Membri progetto: Alessandro Vaprio matricola 20022545

Progetto Ingegneria del Software

Cockatil-Machine
Il cliente ha richiesto lo sviluppo di un piccolo applicativo.
Il progetto richiesto deve essere in grado di implementare gli ordini dei cocktail o di bevande.
Data un'interfaccia utente, composta da bottoni, l'utente potrà scegliere I cocktail che desidera ordinare. A quel punto potrà vedere il costo e la lista dei cocktail, si potrà procedere, oppure annullare l'ordine.
L'obbiettivo è che l'utente potrà in grado di poter ordinare le proprie bevande in autonomia, in modo che le comande arrivino al bar (per ritiro) e in cassa (per il pagamento). In questa prima fase, I messaggi sono in lingua inglese e valuta americana, in futuro si prospetta il multilingua.
Nelle sezioni sottostanti verrano esplicate scelte e vari scenari.
Allo stato attuale, il cliente, ha già altri software per gestire le comande provenienti da altri dipartimenti, vuole integrare cocktail.

1.3.3 Requisiti
L'utente, indipendetemente che sia utilizzato un totem o un palmare touch, sarà in grado di ordinare le bevande. Verranno scritti sotto nel dettaglio.

1.1 Requisiti non funzionali

RNF01: il sistema è composto da due parti: dal database dove vi sono salvate le informazioni e dall'applicativo client per poter eseguire l'ordine.
RNF02: Verrà utilizzata una grafica semplice e quanto più possibile intuitiva.
RNF03: Il cliente potrà vedere solo l'interfaccia utente.
RNF05: L'applicazione è pensata per essere veloce e reattiva.
RNF06: Per essere veloce sarà data priorità alla leggerezza per garantire un migliore funzionamento del sistema ospite.
RNF07: Il linguaggio scelto è Java, che garantisce una buona portabilità ed è ampiamente usato.
RNF08: Il sistema sarà costruito in modo da poter essere facilmente adattabile e revionabile.

1.2 Requisiti Funzionali

1.2.1 RF01: La Schermata presenta una serie di bottoni, corrispondenti alla lista dei cocktail disponibili attualmente.
1.2.2 RF02: Selezionando un cocktail, verrà aggiornata, sulla destra, una lista con I cocktail selezionati e il prezzo.
1.2.2 RF02: Il prezzo aumenterà in maniera corrisposta alle bevande selezionate.
1.2.3 RF03: Il tasto "Cancel" cancellerà la lista dei drink e l'importo riportando il sistema ad uno stato iniziale.
1.2.4 RF04: Il tasto "Confirm" scriverà nella tabella scontrini e nella tabella ordini I rispettivi valori. In questo modo al Bar sapranno che cocktail corrispondono a che ordine e alle casse avranno il numero dello scontrino con il relativo importo da pagare.

1.3 Studio di fattibilità
Studiando la situazione, ho concluso che il progetto fosse fattibile nonostante sia l'unica risorsa disponibile. Non sono previste ulteriori attrezzatture aggiuntive. I tempi della richiesta sono stretti ma raggiungibili, lo sviluppo richiede una settimana circa per la prima versione dimostrativa.

L’utente, allo stato attuale ha già altri software per gestire le comande provenienti da altri dipartimenti, manca l’integrazione coi cocktail. Mi pare una scelta molto conveniente, in quanto, potrebbe integrare questa nuova funzione nel sistema esistente.

