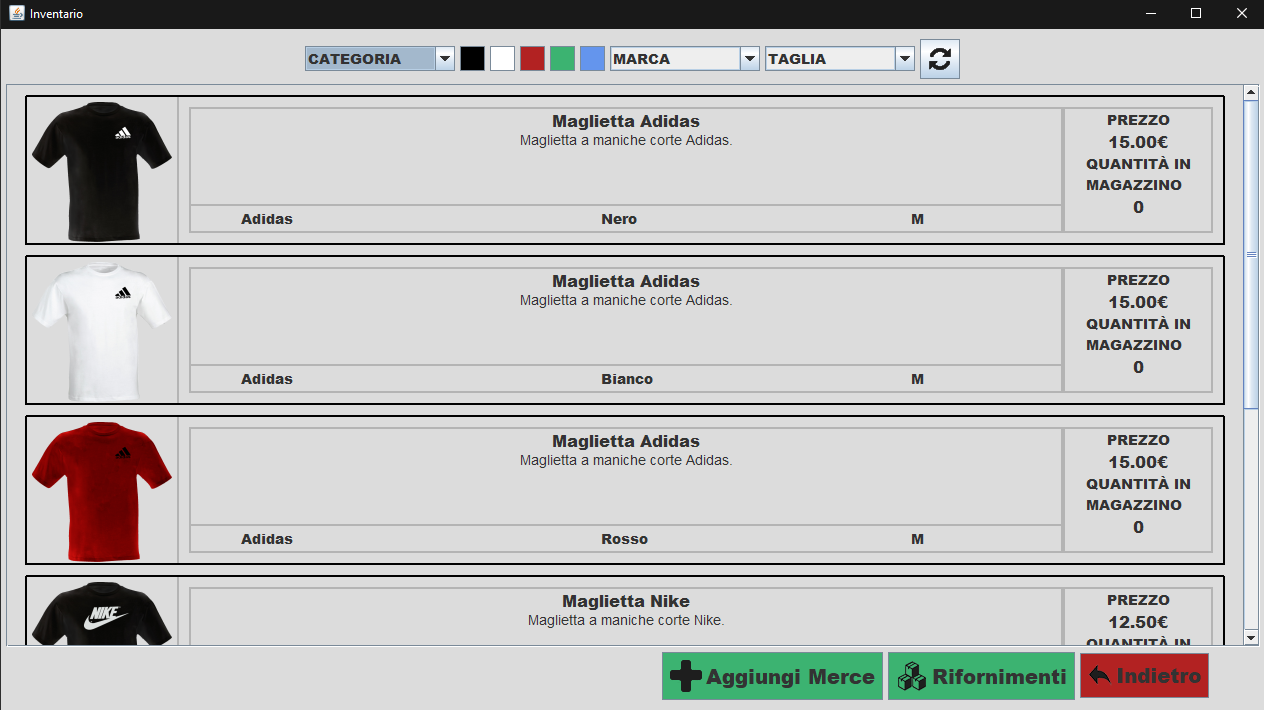


**FinestraInventario**

Mostra gli articoli presenti in magazzino, con le relative descrizioni e la loro quantità. Tali articoli vengono inseriti dentro a una serie di oggetti ***ContenitoreInventario***, che vengono aggiunti ad un JScrollPane.

Nella parte superiore è presente un sistema di filtraggio, che permette di cercare gli articoli desiderati.

Permette l’accesso a ***finestraRifornimenti,*** oppure all’apertura di uno form per l’inserimento di nuova merce in inventario.

