

A.A. 2020/2021

Relazione Progetto Programmazione 3  
Docenti: Angelo Ciaramella e Raffaele Montella  
  
Alunni: Alessio Cinque, Luca Francesco Russo  
Matricola: 0124001950, 0124001976  
Tipo di progetto: Progetto gestionale

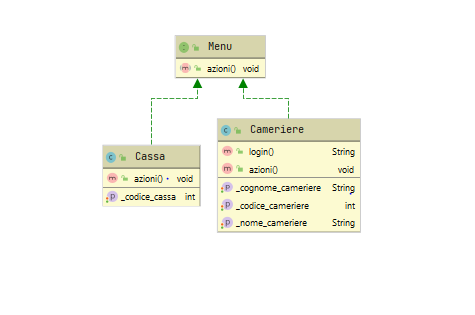
**DESCRIZIONE E REQUISITI DEL PROGETTO**

Si vuole sviluppare un programma per la gestione di una pizzeria   
L’applicazione deve prevedere l’accesso in modalità cameriere e cassiere**.**

Se la scelta è cameriere, ci sarà poi un altro login per scegliere il cameriere, una volta effettuato l’accesso, il cameriere dovrà scegliere che operazione effettuare(CREA ORDINE, CANCELLA ORDINE, MODIFICA ORDINE), se si sceglie crea ordine verrà chiesto di inserire un prodotto fin quando non si inserirà la parola ‘fine’. Se si sceglie modifica ordine viene chiesto se aggiungere o rimuovere un prodotto verrà chiesto poi il nome del prodotto da aggiungere/rimuovere e il codice dell’ordine da cui aggiungerlo/rimuoverlo. Se si sceglie cancella ordine verrà chiesto il codice dell’ordine da cancellare. Se si sceglie il cassiere, uscirà un messaggio che avvisa i tavoli occupati per verificare che sia corretto il numero del tavolo fornito dal cliente, verrà chiesto poi il numero del tavolo da liberare e poi il codice dell’ordine, uscirà poi così il totale di quell’ordine.  
  
**Bisogna permettere al cameriere** di:  
- Creare un ordine  
- Modificare un ordine  
- Cancellare un ordine  
  
Il cassiere potrà invece:  
-Visualizza il totale dell’ordine

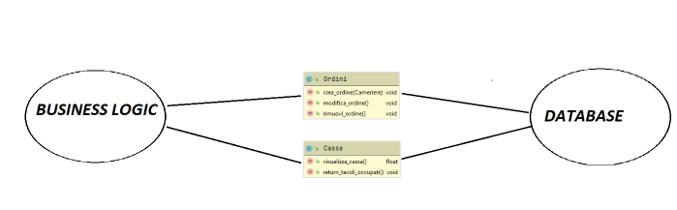
-Libera il tavolo occupato dall’ordine appena visualizzato  
  
  
Uno dei due pattern utilizzati è il **FACADE PATTERN** in quanto, attraverso un’interfaccia semplice, accediamo a sottosistemi che espongono interfacce complesse e diverse tra loro.  
L’utilizzo di questo pattern permette di nascondere la complessità delle operazioni dato che l’utente attraverso il Main sceglie quale utente essere. Per implementare il pattern uso:  
Menu: l’interfaccia contenente il metodo azioni() che verrà implementato dalle altre classi a seconda delle operazioni da fare.  
Cameriere: la classe che implementerà l’interfaccia menu. La classe implementerà le azioni in cui dopo aver scelto dal menù a tendina l’azione che si vuole svolgere, verrà richiamato il metodo appropriato dalla repository Ordini.

.Cassa: implementerà l’interfaccia menù. Anche questa classe andrà a richiamare il metodo per visualizzare il totale dell’ordine dal repository Casse .

