Documentazione progetto

Ingegneria del software

## Contro Filippo VR397847 - Martini Michele VR398611

Specifiche:

Si vuole progettare un sistema informativo per gestire le informazioni relative alla gestione di un negozio virtuale di CD e DVD musicali (vende solo via web).

Il negozio mette in vendita CD di diversi generi: jazz, rock, classica, latin, folk, world-music, e così via. Per ogni CD o DVD il sistema memorizza: un codice univoco, il titolo, i titoli di tutti i pezzi contenuti, eventuali fotografie della copertina, il prezzo, la data dalla quale è presente sul sito web del negozio, il musicista/band titolare, una descrizione, il genere del CD o DVD, i musicisti che vi suonano, con il dettaglio degli strumenti musicali usati. Per ogni musicista il sistema registra il nome d’arte, il genere principale, l’anno di nascita, se noto, gli strumenti che suona.

Sul sito web del negozio è illustrato il catalogo dei prodotti in vendita. Cliccando sul nome del prodotto, appare una finestra con i dettagli del prodotto stesso. I clienti possono acquistare on-line selezionando gli oggetti da mettere in un “carrello della spesa” virtuale. Deve essere possibile visualizzare il contenuto del carrello, modificare il contenuto del carrello, togliendo alcuni articoli. Al termine dell’acquisto va gestito il pagamento, che può avvenire con diverse modalità. Il sistema supporta differenti ricerche: per genere, per titolare del CD o DVD, per musicista partecipante, per prezzo. Coerentemente, differenti modalità di visualizzazione, sono altresì supportate. Ogni vendita viene registrata indicando il cliente che ha acquistato, i prodotti acquistati, il prezzo complessivo, la data di acquisto, l’ora, l’indirizzo IP del PC da cui è stato effettuato l’acquisto, la modalità di pagamento (bonifico, carta di credito, paypal) e la modalità di consegna (corriere, posta, ...). Per ogni cliente il sistema registra: il suo codice fiscale, il nome utente (univoco) con cui si è registrato, la sua password, il nome, il cognome, la città di residenza, il numero di telefono ed eventualmente il numero di cellulare. Per i clienti autenticati, il sistema propone pagine specializzate che mostrano suggerimenti basati sul genere dei precedenti prodotti acquistati. Se il cliente ha fatto già 3 acquisti superiori ai 250 euro l’uno entro l’anno, il sistema gli propone sconti e consegna senza spese di spedizione.

Il personale autorizzato del negozio può inserire tutti i dati dei CD e DVD in vendita. Il personale inserisce anche il numero di pezzi a magazzino. Il sistema tiene aggiornato il numero dei pezzi a magazzino durante la vendita e avvisa il personale del negozio quando un articolo (CD o DVD) scende sotto i 2 pezzi presenti in magazzino.

**Attori e casi d'uso**

Sono stati individuati due attori che interagiscono col sistema: i clienti e il personale autorizzato. I clienti sono coloro che visitano il catalogo, acquistano prodotti e ricevono le offerte personalizzate; il software deve perciò essere semplice ed intuitivo, graficamente accattivante, ma allo stesso tempo deve guidare l'utente nell'inserimento di tutti i dati su acquisti e profili di cui il sistema necessita. Il personale autorizzato invece ha bisogno di uno strumento che permetta di monitorare lo stato del magazzino, avendo tutte le informazioni principali visualizzate insieme, per cogliere subito eventuali situazioni in cui intervenire.

Cliente

Per il cliente sono stati individuati tre casi d'uso principali: la visita del catalogo, la gestione del carrello personale e l'acquisto.

