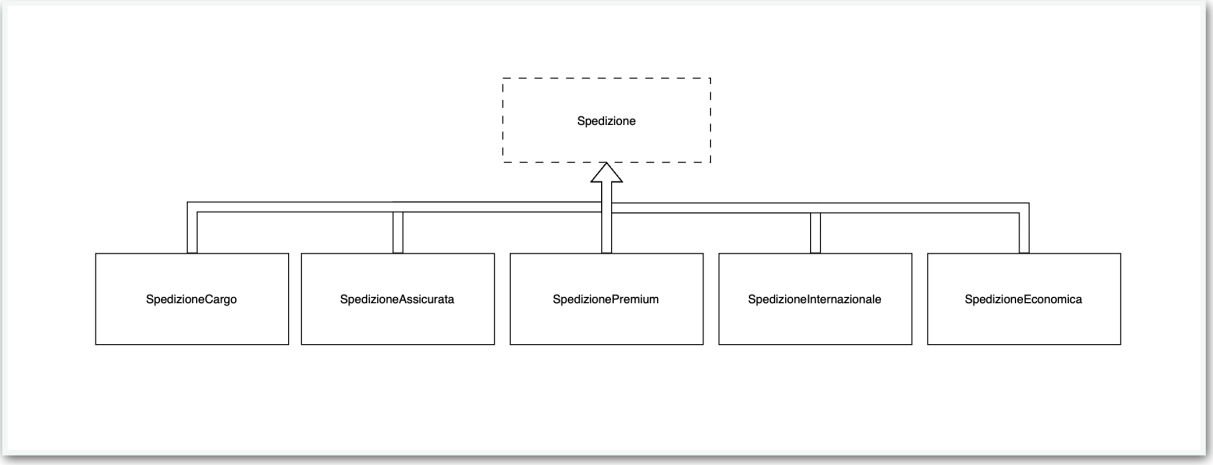
SPEEDIZIONI

Dugo Alberto 2042382
Bencivelli Lorenzo 2006428

1. Introduzione

Speedizioni è un applicazione desktop, progettata per semplificare il processo di spedizione di pacchi. L'obiettivo principale di Speedizioni è offrire agli utenti un'esperienza intuitiva, veloce e accessibile per creare e gestire spedizioni in modo efficiente.
Nell'attuale scenario in cui il numero di spedizioni aumenta esponenzialmente ogni giorno, la velocità è diventata un elemento fondamentale. Speedizioni si distingue per la sua velocità di utilizzo, consentendo agli utenti di creare una nuova spedizione in pochi semplici passaggi. L'applicazione è stata progettata per ridurre al minimo il tempo necessario per compilare i dettagli della spedizione, consentendo agli utenti di inviare i loro pacchi in modo rapido ed efficiente.
Un aspetto chiave di Speedizioni è la vasta gamma di tipologie di spedizioni disponibili. L'applicazione è stata pensata per essere polifunzionale e adatta a diversi tipi di utenti. Sia che si tratti di un anziano che desidera spedire delle cartoline, di un giovane che vuole inviare un prodotto venduto online o di un'azienda che necessita di spedire i propri prodotti, Speedizioni offre un servizio versatile per soddisfare le diverse esigenze dei suoi utenti.
La ragione principale alla base della progettazione di Speedizioni come servizio accessibile e intuitivo è il suo obiettivo di raggiungere un target ampio e variegato. L'applicazione è stata sviluppata tenendo conto delle esigenze di un pubblico eterogeneo, composto da utenti di diverse fasce d'età e con diversi livelli di competenza tecnica. La semplicità d'uso e l'accessibilità sono state considerate priorità per garantire che tutti gli utenti possano beneficiare delle funzionalità offerte da Speedizioni senza difficoltà.
Inoltre, Speedizioni si impegna a fornire un servizio completo e affidabile. L'applicazione permette di gestire non solo le informazioni di base relative alla spedizione, ma anche dettagli specifici come la modalità di consegna, l'assicurazione, le tariffe e altro ancora. Ciò consente agli utenti di avere il pieno controllo sulle loro spedizioni e di adattarle alle proprie esigenze specifiche.
Speedizioni si propone come una soluzione all'avanguardia per semplificare il processo di spedizione di pacchi. La sua interfaccia intuitiva, la velocità di utilizzo e la vasta gamma di opzioni disponibili rendono l'applicazione uno strumento indispensabile per gli individui, i privati e le aziende che desiderano inviare pacchi in modo rapido, affidabile e conveniente.