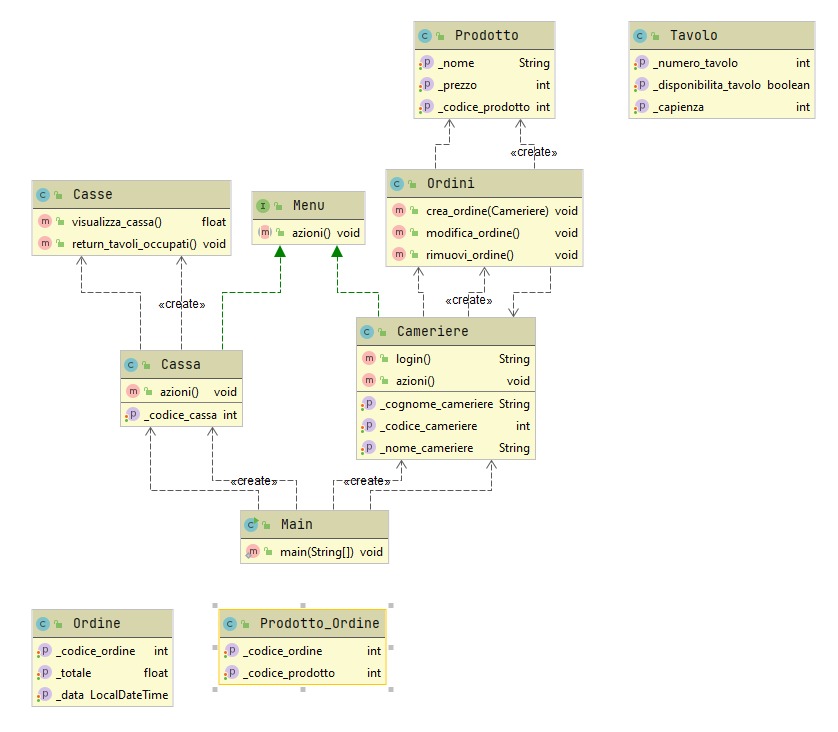
**Scopo**: Fornisce un’interfaccia unificata ad un insieme di interfacce in un sottosistema. Definisce un’interfaccia di alto livello che permette la facile gestione del sottosistema, dato che spesso i sottosistemi diventano complessi nel tempo aggiungendo nuove funzionalità.  
Motivazione: permettere attraverso un’interfaccia più semplice, l’accesso a sottosistemi che espongono interfacce più complesse e molto diverse tra loro.  
Applicabilità: quando vogliamo fornire un’interfaccia semplice ad un sottosistema complesso.   
  


L’altro pattern è il **REPOSITORY PATTERN**

Le classi Ordini e Casse sono classi repository cioè classi che andremo ad usare nel pattern repository. Esso permette di disaccoppiare la logica di business dall’accesso ai dati nell’applicazione, quindi sarà all’interno di queste due classi che andremo a fare le operazioni sul database di creazione, modifica e cancellazione dell’ordine, i vari create e le varie query e anche le operazioni e query della cassa. Quindi la persistenza dei dati e cioè il database sarà associato soltanto a queste due classi

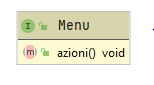


**UML DELLE CLASSI**



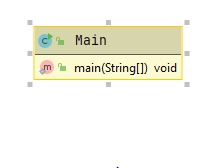
**Menù**

Tramite l’interfaccia Menu andiamo ad accedere ai 2 sottosistemi e una volta entrati l’utente deciderà le azioni da svolgere



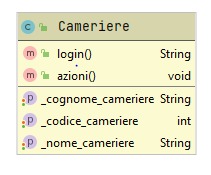
**Main**

Nel main andremo ad usare le librerie JOptionPane per chiedere al cliente se deve loggarsi come cameriere o cassiere



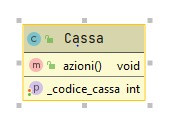
**Cameriere**

La classe cameriere effettuerà il login del cameriere e implementerà l’interfaccia Menù, andando ad implementare il metodo azioni() andando a scegliere l’azione da fare, cioè creare modificare o cancellare un ordine, in base alla scelta verrà richiamato il metodo all’interno della repository ordini



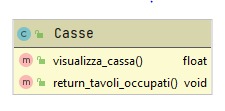
**Cassa**

La classe Cassa implementerà l’interfaccia Menù, implementando il metodo azioni() in cui andrà a richiamare dalla repository Casse il metodo per andare a visualizzare tutti i tavoli occupati nella sala e il metodo per andare a visualizzare il totale dell’ordine di un certo tavolo



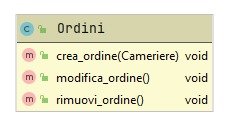
**Casse**

E’ una classe utilizzata nel pattern repository ed è la classe in cui andremo a fare le operazioni su database riguardanti le operazioni del cassiere, cioè le varie query.