Diagramma: UCD Cliente catalogo

|  |  |
| --- | --- |
| UCD 1 | Visita catalogo |
| Precondizioni: | Il catalogo è visibile |
| Sequenza degli eventi | Il cliente visualizza il contenuto del catalogo.  Il sistema ordina i prodotti secondo una caratteristica di default.  In caso di suggerimenti o sconti, compare il pulsante per visualizzarli. |
| Sequenza 1 | Il cliente seleziona un prodotto |
| Post-condizioni 1 | Accesso ai dettagli del prodotto |
| Sequenza 2 | Il cliente seleziona il tasto di ricerca |
| Post-condizioni 2 | Accesso alla ricerca dei prodotti |
| Sequenza 3 | Il cliente seleziona il tasto di ordinamento  Seleziona i criteri di ordinamento |
| Post-condizioni 3 | Il catalogo è riordinato secondo i nuovi criteri |

|  |  |
| --- | --- |
| UCD 1.1 | Visualizza dettagli prodotto |
| Precondizioni: | Il cliente ha selezionato il prodotto di cui vuole conoscere i dettagli |
| Sequenza degli eventi | Il sistema mostra all'utente i dettagli del prodotto selezionato  Vengono mostrate anche le azioni che l'utente piò compiere sul prodotto |
| Sequenza 1 | Il cliente decide di aggiungere il prodotto al carrello |
| Post-condizioni 1 | Accesso alla schermata dedicata |
| Sequenza 2 | Il cliente chiude la pagina del dettaglio |
| Post-condizioni 2 | Il sistema chiude la schermata e torna a quella precedente (Catalogo o carrello) |

|  |  |
| --- | --- |
| UCD 1.1.1 | Aggiungi a carrello |
| Precondizioni: | Il cliente ha selezionato il prodotto che vuole inserire nel carrello |
| Sequenza degli eventi | Il sistema verifica la disponibilità del prodotto in magazzino  Il cliente inserisce la quantità di articoli che intende acquistare |
| Sequenza alternativa | Il cliente annulla l'azione di aggiunta al carrello |
| Post-condizioni | Il cliente torna alla pagina del catalogo |

|  |  |
| --- | --- |
| UCD 1.2 | Ordina prodotti |
| Precondizioni: | Il cliente ha selezionato il riordino del catalogo |
| Sequenza degli eventi | Il sistema propone le opzioni di ordinamento: genere, autore, prezzo... Il cliente sceglie quella che più lo aggrada |
| Post-condizioni | Il sistema riordina il catalogo e aggiorna la pagina |

|  |  |
| --- | --- |
| UCD 1.3 | Ricerca prodotti |
| Precondizioni: | Il cliente ha selezionato il tasto per la ricerca |
| Sequenza degli eventi | Il sistema mostra la finestra di ricerca L'utente seleziona i parametri desiderati  Il sistema filtra il catalogo secondo questi ultimi  Compare il tasto per reimpostare i parametri e annullare la ricerca |
| Sequenza alternativa | L'utente annulla la ricerca |
| Post-condizioni | Il cliente torna alla pagina del catalogo |

|  |  |
| --- | --- |
| UCD 1.4 | Visualizza suggerimenti |
| Precondizioni: | Il cliente è autenticato. In caso contrario viene rimandato alla pagina di log-in  Il cliente ha selezionato il tasto per visualizzare suggerimenti e sconti |
| Sequenza degli eventi | Il sistema elabora e visualizza la pagina con gli sconti e i suggerimenti personalizzati per l'utente  Se non è interessato può tornare alla pagina del catalogo |
| Post-condizioni |  |

La gestione di questi casi d'uso può essere espressa in un unico Sequence Diagram. All'inizio l'utente chiede il catalogo, che a sua volta chiede l'elenco dei prodotti a al magazzino. Una volta ordinati possono essere presentati all'utente. A questo punto il programma continua a proporre delle opzioni all'utente inerenti al catalogo. Se l'utente desidera il dettaglio di un articolo il magazzino provvede a fornire l'informazione, e in caso l'utente può aggiungerlo al carrello. In alternativa può visualizzare suggerimenti e sconti, calcolati dal catalogo sulla base del profilo dell'utente. Infine può ricercare prodotti o riordinare l'elenco; questa procedura è gestita dal catalogo.