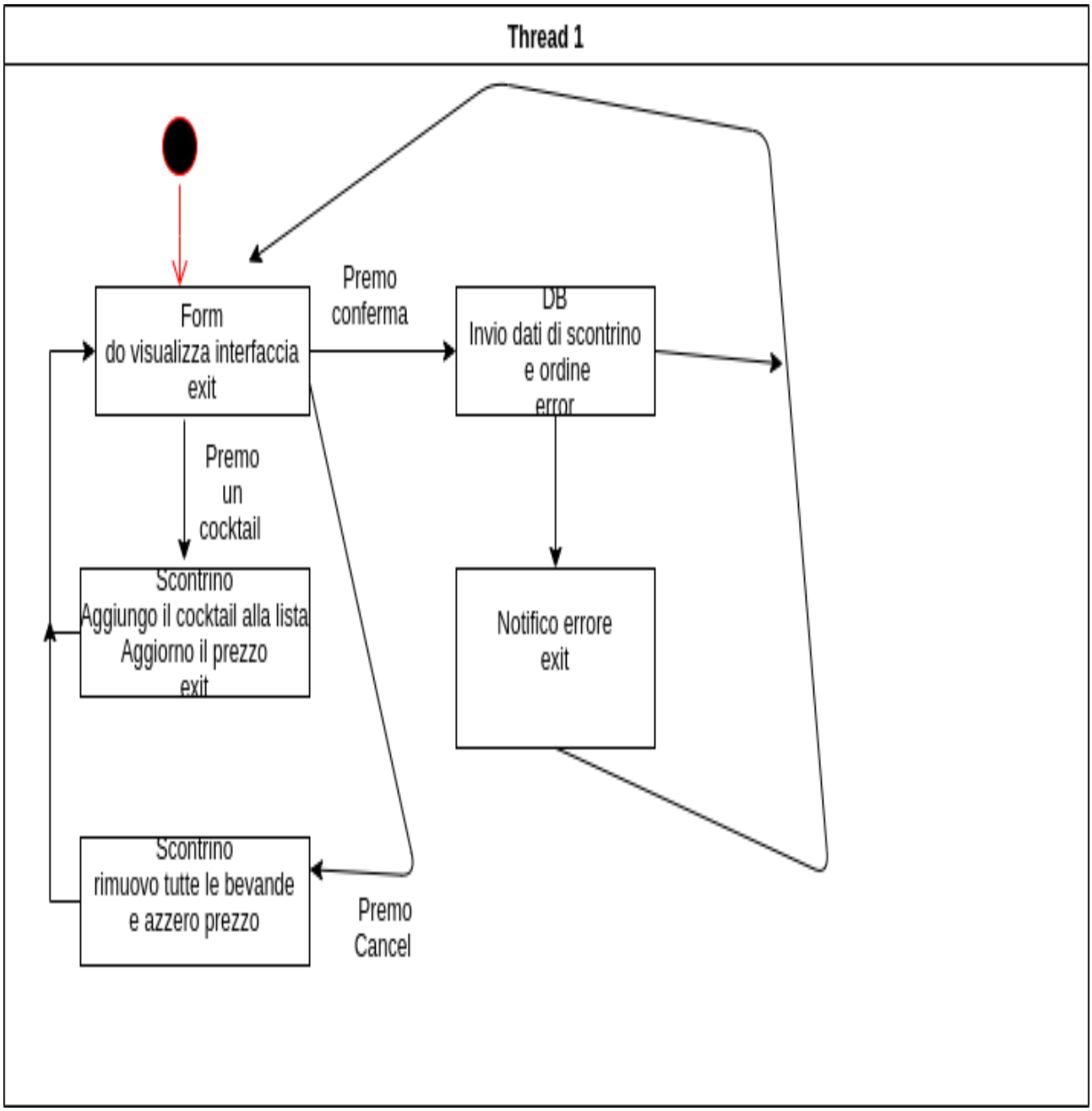
1.4 Evoluzione
Valuteremo, di cambiare l’interfaccia in una html5 con l’utlizzo di CSS. In questo modo sarà più gradevole e implementabile. Inoltre faremo in modo di aggiungere una sezione admin per cambiare prezzo ed eventualmente inserire nuovi cocktail al bisogno. Verrà aggiunto il supporto multi lingua e multivaluta.

Per rendere più chiara l’esecuzione del programma, verranno illustrati diversi scenari.

Scenario “Conferma”
• Flusso di eventi:
1. L’utente sceglie un cocktail
2. Il cocktail scelto viene aggiunto alla lista delle bevande e viene segnalato il prezzo.
3. L’utente può ripetere il passo 1, se accade verrà ripetuto il passo 2 che aggiornerà il prezzo e la lista.
4. Viene premuto il tasto “Confirm” che invierà gli ordini al bar e in cassa.
5. Viene mandato un messaggio per notificare il numero di ordine corrispondente
• Eccezioni:
1. Nel passo 5: errore nella scrittura database viene noticato all’utente.

Scenario “Conferma senza drink”
• Flusso di eventi:
1. Viene notificato di selezionare almeno un drink.