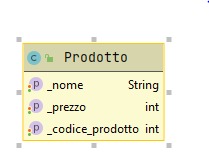
**Ordini**

E’ una classe utilizzata nel pattern repository ed è la classe in cui andremo a fare le operazioni su database che riguardano creazione, modifica e cancellazione dell’ordine.



**Prodotto**

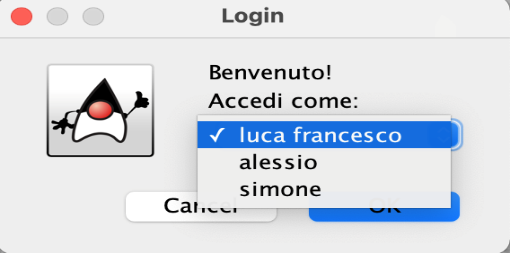
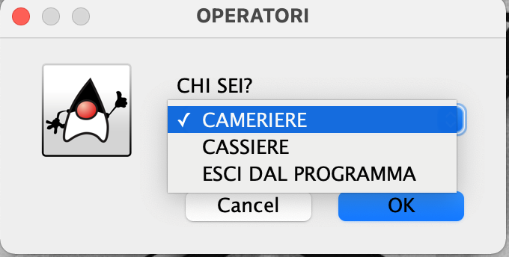
La classe Prodotto è una classe che avrà memorizzate le informazioni sul prodotto che andremo ad utilizzare nella classe Ordini per effettuare le varie query per le operazioni del pattern repository

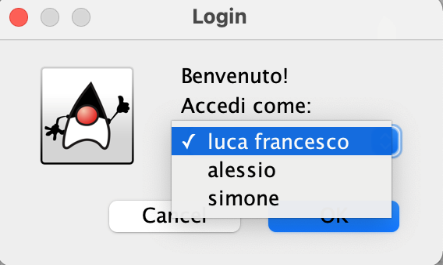


**Ordine, Tavolo, Prodotto-Ordine**

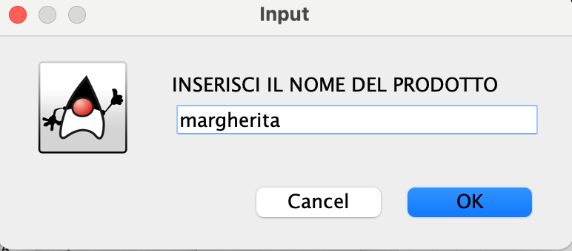
Sono classi in cui andremo solo ad eseguire i metodi set e get.

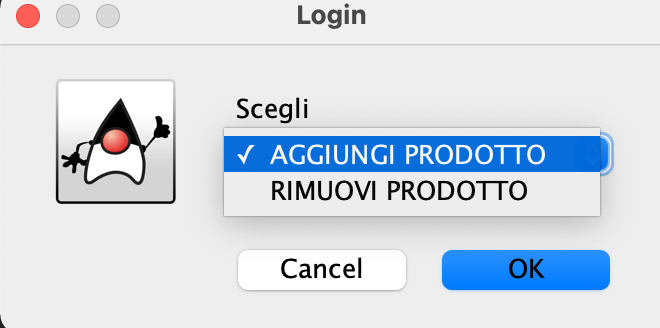
**UTILIZZO DELL’INTERFACCIA GRAFICA**  
  
Per realizzare facilmente finestre di input, allarme o conferma è stato utilizzato **javax.swing.JOptionPane.**  
Esso mette a disposizione tre tipi di pannelli: *Confirm, Input* e *Message Dialog.*  
*Il primo* viene usato quando si chiede all’utente di effettuare una scelta tra un gruppo di possibilità, il *secondo* quando si richiede l’inserimento di una stringa di testo, il *terzo* invece quando si vuole informare l’utente. Attraverso una semplice riga di codice è possibile avere una di queste finestre.  
Un esempio è quando l’utente effettua il login sia per scegliere cameriere o cassiere, sia per scegliere quale cameriere si sta loggando.