Diagramma SD Visita catalogo

Il diagramma delle attività invece è semplice, in quanto il sistema deve solo ottenere i dati dal magazzino e mostrarli all'utente. Ogni altro aspetto (ricerca, riordino, ricerca suggerimenti, visualizzazione dettagli) è simile, in quanto è il sistema che richiede dati al magazzino e li mostra ordinati all'utente

Diagramma AD Visita catalogo

La seconda serie di casi d'uso è inerente al carrello e a tutte le operazioni eseguibili

Diagramma UCD Cliente carrello

|  |  |
| --- | --- |
| UCD 2 | Visualizza carrello |
| Precondizioni: | L'utente seleziona il proprio carrello |
| Sequenza degli eventi | Il sistema mostra una pagina di riepilogo coi prodotti presenti nel carrello  Al cliente vengono offerte diverse opzioni quali:   1. Rimuovere l'articolo 2. Visualizzare i dettagli di un articolo 3. Procedere col pagamento |
| Post-condizioni 1 | Il sistema mostra la schermata di conferma |
| Post-condizioni 2 | Il cliente passa alla schermata di dettaglio |
| Post-condizioni 3 | Il cliente passa alla schermata di pagamento |

|  |  |
| --- | --- |
| UCD 2.1 | Rimuovi prodotto |
| Precondizioni: | Il cliente ha selezionato il tasto di rimozione  Il sistema ha già acquisito il prodotto da eliminare |
| Sequenza degli eventi | Il sistema chiede conferma di rimozione all'utente  Se l'utente conferma il sistema rimuove l'articolo dal carrello |
| Post-condizioni | L'utente torna alla schermata del carrello |

Per questi casi d'uso non sono stati fatti diagrammi di attività o di sequenza in quanto le operazioni di visualizzazione sono già incluse nei diagrammi di pagamento

Il terzo gruppo di casi d'uso, infine, tratta del pagamento e della gestione degli utenti registrati

Diagramma UCD Cliente Pagamento

|  |  |
| --- | --- |
| UCD 3 | Pagamento |
| Precondizioni: | Il carrello non è vuoto |
| Sequenza degli eventi | Se il cliente non ha effettuato l'accesso viene reindirizzato alla pagina di registrazione/log-in  Viene mostrato un riepilogo del carrello  Il cliente inserisce gli estremi del pagamento in un'apposita schermata  Dopo aver rivisto il riepilogo dei dati da la conferma |
| Post-condizioni | Il sistema verifica i dati e se corretti procede con l'ordine, registrandolo e aggiornando il magazzino |

|  |  |
| --- | --- |
| UCD 3.1 | Accesso utente |
| Precondizioni: | Il cliente desidera autenticarsi |
| Sequenza degli eventi | Se il cliente non è registrato viene reindirizzato alla pagina di registrazione  Il cliente immette le credenziali  Il sistema verifica i dati e mostra una schermata di benvenuto |
| Post-condizioni | Il cliente viene reindirizzato alla pagina di provenienza, ma resta autenticato col suo profilo |

|  |  |
| --- | --- |
| UCD 3.1.1 | Registrazione utente |
| Precondizioni: | Il cliente desidera registrarsi |
| Sequenza degli eventi | Il sistema mostra un form per la registrazione  L'utente inserisce i propri dati  Il sistema controlla la validità dei dati e aggiorna l'archivio utenti |
| Post-condizioni | Il cliente è registrato e autenticato |

|  |  |
| --- | --- |
| UCD 3.2 | Seleziona dati pagamento |
| Precondizioni: | Il cliente sta completando un acquisto |
| Sequenza degli eventi | Il sistema mostra un form per l'inserimento dei dati  L'utente inserisce i dati  Il sistema ne controlla la validità |
| Post-condizioni | Il cliente torna alla pagina di acquisto  Il sistema ha acquisito i dati del pagamento |

La Sequenza di attività necessarie per la procedura di pagamento involvono, oltre al cliente, il magazzino, il carrello e il sistema interno, in particolare la sezione che si occupa del profilo personale e dell'accesso. Se il cliente non è registrato dovrà comunicare con l'archivio profili per crearne uno e ricevere i dati di log-in. A questo punto potrà autenticarsi, visualizzare il riepilogo del carrello e inserire i dati di pagamento, di cui il sistema controllerà la validità. Dopo la conferma il sistema aggiornerà il magazzino.

SD Pagamento

Le attività per conseguire il pagamento prevedono l'accesso dell'utente, con la registrazione nel caso non avesse un profilo e l'autenticazione. Dopodiché devono essere completate due attività prima della conferma, la visualizzazione del riepilogo del carrello e l'inserimento dei dati di pagamento. È implicito il controllo della validità dei dati e l'aggiornamento del magazzino.