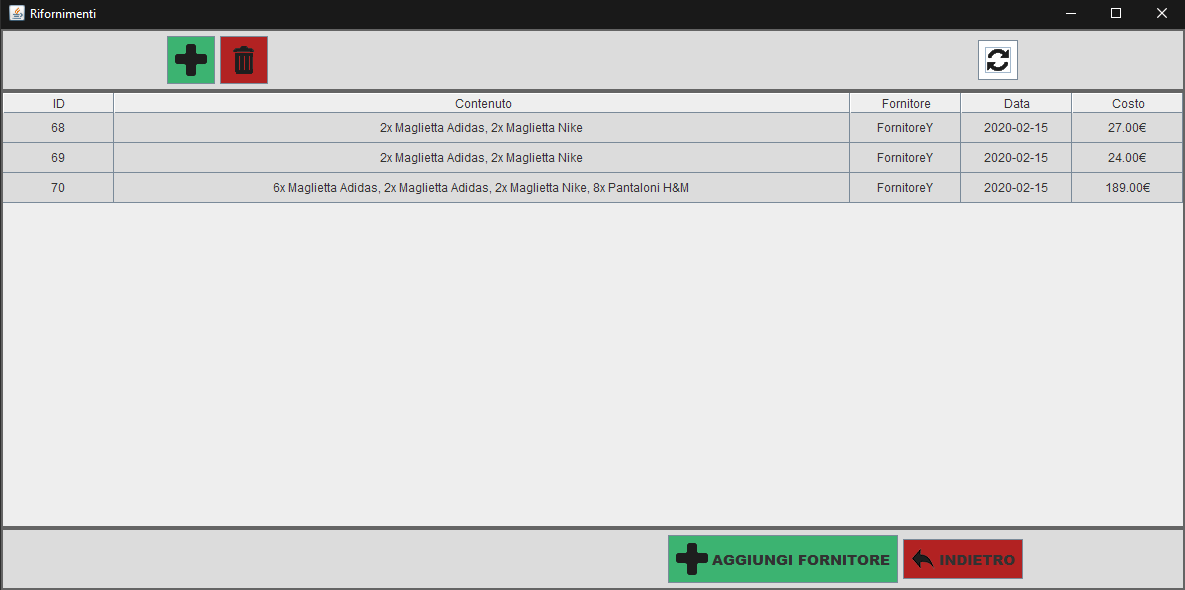


**FinestraRifornimenti**

Simile a ***finestraVendite***, mostra tutti i rifornimenti effettuati.

Permette di aggiungere un nuovo rifornimento, aprendo così ***contenutoRifornimento.*** Inoltre, consente di eliminare rifornimenti precedentemente effettuati.

Permette di aprire uno form per l’aggiunta di un nuovo fornitore.



**ContenutoTransazione**

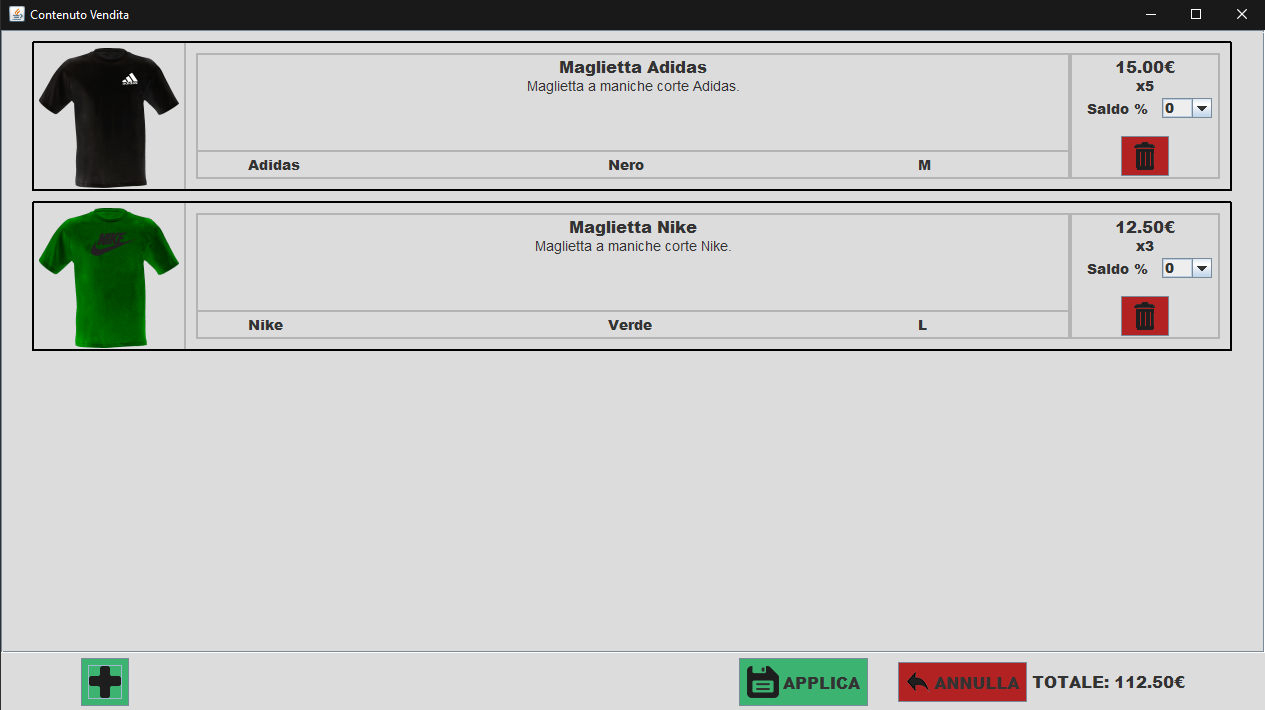
È un frame che può assumere due forme diverse, a seconda dal frame che lo apre:

* ***contenutoVendita***, se viene chiamato da ***finestraVendite***
* ***contenutoRifornimento***, se viene chiamato da ***finestraRifornimenti***

In ogni caso conterrà un insieme di JPanel ***ContenitoreTransazione*** nei quali verranno rappresentati gli articoli facenti parte della transazione da creare.

Nel caso in cui si stia effettuando un rifornimento, verrà mostrata una JComboBox per la scelta del fornitore.

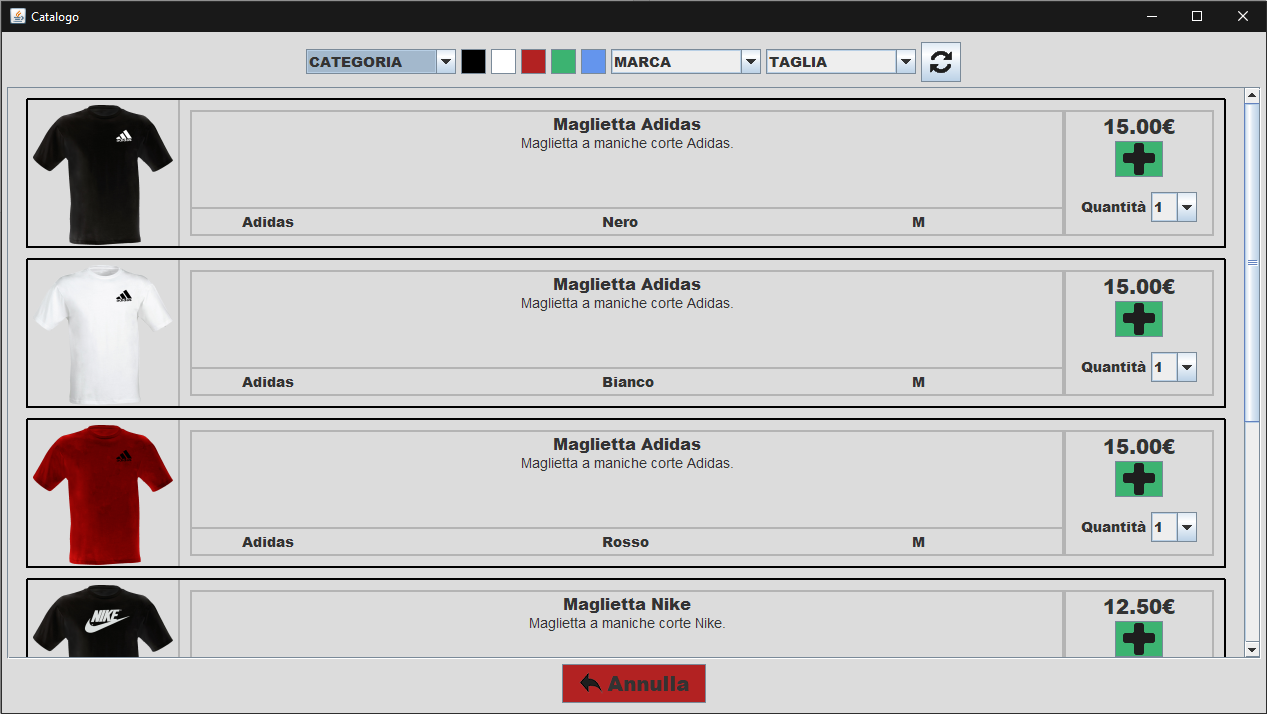
Permette l’accesso alla finestra ***catalogo***.