2. Descrizione modello implementato



2.1 Gerarchia

La gerarchia delle spedizioni in Speedizioni è organizzata in base a una classe padre, chiamata "Spedizioni", da cui ereditano tutte le diverse classi di spedizioni: SpedizioneCargo, SpedizioneAssicurata, SpedizioneEconomica, SpedizioneInternazionale e SpedizionePremium.
La classe SpedizioneCargo rappresenta una spedizione in cui il contenuto può essere trasportato tramite

due modalità: Aereo o Barca. Per implementare questa funzionalità, è stata utilizzata un'enumerazione che consente di selezionare la modalità di trasporto desiderata.
La classe SpedizioneAssicurata è stata progettata per offrire un vantaggio importante: la possibilità di assicurare il pacco durante la spedizione. Questo tipo di spedizione è particolarmente adatto per inviare oggetti di valore, fornendo un ulteriore livello di protezione e tranquillità al mittente.
La classe SpedizioneEconomica rappresenta una soluzione semplice e molto utilizzata. Offre un costo ridotto per spedire pacchi di valore più basso. Tuttavia, è importante notare che esistono restrizioni per quanto riguarda il volume e il peso massimo dei pacchi che possono essere inviati attraverso questa modalità.
La classe SpedizioneInternazionale è specificamente progettata per spedizioni destinate all'estero. A causa delle diverse necessità e regolamentazioni internazionali, questa tipologia di spedizione ha un costo maggiore rispetto alle altre, ma offre la possibilità di raggiungere destinazioni al di fuori del proprio paese di origine.
La classe SpedizionePremium è stata sviluppata per soddisfare le esigenze degli utenti che non sono disponibili per l'intera giornata. Grazie a questa opzione, è possibile programmare l'orario di arrivo della spedizione in modo da garantire che il destinatario sia presente al momento della consegna. Questo servizio offre flessibilità e comodità a coloro che hanno limitazioni di tempo.
La gerarchia delle spedizioni in Speedizioni consente di scegliere la tipologia di spedizione più adatta alle esigenze del mittente, offrendo una vasta gamma di opzioni che spaziano dal trasporto di merci via aereo o barca, all'assicurazione del pacco, alle spedizioni internazionali e alla possibilità di programmare l'orario di arrivo. Questa struttura gerarchica offre flessibilità e personalizzazione, consentendo agli utenti di selezionare la soluzione migliore per le proprie esigenze di spedizione.
Speedizioni offre non solo la creazione di diversi tipi di spedizioni, ma anche la possibilità di aggiungere, modificare ed eliminare componenti ausiliari come assicurazioni e filiali. Questo consente una gestione completa delle spedizioni. Pur non facendo parte della gerarchia principale, questi elementi sono ampiamente utilizzati e sono stati implementati per consentire l'utilizzo di Speedizioni come struttura di base da parte di enti esterni. Ciò offre un livello di flessibilità e personalizzazione superiore, adattando il sistema alle diverse esigenze degli utenti e consentendo una gestione ancora più completa delle spedizioni.

2.2 Contenitore

All'interno di Speedizioni, è stato scelto di utilizzare l'ArrayList come contenitore per il suo essere templetizzato e generale, il che lo rende estremamente flessibile e adatto a diverse situazioni all’interno di speedizioni stessa. ArrayList comprende quindi l’aggiunta in una determinata posizione o in coda, l’eliminazione di un determinato elemento o di tutti, la ricerca attraverso l’id e la lettura della lista attraverso il metodo toString.