AD Pagamento

Personale autorizzato

Anche per il personale autorizzato sono stati individuati 3 casi d'uso principali, ossia la visualizzazione del riepilogo del magazzino, la creazione o modifica di un CD e il rimpinguamento delle scorte. Tutte queste azioni prevedono una forte interazione col sistema e in particolare con la gestione dei dati.

Diagramma UCD Personale autorizzato

Il primo è la visualizzazione dello stato del magazzino, in cui l'addetto deve avere una chiara visione della situazione e deve essere notificato in caso di giacenze inferiori a 2. E' stato scelto inoltre di implementare una funzione di ricerca per aiutare l'addetto nella ricerca di un particolare prodotto.

|  |  |
| --- | --- |
| UCD 1 | Visualizza stato magazzino |
| Precondizioni: | Il personale autorizzato è entrato nel sistema |
| Sequenza degli eventi | Il sistema mostra il riepilogo dei prodotti e la relativa giacenza  Il sistema notifica all'addetto se ci sono prodotti con giacenza scarsa  Il sistema mette a disposizione tutti gli strumenti di cui il gestore necessita |
| Post-condizioni | Il cliente ha una chiara visione d'insieme, così come la possibilità di utilizzare tutte le funzionalità del software |

|  |  |
| --- | --- |
| UCD 1.1 | Ricerca |
| Precondizioni: | Il cliente desidera cercare un determinato articolo |
| Sequenza degli eventi | Il sistema mostra una finestra per l'inserimento dei parametri  L'utente li immette  Il sistema filtra l'elenco prodotti |
| Post-condizioni | L'utente ha una visione ristretta ai risultati della ricerca |

I diagrammi di sequenza di queste azioni prevedono azioni già incluse nell'aggiunta / modifica dei CD: Nella parte iniziale l'addetto interroga il magazzino per avere la lista dei prodotti. La ricerca è semplicemente un'interrogazione in cui i risultati sono filtrati. Anche il diagramma di attività è molto simile, in quanto l'azione da eseguire è una sola. Per un maggiore livello di dettaglio vedere i diagrammi del prototipo.

AD Visualizza magazzino

Il secondo caso d'uso comprende l'aggiunta o modifica di CD. Questa azione implica la modifica del magazzino.

|  |  |
| --- | --- |
| UCD 2 | Aggiungi o modifica CD |
| Precondizioni: | Il gestore vuole aggiungere un nuovo CD o ne ha selezionato uno da modificare |
| Sequenza degli eventi | Il sistema mostra la schermata di inserimento dati  Se si tratta di una modifica precompila già i campi coi valori attuali del CD da modificare  L'utente inserisce i dati e conferma  Il sistema controlla i dati e se validi aggiorna il magazzino e la pagina di riepilogo |
| Post-condizioni | Il cliente torna alla visualizzazione del magazzino aggiornata |

|  |  |
| --- | --- |
| UCD 2.1 | Modifica magazzino |
| Precondizioni: | L'utente ha eseguito un'operazione di modifica che prevede l'aggiornamento del magazzino |
| Sequenza degli eventi | Il sistema aggiorna il magazzino con i nuovi dati  Tutte le schermate si aggiornano per mostrare i dati modificati |
| Post-condizioni | L'utente ha una vista aggiornata dello stato del magazzino |

La sequenza di attività prevede il caricamento dei dati attuali nel caso di una modifica, l'inserimento dei nuovi dati da parte dell'addetto, l'aggiornamento del magazzino e della vista dell'utente sullo stesso.

SD Nuovo CD

Il diagramma delle attività prevede l'inserimento dei dati, il controllo della validità, la ricerca di eventuali duplicati e infine l'inserimento della quantità desiderata.