**Catalogo**

Mostra gli articoli che il negozio mette a disposizione sotto forma di oggetti ***ContenitoreCatalogo.***

Permette di aggiungere una quantità di un certo articolo al ***ContenutoTransazione***che l’ha chiamato.



**AggiuntaArticolo e AggiuntaFornitore**

Sono due form che permettono l’inserimento di un nuovo articolo o fornitore nel database.

Entrambi effettuano un controllo di validità dei campi di testo prima di effettuare l’inserimento, e mostrano un messaggio d’avviso nel caso in cui le informazioni inserite non sono valide.



**Class Diagram**

**Premessa**

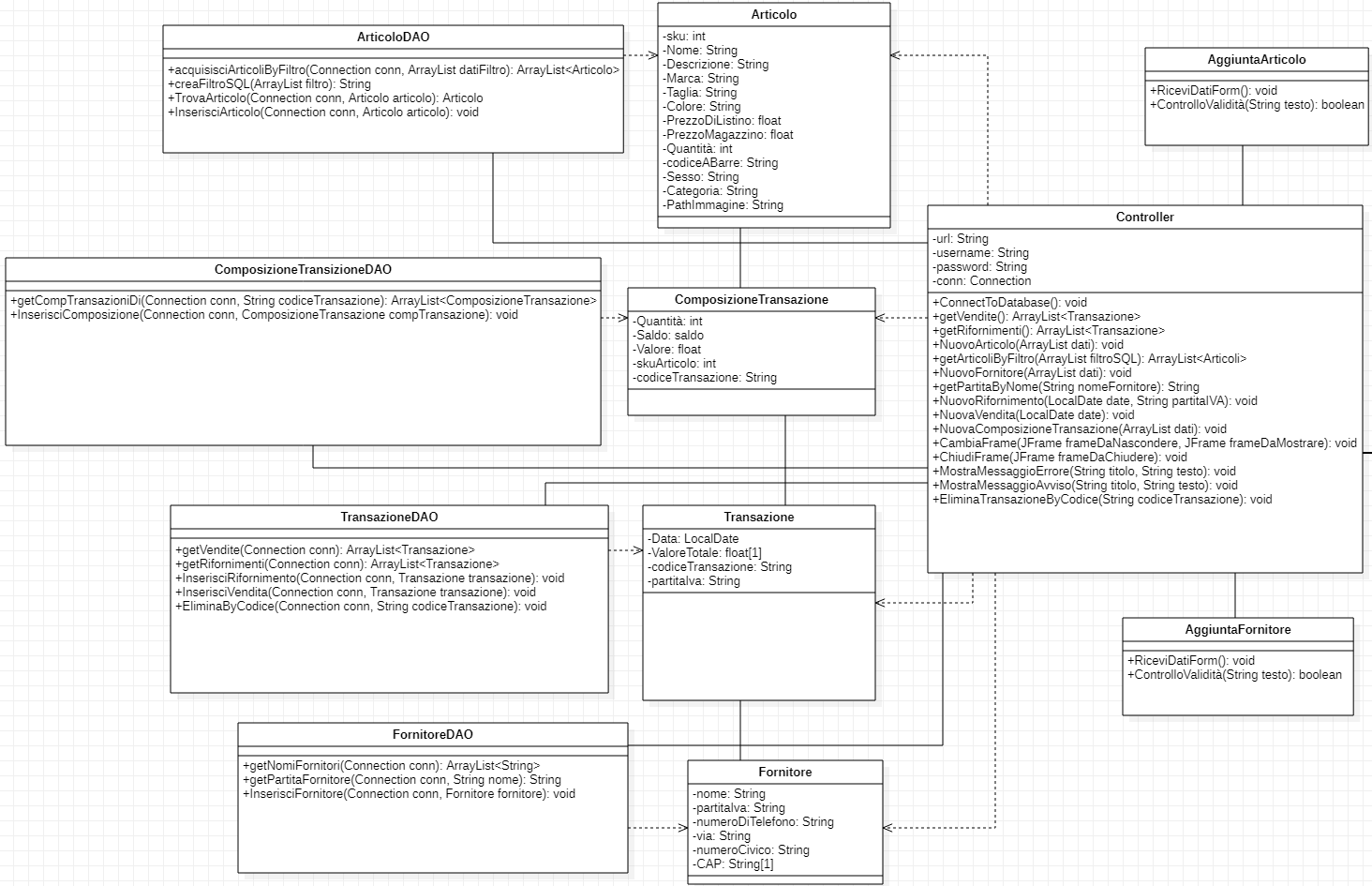
Da notare che nella cartella del PDF è presente un’immagine dell’intero diagram di qualità migliore.