2.3 Persistenza dei dati

Nell'implementazione di Speedizioni è stata inclusa una classe dedicata al collegamento dell’interfaccia con il FileSystem del sistema. Questa classe è stata progettata per rendere l'utilizzo e il salvataggio dei file il più trasparente possibile, offrendo una gestione automatica e invisibile all'utente.
Attraverso questa classe, l'applicazione è in grado di gestire il salvataggio dei dati nel FileSystem in modo efficiente e affidabile. L'utente non deve preoccuparsi di eseguire manualmente operazioni di salvataggio, poiché il sistema si occupa automaticamente di mantenere i dati aggiornati e sincronizzati.
L'utente può concentrarsi sulla gestione delle spedizioni senza dover preoccuparsi di complessi passaggi di salvataggio o recupero dei dati.
Inoltre, l'interfacciamento con il FileSystem garantisce la trasparenza del processo di salvataggio dei file. Gli utenti possono accedere ai loro dati in modo familiare, come se fossero semplici file sul loro sistema.

3. Polimorfismo

3.1 ArrayList

L'ArrayList utilizzato in Speedizioni sfrutta il concetto di polimorfismo, in quanto è templetizzato e quindi può contenere istanze di tutte le classi presenti nella gerarchia di spedizioni. Questo approccio consente di trattare gli oggetti delle diverse classi come oggetti dello stesso tipo generico, consentendo operazioni comuni e una gestione uniforme dei dati.

3.2 Spedizione

La classe Spedizione in Speedizioni implementa diversi metodi virtuali puri, come ad esempio "getCosto()" e "getTypeName()".
Il metodo virtuale puro "getCosto()" restituisce il costo specifico della spedizione e la sua implementazione è fornita nelle classi derivate. Questo permette di ottenere il costo corretto per ciascuna tipologia di spedizione all'interno della gerarchia.
Il metodo virtuale puro "getTypeName()" restituisce una stringa che rappresenta il tipo di spedizione. Anche questo metodo è implementato in modo polimorfico nelle classi derivate, consentendo di ottenere il tipo corretto per ogni istanza di spedizione.
Inoltre, la classe Spedizione implementa metodi virtuali come "modifica()" per consentire la modifica delle varie spedizioni. Questo metodo virtuale può essere implementato in modo specifico nelle classi derivate per effettuare le modifiche necessarie.
I metodi virtuali "Accept()" e "toSaveFormat()" sono implementati per supportare il pattern Visitor. Il metodo "Accept()" consente di accettare un oggetto Visitor e di eseguire operazioni specifiche su ciascuna istanza di spedizione all'interno della gerarchia. Il metodo "toSaveFormat()" permette, mediante lo sfruttamento del polimorfismo, che ogni elemento della gerarchia ritorni una stringa composta in maniera rigorosa, da tutte le informazioni dell'oggetto, con lo scopo di salvarle secondo una corretta formattazione all'interno del file di salvataggio, garantendo così una facile gestione dei dati salvati e del loro reperimento.

4. Interfaccia grafica (GUI)

4.1 Single Page Application

Speedizioni è progettato come una Single Page Application, in cui tutte le schermate dell'applicazione sono visualizzate all'interno di una singola finestra. Questa struttura consente una navigazione fluida tra le varie funzionalità dell'app. Tuttavia, ci sono alcune eccezioni a questa modalità, come i pop-up di avviso o la creazione di assicurazioni e filiali. Queste operazioni richiedono un numero limitato di campi di inserimento, rendendo inefficace dedicare loro un'intera schermata vuota. Pertanto, sono state progettate soluzioni più efficienti per consentire una gestione agevole e ottimizzata di queste componenti ausiliarie.