AD Nuovo CD

L'ultimo caso d'uso è il rifornimento o la rimozione del CD. Dopo essere stato selezionato il prodotto e l'azione desiderata il sistema la esegue aggiornando poi il magazzino e la vista che ne ha l'utente.

|  |  |
| --- | --- |
| UCD 3 | Rifornisci o rimuovi CD |
| Precondizioni: | Il cliente ha selezionato il CD su cui operare |
| Sequenza 1 | Il cliente clicca sul pulsante di rifornimento  In una schermata apposita inserisce la quantità desiderata  Dopo la conferma il sistema verifica i dati e provvede a notificare errori o ad aggiornare il magazzino |
| Sequenza 2 | Il cliente clicca sul pulsante di rimozione  Il sistema chiede all'utente la conferma della decisione  In caso positivo provvede ad aggiornare il magazzino |
| Post-condizioni | Il cliente torna alla visualizzazione del magazzino aggiornata |

Il diagramma di sequenza prevede ancora una volta uno scambio di messaggi tra addetto al magazzino e sistema: l'utente dopo aver visto lo stato del magazzino seleziona il prodotto e l'azione da eseguire (set quantità o rimozione) e trasmette queste informazioni al sistema che aggiorna i dati e la vista dell'utente.

SD Rifornimento

Anche il diagramma delle attività è molto simile, in quanto si tratta di richiedere i dati, selezionare l'operazione ed eseguirla sul magazzino

AD Rifornimento

Nota: Per la rimozione non sono stati fatti diagrammi perché sarebbero stati una ripetizione di quanto fatto per il rifornimento.

Dopo l'analisi delle specifiche è stato realizzato uno schema di classi generico per la gestione dei dati. Le classi principali sono Musicista, CD, Cliente e pagamento; per ognuno di questi vengono memorizzate le informazioni richieste. Ogni CD contiene un musicista come autore, e più di uno come partecipante; Il pagamento invece è associato ad un Utente e ai dati di un carrello. Dopodiché vi è una classe Archivio, che raccoglie l'insieme di musicisti, generi, strumenti, utenti e pagamenti; una classe Magazzino, che raccoglie l'insieme di CD con la relativa giacenza; infine una classe Carrello, associata ad un utente, contente dei CD con la relativa quantità acquistabile.

**Implementazione del prototipo**

Per questo progetto è stato implementato un prototipo del software che l'addetto al magazzino utilizzerebbe per gestirlo. Sono state implementate:

1. Una funzione di log-in, da rifinire una volta concordato con lo stakeholder il metodo di autenticazione del personale autorizzato
2. La vista d'insieme dello stato del magazzino, con tutte le informazioni principali visibili, le notifiche per giacenze scarse e la possibilità di vedere i dettagli dei prodotti cliccandoci sopra
3. Una funzione di ordinamento della lista (per titolo, autore, prezzo e giacenza) e una funzione di ricerca testuale che considera titolo, autore e genere
4. La gestione dell'archivio di generi, strumenti e musicisti; è possibile aggiungerne di nuovi, visualizzare quelli esistenti in dettaglio, modificarli e rimuoverli
5. La possibilità di creare CD nuovi o modificare quelli esistenti
6. La possibilità di rifornire la quantità di CD nel magazzino
7. La possibilità di salvare le modifiche effettuate, sia all'uscita dal programma, sia tramite uno specifico pulsante

Il tutto è fruibile grazie all'implementazione di un'interfaccia grafica.

Per la realizzazione si è seguito uno schema di programmazione ad oggetti (OOP) col linguaggio Java. Per l'interfaccia grafica si è scelto di utilizzare JavaFX, in quanto esso è moderno, aggiornato e il risultato è più gradevole di altre librerie più datate. Abbiamo scelto di non usare Swing in quanto Oracle ha annunciato che lo abbandonerà in favore di JavaFX. Tra gli strumenti utilizzati per la realizzazione vi sono l'IDE Eclipse e il tool SceneBuilder, che permette di creare graficamente layout per la GUI e di esportare un file FXML, da importare successivamente e utilizzare nel codice. Per il salvataggio dei dati si è scelto di implementare la serializzazione delle classi, che permette di salvarle su dei file ".ser", seppur con la consapevolezza che una realizzazione effettiva e completa del software necessiterebbe di un database. Infine per realizzare la documentazione sono stati utilizzati StarUML e Word.

**Pattern utilizzati**

MVC (Model - view - controller)

L'utilizzo di JavaFX guida lo sviluppatore all'utilizzo di questo pattern, che prevede una separazione di model (dati), view (viste dell'utente) e controller (che permette la comunicazione tra i due). Per un'ulteriore separazione sono stati racchiusi i vari file del prototipo in 3 package, uno per componente.