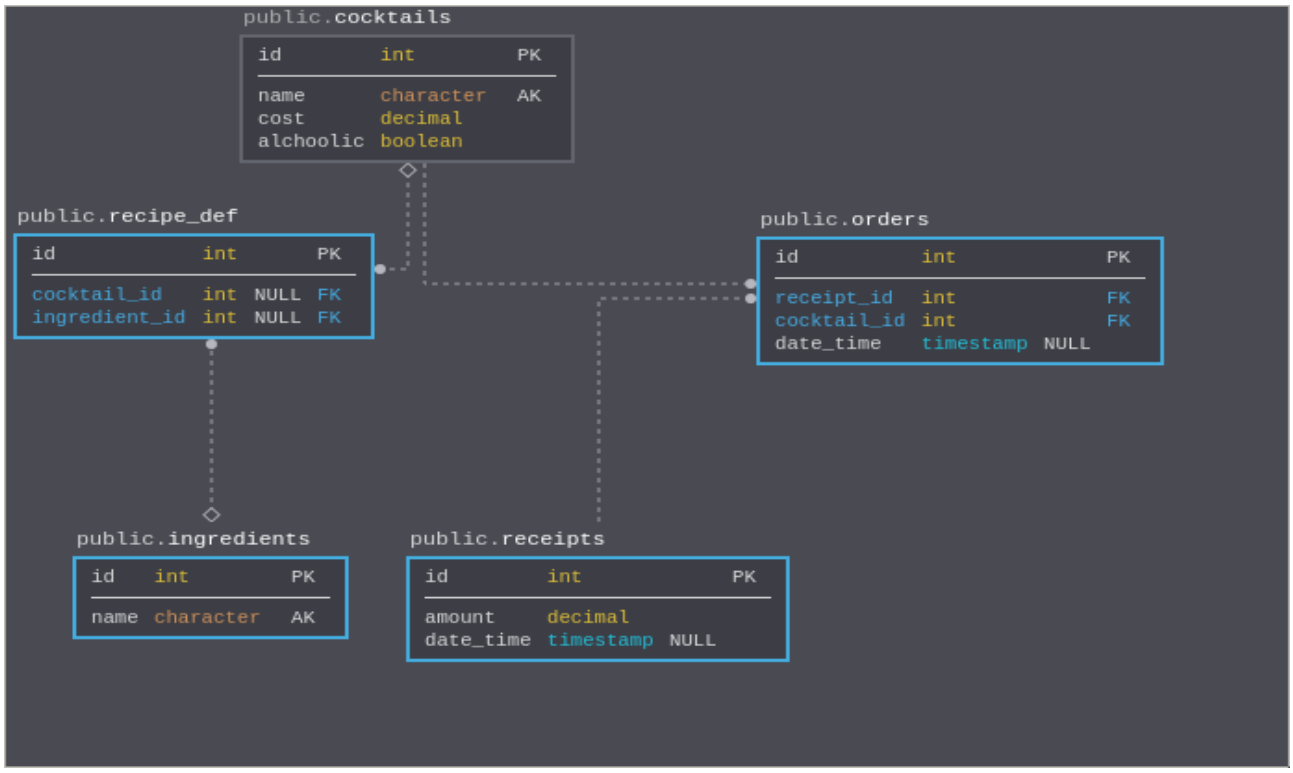
Scenario “Cancel”
• Flusso di eventi:
1. Ogniqualvolta viene premuto azzera il contenuto della lista dei cocktail e del prezzo.



DATABASE

IL database scelto è PostgreSQL per la sua versatilità e potenza.
Le tabelle sono 5, collegate tra loro:
- cocktails per identificare I singoli cocktail con il loro nome, prezzo e se sono alcolici.
- orders contiene le linee con l'identificativo degli ordini effettuati.
- recipe_def identifica l'associazione tra gli ingredienti e I cocktail
- ingredient contiene la lista degli ingredienti
- receipts identifica gli scontrini, l'id viene usato come numero ordine per la tabella orders.

Nella cartella del progetto sono present gli script per la generazione del del database e un seed iniziale.