4.2 Visitors

L'utilizzo dei Visitors in Speedizioni è stato implementato con l'obiettivo di sfruttare appieno il concetto di polimorfismo. I Visitors consentono di interfacciare il codice della GUI e del core del progetto, garantendo la loro funzionalità e indipendenza reciproca, mantenendo la parte logica separata dalla parte grafica, favorendo una maggiore leggibilità e manutenibilità del codice.
L'impiego di diversi Visitors per compiti specifici, come ad esempio il VisitorById per la ricerca degli elementi della gerarchia attraverso un identificatore numerico, offre un approccio generale nell'utilizzo degli oggetti all'interno del sistema. Questo consente di gestire in modo efficiente diverse operazioni senza compromettere la generalità dell'implementazione.
Inoltre, sono stati utilizzati Visitors per la generazione dinamica di Dialog basati sull'oggetto chiamante, come ad esempio Assicurazione o Filiale. Questa soluzione offre una flessibilità e un'adattabilità maggiore nel processo di creazione di dialoghi specifici per ogni tipo di componente ausiliaria.

4.3 QSS

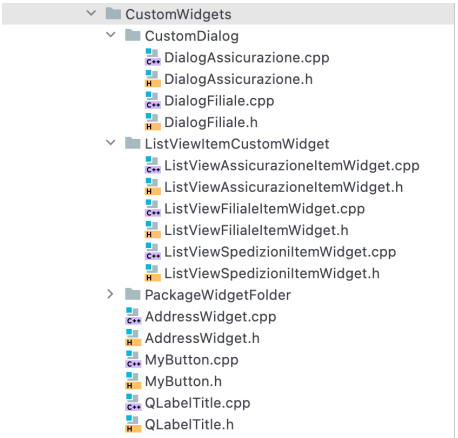
Speedizioni fa un ampio utilizzo di QSS (Qt Style Sheets) e icone al fine di migliorare l'esperienza utente e rendere l'interfaccia visivamente piacevole.
Inoltre, l'utilizzo di icone adeguate contribuisce a migliorare l'usabilità e la comprensione delle funzionalità dell'applicazione. Le icone vengono utilizzate per identificare in modo visivo le diverse azioni, migliorando la navigazione e la comprensione delle operazioni disponibili.
L'obiettivo principale di questo approccio è quello di rendere l'esperienza utente quanto più godibile possibile attraverso l'utilizzo di stili personalizzati e icone.

4.4 Filtri

Speedizioni implementa un avanzato sistema di filtraggio dinamico che consente agli utenti di effettuare ricerche in tempo reale e visualizzare solo gli elementi desiderati nella schermata principale. Questo sistema utilizza filtri e una barra di ricerca per aggiornare istantaneamente la lista degli elementi in base ai criteri specificati.
Grazie a questo sistema, gli utenti possono facilmente definire i parametri di ricerca desiderati, come ad esempio il tipo di spedizione, oppure lo stato di essa. I risultati vengono filtrati in tempo reale, consentendo agli utenti di visualizzare solo gli elementi pertinenti alla loro ricerca.
L'obiettivo principale di questo sistema è quello di migliorare l'efficienza e la comodità nella gestione delle spedizioni, consentendo agli utenti di trovare rapidamente le informazioni desiderate senza dover navigare attraverso una lista completa di elementi. Il filtraggio dinamico ci permette di offrire quindi un'esperienza utente più fluida e un accesso immediato alle informazioni pertinenti.
È stato inoltre implementata la possibilità di ricerca dinamica nella pagina principale, essa consente di ricercare in tutte le spedizioni, attraverso l'ID numerico della spedizione, è stato inoltre implementata la possibilità di ricerca attraverso sottostringhe. Es: ID 59, è possibile trovare quella determinata spedizione cercando solamente 5 oppure 9.