Il model è costituito da 4 classi: CD, Musicista, Magazzino e Archivio. CD raccoglie i dati in stringhe o numeri, mentre ha un riferimento alla classe musicista per l'autore, uno a java.timei.LocalDate per la data di vendita, e due Liste: una di stringhe per le canzoni e uno di coppie Musicista-stringa per i partecipanti col relativo strumento. Musicista ha a sua volta le informazioni salvate in stringhe o numeri, insieme ad un java.utils.Set di stringhe per gli strumenti suonati, nello specifico un HashSet. Queste due classi hanno come metodi i costruttori, i getter e i setter; implementano inoltre le interfacce Comparable (per ordinare i dati) e Serializable (per il salvataggio su file). La classe Archivio contiene tre liste, due di stringhe per generi e strumenti e una di musicisti, e un riferimento privato a sè stesso per l'implementazione del pattern singleton (vedi successivo). I metodi sono costruttori, adder e remover. La classe magazzino infine contiene una lista di coppie CD-Integer per raccogliere i cd con la relativa giacenza e un riferimento privato a sé stesso per l'implementazione del pattern singleton. I metodi sono i costruttori e i metodi per aggiungere cd, cancellarne uno o rifornirlo. Inolte entrambe queste classi hanno il getter per il patter singleton e un metodo per salvare i dati su file tramite serializzazione.

CD model

La view nel pattern MVC è costituita dall'insieme delle viste che l'utente ha sul sistema. In questo prototipo, grazie a JavaFX, i layout delle diverse view sono stati realizzati con SceneBuilder in file FXML. Questi file vengono poi importati dalla classe FXMLLoader, che li carica e genera una javafx.Scene che può essere inserita in uno Stage e visualizzata.

Il controller è un package costituito da varie classi che gestiscono le finestre create dalla View. Il programma comincia con Main, che crea LoginController per gestire l'autenticazione; se l'utente accede correttamente viene avviata la vista principale, quella sul magazzino, gestita da MagazzinoController. In questa classe vi sono tutti i listener per i pulsanti presenti sulla finestra, così come un riferimento al model del magazzino. La maggioranza di questi listener, implementati come metodi, crea delle nuove view gestite dagli specifici controller: GestisciGeneri, GestisciStrumenti, GestisciMusicisti, ModificaCD. La finestra di gestione dei musicisti ha inoltre un'altra schermata per la creazione/modifica di nuovi artisti. Ognuno di questi controller implementa l'interfaccia Initializable, che prevede l'esecuzione di uno specifico metodo alla creazione della finestra per il popolamento delle componenti (testi, elenchi, etc.). Infine vi sono due classi che estendono ListCell e controllano gli elementi delle ListView dei prodotti in magazzino e dei partecipanti nei singoli CD; queste si occupano di ricevere i dati e inserirli nei loro componenti.

CD Controller

Observer

Questo pattern è utilizzato per legare un oggetto che viene modificato a degli altri oggetti che "osservano" la modifica e reagiscono di conseguenza. Con JavaFX l'applicazione di questo pattern è, in alcuni casi, semplificata; nella dichiarazione dei componenti nel file FXML è possibile impostare direttamente il metodo che deve essere eseguito quando una certa azione avviene, per esempio nella creazione di un bottone è possibile dichiarare sul file FXML il nome del metodo che viene invocato al click del mouse, che deve essere implementato nella classe controller. Un altro caso in cui è stato applicato questo pattern sono le ListView; queste nel metodo initialize() vengono collegate ad un ObservableArrayList, e ogni qual volta vi è una modifica nei suoi elementi la view si aggiorna automaticamente.

Singleton

Questo pattern si utilizza quando si vuole creare una sola istanza di una certa classe evitando duplicati; ciò si realizza con un costruttore privato e un metodo statico getInstance(), che crea l'oggetto alla prima chiamata e lo ritorna. In questo prototipo è stato utilizzato due volte, nel magazzino e nell'archivio, in quanto questi oggetti devono avere una sola istanza. In particolare è stato implementato il singleton pigro, anche se quello non pigro sarebbe stato sufficiente poiché queste classi vengono create in ogni caso ad ogni avvio per leggere i dati.