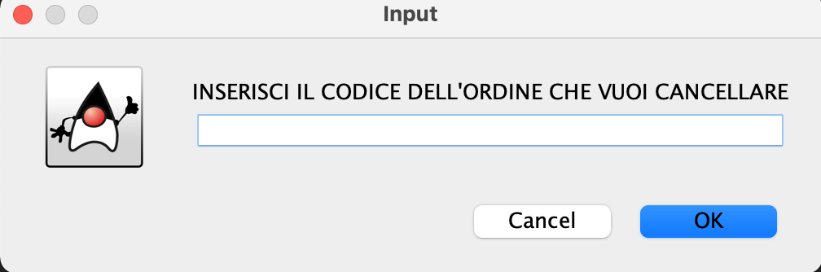
Quando poi si selezionerà un cameriere avremo la scelta dell’azione 

Successivamente dovremo andare ad inserire nel caso di creazione un prodotto finché non inseriremo ‘fine’



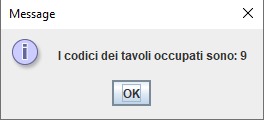
Se invece scegliamo modifica ordine dobbiamo andare a selezionare una delle due voci (aggiungi o rimuovi prodotto) e poi inserire il nome del prodotto e il numero dell’ordine

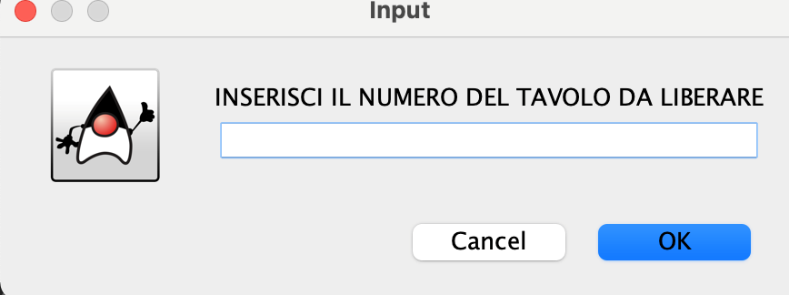
Scegliendo invece cancella ordine dovremmo andare a inserire il codice dell’ordine da cancellare

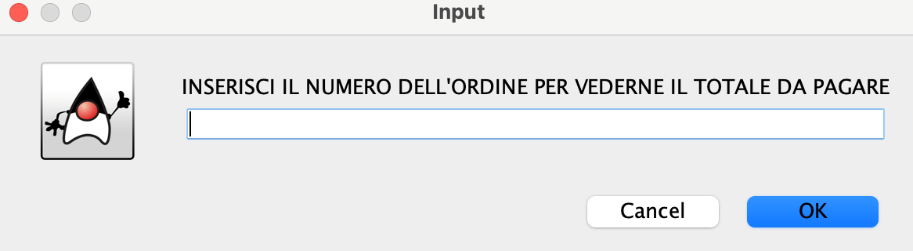


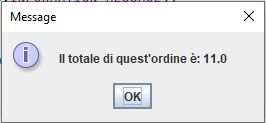
Se invece al login iniziale scegliamo cassiere non dovremmo poi selezionare il cassiere dato che si baserà su due turni, ci sarà un confronto tra l’ora attuale e le 16.43 del pomeriggio, dove prima e dopo quell’ora ci saranno due cassieri diversi

Al cassiere verranno mostrati prima i tavoli occupati e poi verrà chiesto il tavolo da liberare





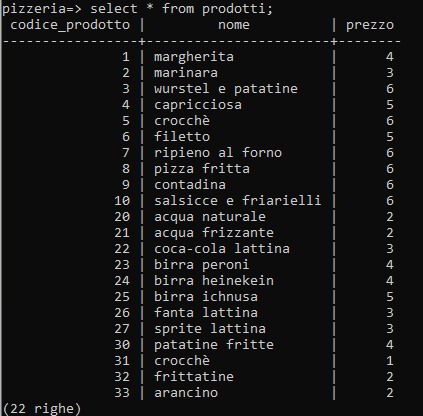
E successivamente il numero dell’ordine di cui visualizzare il totale 



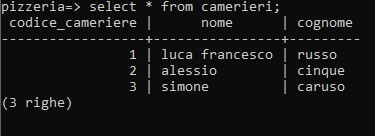
**DATABASE**

Per la gestione del progetto abbiamo utilizzato un database postgres.