4.5 Custom widget

Speedizioni include la creazione di widget personalizzati, progettati e sviluppati in base alle specifiche e alle richieste specifiche sia dal punto di vista funzionale che grafico. Questi widget personalizzati sono disponibili nella cartella "Widgets/CustomWidgets" all'interno del codice dell'applicazione, qui di fianco possiamo ritrovarli tutti.
I widget personalizzati offrono funzionalità specifiche e sono progettati per soddisfare esigenze particolari nell'ambito della gestione delle spedizioni.
Ogni widget personalizzato è stato progettato tenendo conto dell'aspetto grafico e funzionale desiderato. Questo permette di integrarli in modo coerente all'interno dell'interfaccia utente, offrendo una navigazione fluida e una chiara visualizzazione delle informazioni.



5. Tempistiche

Attività	Ore previste	Ore svolte
Progettazione	10	11
Codice model	13	16
Codice GUI	15	23
Test e debug	7	10
Relazione	5	5
Totale	50	65

Durante lo sviluppo dell'applicazione Speedizioni, ho riscontrato che le tempistiche previste non sono state rispettate come originariamente pianificate. Dopo alcune valutazioni, ritengo che ciò sia principalmente dovuto alla sottostima delle ore effettive necessarie per la stesura del codice e per l'apprendimento effettivo dei signal e slot in Qt. È stato quindi fondamentale per evitare i medesimi problemi nei progetti futuri comprendere che la fase di progettazione è fondamentale per evitare di sottovalutare i vari compiti.

6. Divisione dei compiti

Nel complesso del progetto ho svolto le seguenti attività:

- Model
    - hierarchy
        - Spedizione
        - SpedizionePremium
        - SpedizioneAssicurata
- Librerie
    - FileManager
- App
    - widgets
        - Custom widget
            - AddressWidget
            - CustomDialog/*
            - ListViewItemCustomWidget/*
            - PackageWidgetFolder/*
            - QLabelTitle
    - Visitor/*
    - Pages
        - Home
        - HierachyPages
            - HierachyPageInterface
            - SpedizioneAssicurataPage
            - SpedizionePremiumPage
- CmakeLists

Ho fornito anche un file CMake con cui compilare, per via del nostro ambiente di sviluppo (CLion), il quale non permette l'uso integrato di QMake. Abbiamo poi, in un secondo momento, creato un file QMake per permettere la compilazione anche tramite esso.

7. Modifiche riconsegna

7.1 [-] Non gestisce I/O su file con formato strutturato (Vincolo 8)

Per la persistenza dei dati abbiamo aggiornato la metodologia di salvataggio dei dati, Speedizioni ora utilizza un formato strutturato: JSON. È possibile trovare i vari file nella cartella "Librerie".
Abbiamo inoltre implementato la capacità di Speedizioni di avviarsi e creare in automatico i file di salvataggio se questi sono stati eliminati.

7.2 [-] Il metodo virtuale puro "getTypeName"

Ora il metodo getTypeName è stato sostituito da un visitor "VisitorForClassName()".
la classe ha diversi metodi visit che prendono come argomento un puntatore polimorfo a diverse classi derivate, e ogni metodo visit imposta il valore della variabile "className" in base al tipo dell'oggetto visitato. Ad esempio, se viene visitato un oggetto di tipo "SpedizioneCargo" con la tipologia di trasporto "AEREO", il metodo visit corrispondente imposta "className" come "SPEDIZIONECARGO". Il metodo "getClassName" della classe Visitor restituisce il valore className, che conterrà il nome della classe dell'ultimo oggetto visitato. Quindi, l'utilizzo di VisitorForClassName consente di ottenere il nome della classe di un oggetto polimorfo in modo dinamico utilizzando il visitor pattern.

7.3 [-] Il metodo "modifica" in Spedizione dovrebbe chiamarsi "copy"

Abbiamo inoltre rinominato il metodo "modifica" in "copy" in modo tale da non ricadere in un utilizzo di un metodo di modifica con polimorfismo banale.