Factory

Questo pattern prevede l'utilizzo di una classe per creare e ottenere istanze di altre classi senza invocarle direttamente. Nel nostro caso è stato utilizzato con le ObservableArrayList, in quanto non prevedono un costruttore, ma si inizializzano chiamando il metodo apposito dalla classe FXCollection.

**Funzionamento del prototipo**

Il programma al lancio chiama la classe LoginController e mostra una schermata di log-in all'utente, che deve inserire le sue credenziali per poter accedere al sistema. Ciò è stato fatto per evitare che utenti non autorizzati (quindi senza una combinazione username-password valida) possano entrare illecitamente nel sistema. La classe Login (controller della finestra) gestisce i dati inseriti e, al click o invio dell'utente, controlla i dati con la funzione check; in caso di campi mancanti o combinazione errata notifica l'utente con un messaggio.  
In questa versione del prototipo il metodo check() controlla solo che i campi username e password siano uguali, ma in una versione più completa potrebbe accedere ad un database di gestori e controllare questi dati.

//SD login

//AD login

Se il login ha avuto esito positivo la classe Main genera la finestra del magazzino, caricando il layout e assegnandoli come controller MagazzinoController. Quest'ultimo chiede al model (in particolare al magazzino) di caricare i dati e, con questi ultimi, popola le liste che ha creato. A questo punto il controller finisce il lavoro e diventa osservatore di ciò che l'utente fa sulla finestra; la scena prodotta viene ritornata a Main per essere racchiusa in uno Stage e mostrata.

//SD avvio

//AD avvio

Una volta che la finestra è stata creata l'addetto può fare molte cose. Una di queste è la creazione o modifica dei CD, che l'utente avvia col pulsante di creazione o selezionando un CD e successivamente il pulsante di modifica. Il controller del Magazzino reagisce in maniera similare a entrambi gli eventi, creando una finestra di creazione cd e associando come controller la classe ModificaCDController. Questa classe nel creare la scena precompila i campi coi dati attuali nel caso di modifica, li lascia vuoti nel caso di creazione. Una volta che l'utente conferma i dati inseriti il controller controlla tutti i campi, stampando un warning all'utente nel caso di errori. Una volta passati i controlli verifica anche l'esistenza di doppioni in magazzino, notificando eventuali errori. Se tutta la procedura è corretta il model viene aggiornato e la finestra chiusa. Grazie al pattern observer la ListView del magazzino rileva la modifica e aggiorna automaticamente l'elenco.

//SD aggiungi CD

//AD Aggiungi CD

Un'altra attività è la gestione dell'archivio, che si divide tra la gestione dei musicisti, dei generi e degli strumenti. I diagrammi tratteranno solo la gestione dei musicisti, poiché le altre due sono analoghe e più semplici; unico particolare degno di nota delle due attività è l'utilizzo dello stesso layout associato a diversi controller.  
Una volta che l'utente seleziona il pulsante di gestione musicisti il controller ricava dal model l'elenco e prepara la finestra che poi mostra all'utente. A questo punto l'utente ha diverse alternative. Può eliminare un musicista selezionandolo dall'elenco e cliccando l'apposito bottone; il controller provvederà ad informare in model del cambiamento e ad aggiornare la finestra. Oppure può aggiungerne uno o modificarlo; il controller aprirà l'apposita finestra precompilando i campi coi dati attuali in caso di modifica, per poi notificare il model una volta ottenuti i dati nuovi. Infine è possibile visualizzare i dettagli di un musicista selezionandolo; in questo caso il controller, una volta ottenuti i dati dal model, aggiorna la finestra con le informazioni ottenute. Nel diagramma sono stati omessi per semplicità i controlli sui dati, sia di correttezza che di eventuale duplicità. Non è stato inoltre specificato l'inserimento/modifica del nuovo musicista in quanto equivalente a quello del CD.