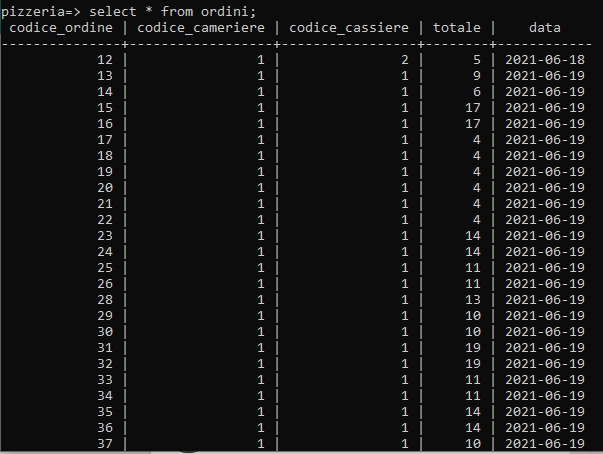
Prodotti:



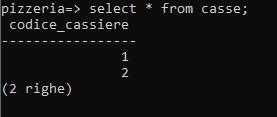
Camerieri:



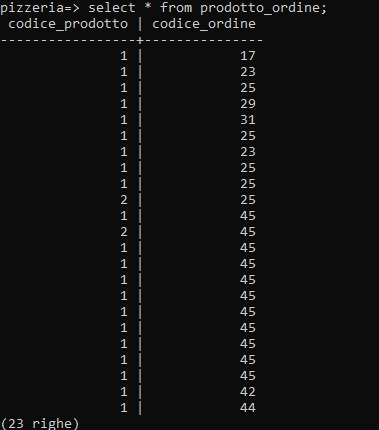
Ordini:



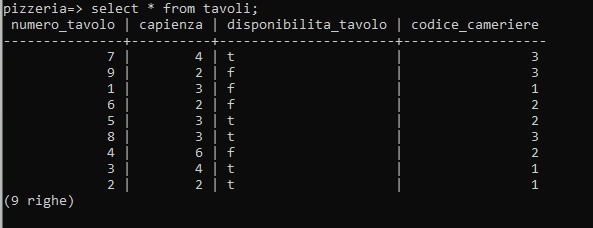
Casse:



Prodotto-Ordine:



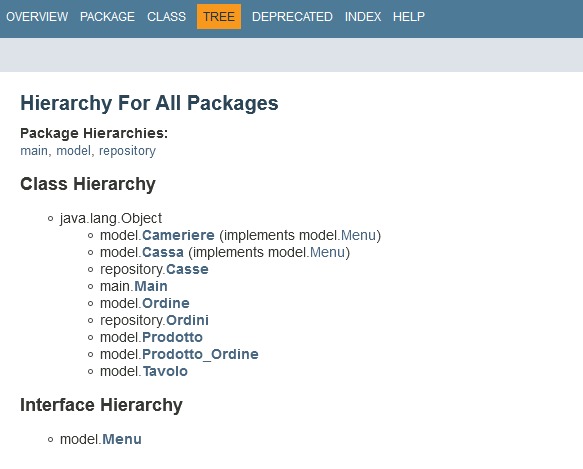
Tavoli:



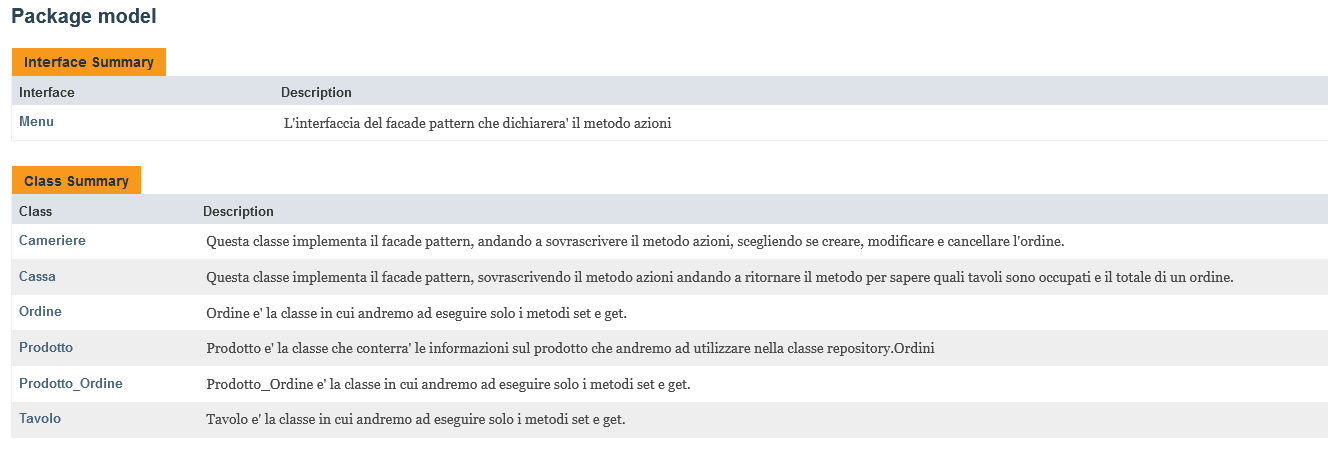
**JAVADOC**  
Javadoc ci permette di documentare tutte le informazioni che riguardano le classi implementate.

Package:

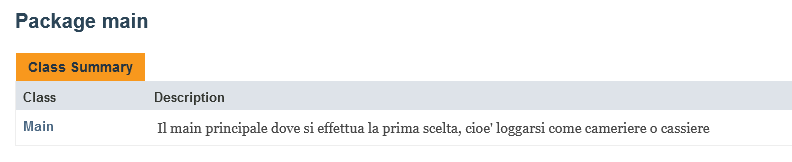
Gerarchia:



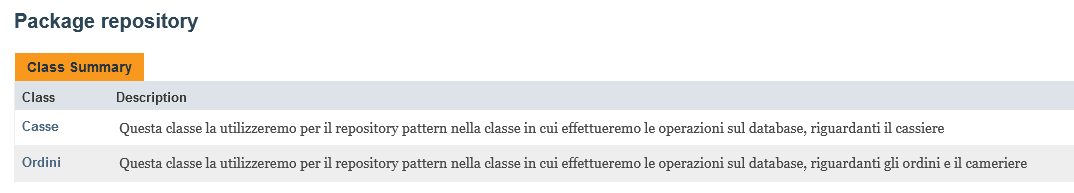
Summary package model:

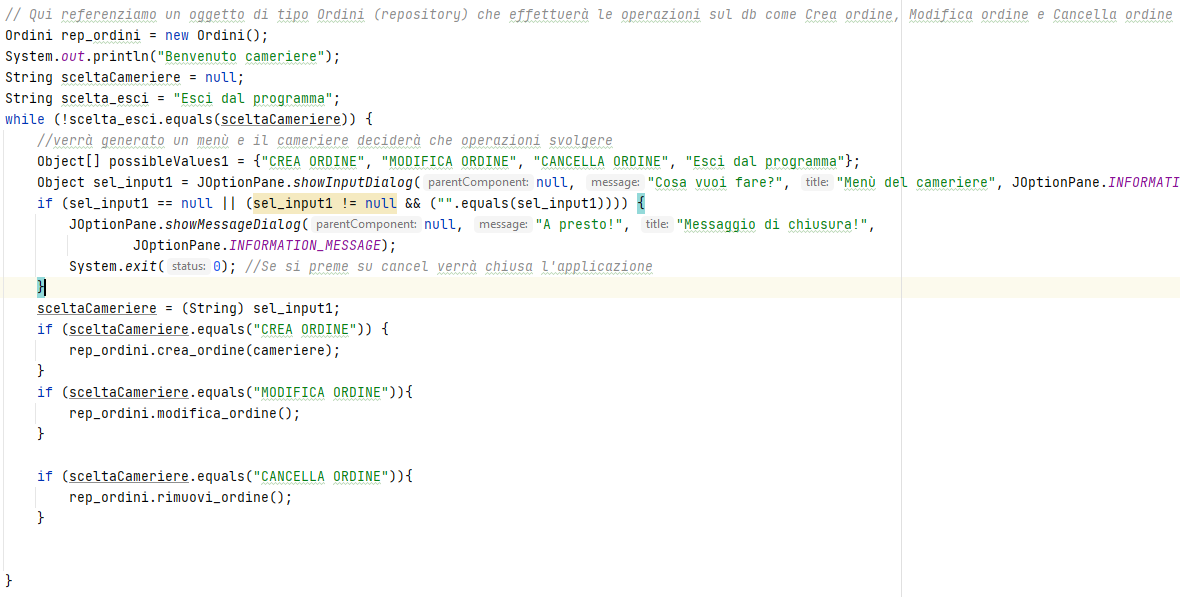


Summary package main:



Summary package repository:



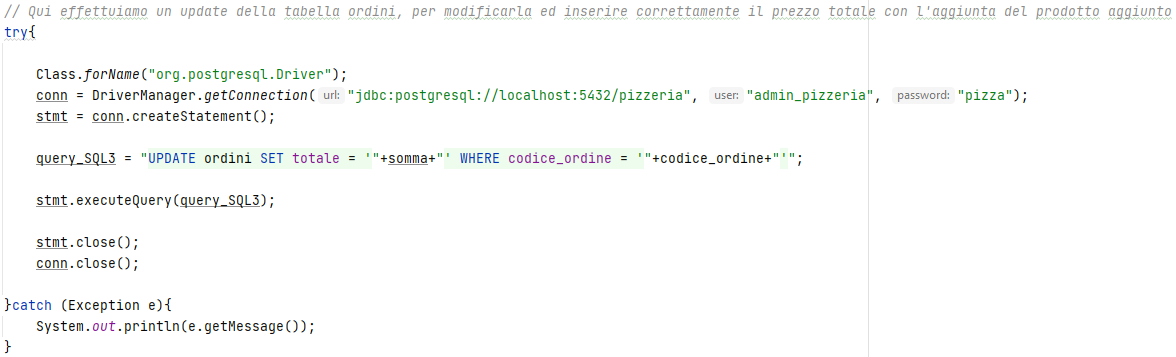


La parte fondamentale del cameriere è questa piccola implementazione, gestita tramite un while che ripete le operazioni che può fare il cameriere, come:

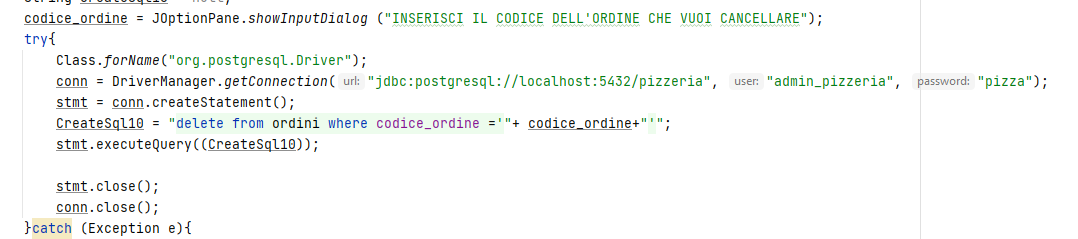
* CREA ORDINE: in cui viene creato l’ordine
* MODIFICA ORDINE: in cui viene modificato l’ordine
* CANCELLA ORDINE: in cui viene cancellato l’ordine



In questo caso, gestiamo il crea ordine tramite l’inserimento di prodotti da parte del cameriere per poi scrivere “fine” nel momento in cui l’ordine doveva terminare. Utilizzo il Try Catch per gestire l’eccezione nel momento in cui ci dovesse essere un problema nella creazione dell’ordine.



In quest’altro caso, utilizziamo il modifica ordine per far si che a seconda della scelta che sarà o aggiunta o rimuovi prodotto, andremo a modificare la tabella, per poi sottrarre i valori in caso di rimuovere un prodotto o addizionare i valori in caso di aggiunta di un prodotto.



In questo caso, utilizziamo il cancella ordine per cancellare completamente l’ordine, inserendo appunto il codice dell’ordine da cancellare.

Il cassiere invece può verificare tutti i tavoli occupati per liberarli e poi può verificare il totale dell’ordine.