Per evitare l’ingombro del diagram, verranno ommesse le seguenti informazioni:

* metodi getter, setter ed equals
* i componenti grafici all’interno delle classi GUI
* metodi che contengono codice generato dall’editor WindowBuilder, come ***ImpostaFinestra(), ImpostaPannello()***, ecc...
* metodi ***AggiungiListener()*** all’interno delle classi GUI, che si occupano di aggiungere gli ActionListener per i diversi JButton
* la classe ***Driver*** che contiene il metodo ***main()***, il cui unico compito è istanziare il ***Controller***

**Parte di logica**

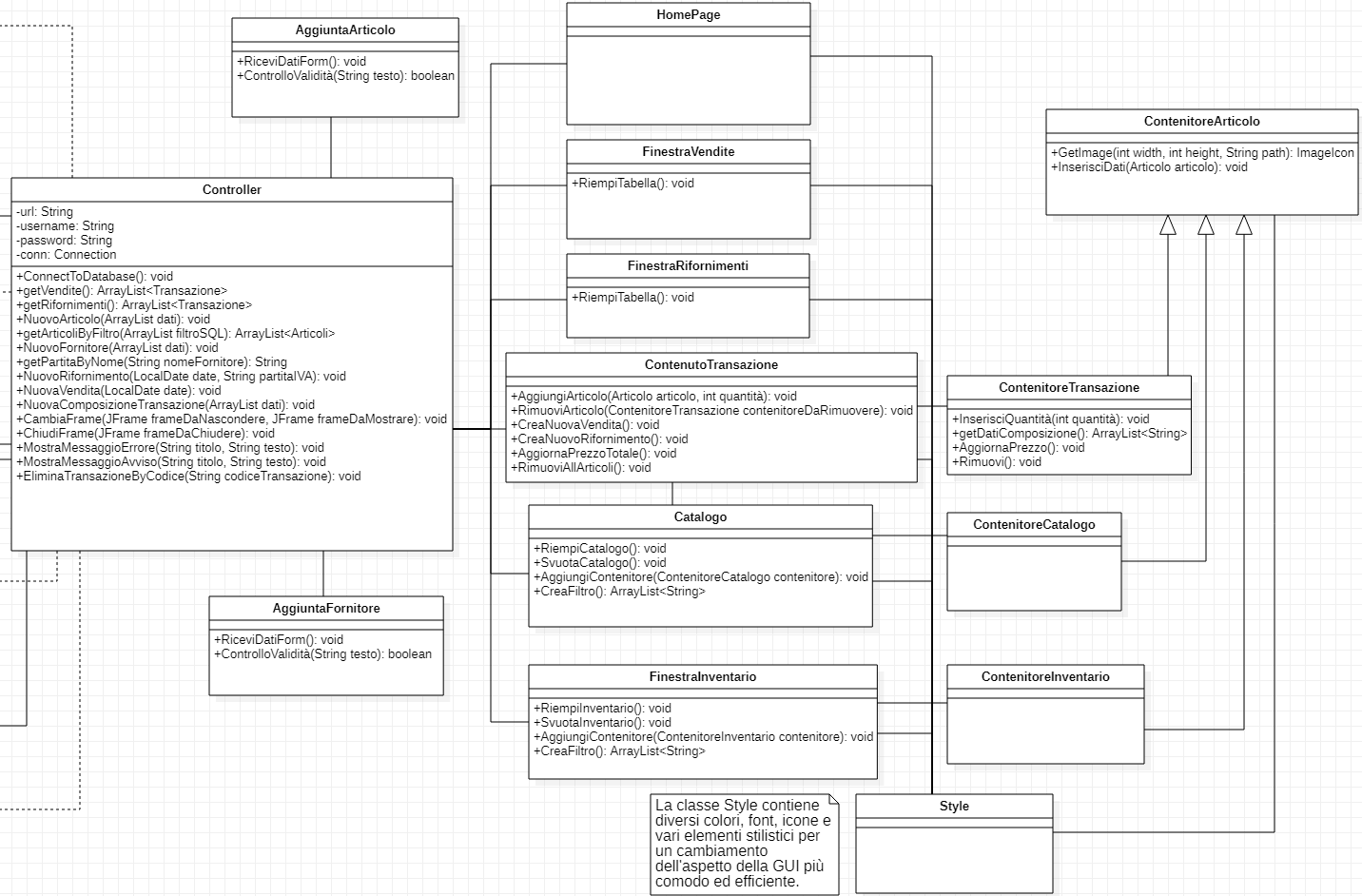
Questa porzione di class diagram contiene tutte le classi che si trovano nel pacchetto ***logic*** e che riguardano funzionalità come la manipolazione dei dati a livello logico, la connessione e l’accesso alla base di dati.