//SD gestione musicisti

//AD gestione musicisti //TODO

Un'altra attività implementata è la ricerca di un determinato prodotto, che avviene tramite una barra di ricerca e un pulsante nella finestra principale del programma. Una volta inserito il testo e cliccato il pulsante il controller del magazzino gestisce l'evento stampando un warning nel caso in cui l'utente abbia inserito meno di 3 caratteri. se invece il testo è valido il controller ricerca nell'elenco tutti i prodotti che hanno quel testo nel titolo, nel genere o nel musicista; una volta ottenuti i dati il controller aggiorna la finestra mostrando solo quelli nell'elenco, rimuovendo la barra di ricerca e sostituendo il pulsante di ricerca con uno di "annulla ricerca", che reimposta l'elenco coi dati iniziali.

//SD ricerca

//AD ricerca //TODO

Infine vi sono altre azioni di cui non è stato realizzato un diagramma perché sarebbe risultato una ripetizione di quanto illustrato finora. Queste azioni sono: la rimozione di un CD, il rifornimento di giacenza e l'ordinamento dell'elenco secondo i 4 parametri (titolo, autore, prezzo e giacenza). Queste azioni prevedono poca interazione con l'utente e opzionalmente l'aggiornamento di un dato nel model.

**Testing**

La progettazione di ogni elemento del software è stata fortemente legata ad una fase di testing. Ogni volta che una nuova funzionalità o schermata grafica veniva implementata vi era subito un test da parte degli sviluppatori per assicurarsi che funzionasse in modo appropriato. Per esempio prima dell'implementazione delle finestre di inserimento informazioni erano stati creati dati di esempio per testare la vista del magazzino, per poi essere rimossi una volta implementata la serializzazione.

Per ogni componente sono stati testati anche i casi limite, inserendo dati errati, per assicurarsi che il programma rispondesse adeguatamente fermando le operazioni di modifica e notificando l'utente dell'errore.

Quando poi il software è diventato operativo sono stati eseguiti dei test sull'intero sistema, in ogni sua componente; infine dopo un refactoring del codice per uniformarlo e limare alcune imprecisioni questi test sono stati ripetuti.

Gli autori del test sono stati in primis gli sviluppatori, per poi passare ad altri utenti estranei al progetto ma comunque inseriti nel contesto universitario e con un background informatico. Infine hanno eseguito dei test anche altri utenti completamente estranei al software e mediamente inesperti in ambito informatico; questi ultimi sono stati di fondamentale importanza per avere un feedback sull'usabilità del software e sulla gradevolezza dell'interfaccia grafica, e ci hanno permesso di limare alcune imperfezioni al fine di rendere il programma più user-friendly.

**Scelte progettuali**

Nella realizzazione di questo prototipo sono state operate diverse scelte per trovare un compromesso tra usabilità e realizzabilità.

I dati sono stati salvati su file locali, sebbene una realistica implementazione di questo software necessiterebbe di una base di dati accessibile da remoto.

Le copertine dei CD sono anch'esse salvate in locale, quindi cambiando postazione non possono essere caricate. In una realizzazione futura queste dovrebbero essere caricate in un server FTP o simile da cui poter essere scaricate indipendentemente dalla macchina su cui viene eseguito il software.

JavaFX offre la possibilità di personalizzare gli stili dei componenti tramite file CSS. Nonostante l'enorme potenziale di questa funzione si è scelto di mantenere lo stile di default in quanto lo scopo di questo prototipo è solo quello di offrire all'utente un'anteprima del prodotto finale. Inoltre lo stile grafico è un aspetto estremamente personale e dovrebbe essere deciso direttamente dal committente.

La gestione degli utenti autorizzati nella fase di log-in è stata implementata semplicemente con un controllo di uguaglianza tra username e password. In una realizzazione futura del progetto questa funzionalità sarebbe da concordare col committente, con la possibilità di un sottosistema per la gestione dei profili autorizzati, oppure con un elenco gestito da software esterni.

Si è scelto di utilizzare un approccio più rigido nella gestione di strumenti e generi. Sebbene siano trattati come semplici stringhe per poterne aggiungere o rimuovere è necessario utilizzare l'apposito pannello, e non possibile farlo in fase di creazione di cd o musicisti. Questa procedura è più lenta per l'utente, specie durante i primi utilizzi, ma evita possibili ripetizioni.