Gruppo OOBD2324_18

Davide Gargiulo	david.gargiulo@studenti.unina.it	N86004689
Vincenzo Di Carluccio	vi.dicarluccio@studenti.unina.it	N86004800

UNINA DELIVERY

UNIVERSITÀ
DEGLI STUDI
DI NAPOLI
FEDERICO II

Contents

1 Introduzione

Questa documentazione fornisce una guida esaustiva sull'implementazione e sull'uso del sistema di gestione della logistica delle spedizioni di merci di **UninaDelivery**. L'applicativo, progettato con un'interfaccia grafica intuitiva, è destinato agli operatori di un'azienda di delivery, consentendo loro di gestire le operazioni logistiche in modo efficiente, preciso e organizzato.

Il sistema offre le seguenti funzionalità principali:

- Visualizzare in tempo reale la lista degli ordini in attesa di evasione;
- Pianificare e creare spedizioni per tali ordini, ottimizzando le risorse e tenendo conto della disponibilità delle merci nei depositi;
- Generare e consultare report mensili sugli ordini evasi per migliorare supervisione, monitoraggio e pianificazione strategica.

1.1 Assunzioni sul dominio

Il sistema **UninaDelivery** è stato progettato per un'azienda che gestisce spedizioni di:

- **Merce propria**, prodotta internamente;
- **Merce di terzi**, gestita in custodia per conto di clienti esterni.

L'azienda opera a livello **nazionale**, coprendo l'intero territorio con una rete di depositi distribuiti.

Gli operatori possono gestire sia spedizioni dirette verso i clienti, sia spostamenti intermedi tra depositi per ottimizzare la disponibilità di merci nei punti strategici. Inoltre, si assume che:

- Le spedizioni dirette verso un cliente avvengano preferibilmente da depositi nella stessa città del destinatario.
- In caso di indisponibilità di prodotti nel deposito locale, il sistema supporti spedizioni intermedie da altri depositi o spostamenti interni, garantendo la completa evasione dell'ordine.
- Ogni ordine sia limitato a un solo prodotto, con quantità variabile. Gli ordini complessi (contenenti più prodotti) vengono suddivisi automaticamente in ordini separati, a condizione che siano destinati alla stessa area geografica (identificata dal CAP) o al medesimo deposito finale.

1.2 Obiettivo della documentazione

Questa documentazione mira a fornire una guida dettagliata e completa sulle funzionalità del sistema **UninaDelivery**, coprendo tutti gli aspetti fondamentali:

- Configurazione iniziale e requisiti di sistema;
- Utilizzo quotidiano per la gestione operativa delle spedizioni;
- Procedure di manutenzione e aggiornamento del software.

Il documento è pensato per diversi tipi di utenti:

- **Operatori logistici**, che utilizzeranno il sistema per la gestione pratica delle spedizioni;
- **Amministratori di sistema**, responsabili della configurazione e manutenzione tecnica;
- **Analisti e supervisori**, che sfrutteranno i report per monitorare e ottimizzare le attività logistiche.

Sia che siate nuovi al sistema o utenti esperti in cerca di informazioni dettagliate, questa guida è stata progettata per essere un riferimento intuitivo e affidabile.

2 Progettazione del Sistema

2.1 Strumenti e Metodologie Utilizzate

Per progettare l'applicativo grafico per il sistema di gestione delle spedizioni, si è scelto di seguire il **pattern architetturale EBC** (**Entity-Boundary-Control**). Questo approccio è stato preferito in quanto promuove una chiara **separazione delle responsabilità** tra le componenti del sistema, facilitando la **manutenzione**, l'**aggiornamento** e la **scalabilità** dell'applicazione.

Per la realizzazione dell'interfaccia grafica, è stato adottato il **framework JavaFX**, supportato dalla libreria **MaterialFX** per migliorare l'estetica dei componenti grafici e da **SceneBuilder** per la creazione e la gestione dei layout. Questa scelta è motivata dalla moderna implementazione del framework e dal supporto diretto di Oracle, che lo rende una soluzione stabile e ben documentata.

La connessione con il database, implementato in **PostgreSQL**, è stata gestita attraverso **JDBC**, una soluzione semplice e diretta per la comunicazione tra il sistema e il database. Per la gestione delle dipendenze e delle routine di compilazione del progetto, si è utilizzato **Maven**, uno strumento standard per l'ecosistema Java.

2.2 Architettura del Sistema e Class Diagram

Il sistema segue il pattern architetturale **EBC**, che suddivide il sistema in tre componenti principali: **Entity**, **Boundary** e **Control**.

2.3 Entity

La componente **Entity** è responsabile della gestione e della rappresentazione dei dati del sistema.

Le classi che implementano il **design pattern DAO** (**Data Access Object**) si occupano dell'interazione con il database e delle operazioni di **CRUD** (**Create, Read, Update, Delete**).

Le classi che implementano il **design pattern DTO** (**Data Transfer Object**) rappresentano il modello di dominio del sistema e contengono metodi getter e setter.

Per migliorare la flessibilità, il package DAO utilizza un'interfaccia generica **BaseDAO** che definisce le operazioni di base. Ogni classe specifica estende questa interfaccia per implementare operazioni personalizzate.

2.4 Boundary

La componente **Boundary** gestisce l'interfaccia utente, implementando la logica di visualizzazione e l'interazione con gli utenti.

Le classi **Boundary** agiscono come **listener** per gli eventi generati dall'interfaccia grafica.

Per garantire coerenza e ridurre la duplicazione di codice, si utilizzano classi astratte che generalizzano le operazioni comuni.

2.5 Control

La componente **Control** si occupa della **logica di business**, della validazione dei dati e della comunicazione tra Entity e Boundary.

Ogni classe Control corrisponde a un caso d'uso specifico e implementa le operazioni richieste.

Tutte le classi Control seguono il **design pattern Singleton** per garantire una gestione centralizzata delle operazioni.

2.6 Utils

Il package **Utils** contiene classi di utilità per funzioni comuni, come la gestione delle stringhe, la cifratura dei dati e la gestione delle eccezioni personalizzate. Tutte le classi in questo package sono **final** e contengono esclusivamente metodi statici.

2.7 Class Diagram Completo