**Parte di interfaccia grafica**

Questa porzione di class diagram contiene tutte le classi che si trovano nel pacchetto ***gui***.

Tutte queste classi sono dei JFrame, tranne ***ContenitoreArticolo*** e le sue sottoclassi che sono dei JPanel, e la classe ***Style***, contenente alcuni oggetti stilistici.

****

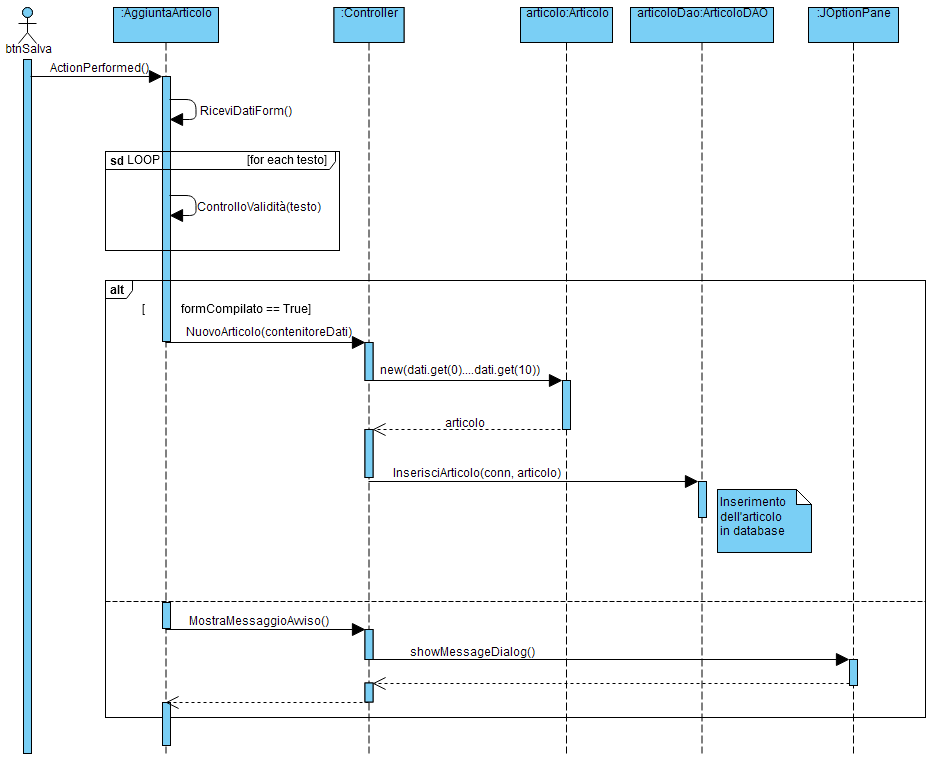
**Sequence Diagrams**

In seguito sono presentati i sequence diagram che descrivono il funzionamento di alcuni metodi fondamentali del software.