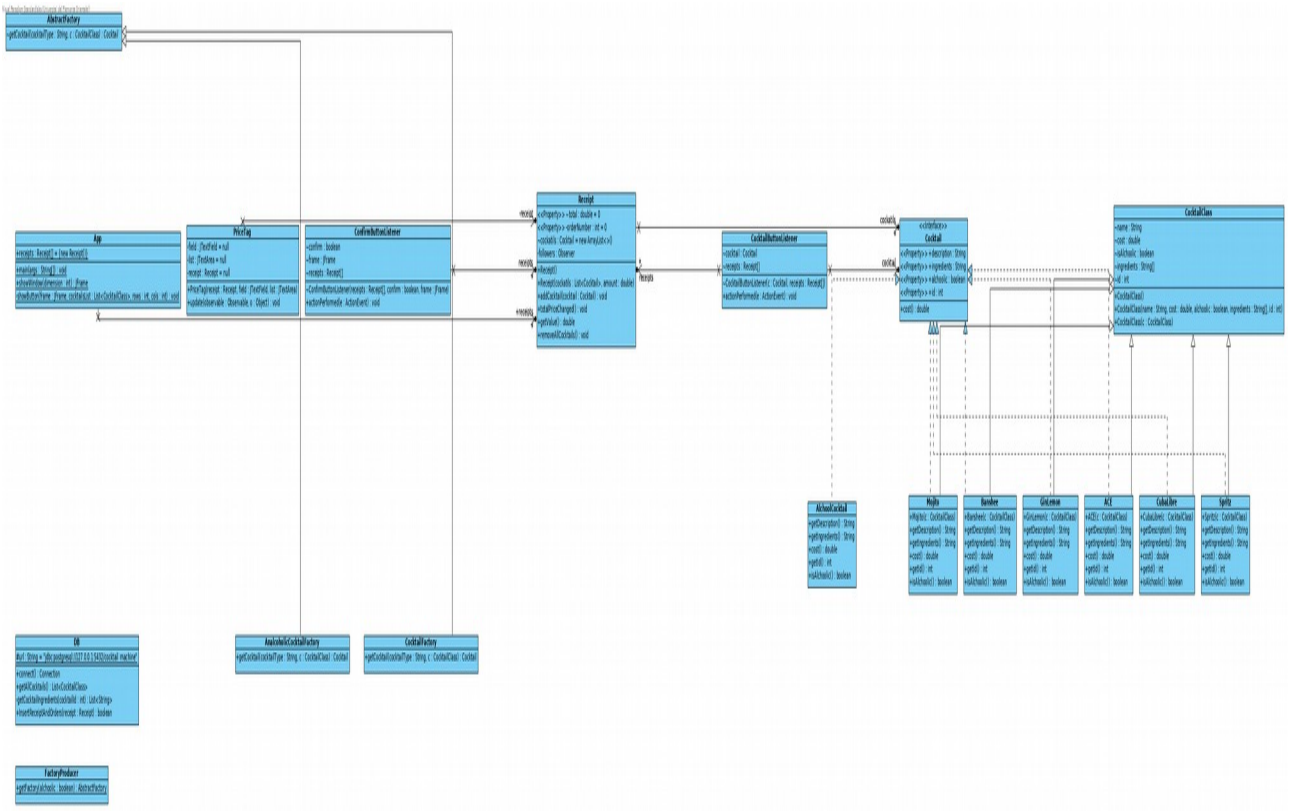
Qui sotto il modello relazionale.

Di seguito schema UML delle classi del progetto.
Per questo progetto sono stati utilizzati due pattern principali: Abstract Factory e obeserver.
Abstract factory è stato scelto per poter rendere polimorfico il comportamento dei diversi cocktail senza sapere quale cocktail stiamo trattando (Si considera solo la l'interfaccia Ccktail). Questo rende possibile cambiare il comportamento, nel nostro caso, della descrizione, ove nel caso di analcolici , viene specificato la categorai analcolica.
Il pattern observer invece è utilizzato per avere il dato del prezzo sempre e della lista dei cocktail aggioranti in tempo reale.

Le classi coinvolte nell'Abstract Factory sono:
- Abstract Factory
- Factory Producer, decide che tipo di factory user tra alcolico e analcolico.
- CocktailFActory e AnalcholicFactory istanziano le classi ai cocktail corrispondenti.
- CubaLibre, Mojito, ACE, ecc implementano l'interfaccia Cocktail (con I metodi) ed estendono la classe cocktailClass con alcuni campi.
- L'interfaccia Cocktail contiene I metodi da implementare

Le classi coinvolte nel pattern observer:
- Receipt estende Observable in modo da essere seguito da classi observer e di notificare loro il cambiamento.
- PriceTag è la classe observer, che viene notificata del cambiamento. Nella metodo update vengono aggiornate la lista dei cocktail e il prezzo.
- In App, nel metodo main, viene aggiunto un observer per una classe istanziata di tipo PriceTag.

Sotto lo schema UML corrispondente, verrà fornito all’interno del progetto, lo stesso in formato png.



Versioning
Pre poter gestire al meglio il progetto, è stato scelto github come repository.
https://github.com/alessandrovaprio/cocktail-machine.git