**Inserimento di un nuovo articolo in magazzino**

Questa serie di eventi viene eseguita quando viene premuto il JButton ***btnSalva*** in ***AggiuntaArticolo***.

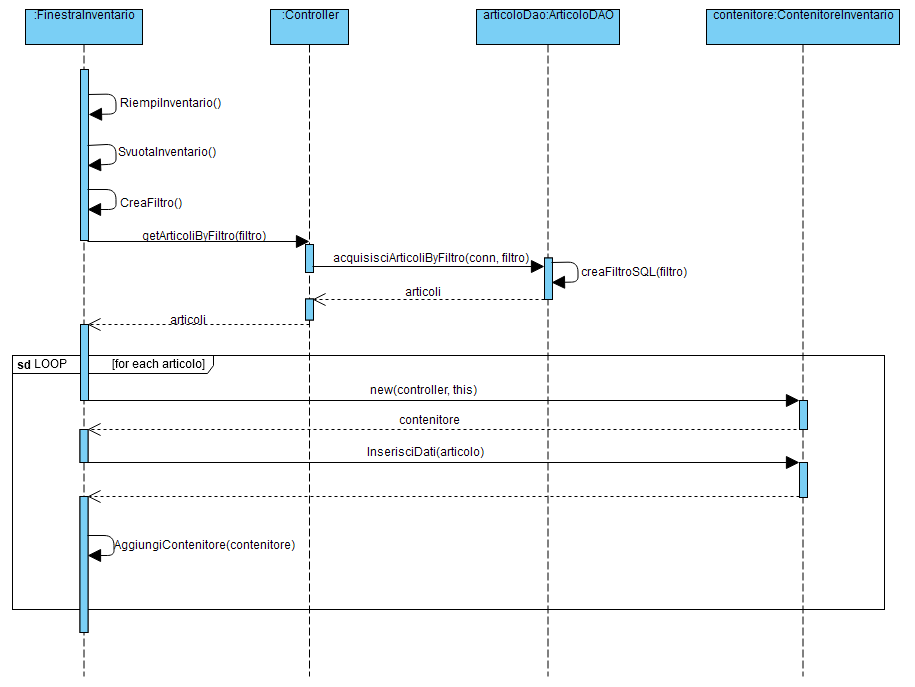
I dati del form vengono ricevuti e sottoposti ad un controllo, dopo di che vengono salvati in un oggetto di tipo ***Articolo*** e salvati in database dalla classe ***ArticoloDAO***. Se i dati non sono validi, viene mostrato un messaggio d’avviso.

****

**Riempimento della finestraInventario con gli articoli in magazzino**

Quando viene istanziato un oggetto di tipo ***FinestraInventario*** viene fatta una richiesta ad ***ArticoloDAO*** di prelevare gli articoli dalla base di dati che rispettano le condizioni di un filtro. Una volta ottenuti gli articoli desiderati, viene eseguito un ciclo for each che riempe la finestra con i ***ContenitoreInventario***contenenti i dati degli articoli.

La funzione ***RiempiInventario()*** può anche essere chiamata quando viene premuto il JButton ***btnRefresh***.



**Inserimento di una nuova transazione di tipo vendita**

Una volta scelti gli articoli da aggiungere ad una transazione e premendo il JButton ***btnApplica*** in ***ContenutoTransazione***, viene chiamato il metodo ***CreaNuovaVendita()*** che provede all’inserimento di una nuova vendita attraverso ***TransazioneDAO***, utilizzando come parametro la data odierna prelevata da ***LocalData***.

