



*Simone Contini, Danilo Dolce*

*Progetto d'esame di Metodi Avanzati per la Programmazione*

# Indice

Indice .....	2
Abstract .....	3
1. Introduzione .....	4
2. Il Simulatore .....	5
2.1 Guida al software .....	5
2.1.1 Modalità Manager .....	5
Pre-Configurazione del ristorante .....	5
Home Manager.....	6
Impostazioni generali.....	7
Impostazioni tavoli.....	8
Menu .....	9
2.1.2 Modalità Cliente .....	16
Autenticazione.....	17
Home Cliente.....	18
Prenotazione .....	18
I miei ordini .....	22
2.2 Raccolta dei requisiti: casi d'uso .....	25
2.3 Diagramma delle classi e motivazione delle scelte .....	26
2.3.1 Builder .....	27
2.3.2 Chain of Responsibility .....	30
2.3.3 State .....	32
2.3.4 Command.....	34
2.3.5 Decorator.....	36
3. Conclusione .....	38
3.1 Riferimenti e strumenti di sviluppo utilizzati.....	38

## **Abstract**

Abbiamo creato un simulatore di ristorante, *Restaurant Simulator*, come progetto abilitante per l'esame di Metodi Avanzati per la Programmazione, sfruttando i vantaggi dell'utilizzo dei Design Pattern Object-Oriented nella progettazione e alcune tecniche avanzate di programmazione in Java per lo sviluppo. Tramite l'implementazione di questi strumenti siamo, infatti, riusciti a creare un software funzionale e facilmente manutenibile.

# 1. Introduzione

La progettazione di un software object-oriented è un lavoro molto complesso che senza un adeguato design strutturale potrebbe portare a problemi di vario genere, quali ad esempio, scarsa manutenibilità del software e difficoltà nell'effettuare il refactoring e il riutilizzo del codice. Per ovviare a questi problemi sono stati ideati i design pattern, ovvero degli schemi progettuali per la programmazione orientata agli oggetti.

Per il nostro software, *Restaurant Simulator*, un simulatore di ristorante, realizzato per l'esame di Metodi Avanzati per la Programmazione, abbiamo utilizzato tutti i design pattern che ritenevamo opportuni per la sua struttura, senza forzare l'utilizzo di altri modelli ritenuti poco idonei. Tra questi pattern, abbiamo considerato il builder per la creazione delle entità ristorante e utente; il chain of responsibility per la simulazione del pagamento in contante dell'ordine; il decorator per la personalizzazione dei prodotti ordinabili dal menu; lo state per modificare lo stato degli ordini; infine il command per la gestione dei comandi di stampa e salvataggio su file del menu e delle ricevute. Ciascun pattern utilizzato ha una sua motivazione nell'uso di tale software; alcuni sono stati usati più volte, mentre altri sono stati usati una sola volta.

Il software è stato scritto in linguaggio Java, sfruttando alcune delle sue tecniche avanzate fornite con la versione 8, attraverso l'uso dell'ide Apache NetBeans e l'ausilio delle librerie Swing e JCalendar per le interfacce grafiche.

*Restaurant Simulator* consente all'utente di immedesimarsi nel ruolo di un manager di un ristorante o di un suo semplice cliente. Il software pertanto si divide in due modalità selezionabili al suo avvio: la *Modalità Manager* e la *Modalità Cliente*.

Nella *Modalità Manager*, l'utente, nel ruolo di *Manager*, crea il suo ristorante e ne configura i vari aspetti, quali il nome, il menu e i tavoli. Inoltre ha la possibilità di visualizzare la lista dei suoi clienti; lo storico degli ordini e, per ciascuno di essi, il dettaglio e la ricevuta; la disponibilità di tavoli per un dato giorno e per una specifica ora. Questa modalità contiene uno degli elementi fondamentali di questo software, il *comando di simulazione*. Quest'ultimo, selezionata una data, permette di simulare il completamento degli ordini antecedenti alla data scelta.

Nella *Modalità Cliente*, accessibile esclusivamente nel caso in cui sia stata effettuata una prima configurazione del ristorante, l'utente, nel ruolo di *Cliente*, può effettuare una prenotazione nel ristorante creato, visualizzare la lista dei suoi ordini e, per ciascuno di questi, il dettaglio e la ricevuta, con la possibilità di rilasciare anche un feedback. Quest'ultimo si riferisce alla valutazione di tre aspetti riscontrati durante la consumazione dell'ordine: Servizio, Rapporto qualità\prezzo e Ambiente.

Il risultato finale ottenuto con *Restaurant Simulator* è quello di un software di simulazione a tutti gli effetti, che grazie all'uso dei design pattern può ritenersi facilmente manutenibile e rifattorizzabile.

## 2. Il Simulatore

### 2.1 Guida al software

Il software che abbiamo implementato è un simulatore di ristorante che permette a ciascun utente di immedesimarsi nel manager di un ristorante, che può essere inteso come il proprietario o il gestore, oppure in un suo cliente.

All'avvio del software pertanto, l'utente deve scegliere se accedere in *Modalità Manager* o in *Modalità Cliente*.



Figura 2.0: L'interfaccia principale di *Restaurant Simulator*.

#### 2.1.1 Modalità Manager

La Modalità Manager rappresenta quella modalità in cui l'utente si immedesima nella figura di *Manager* per la gestione e la configurazione di un ristorante.

##### Pre-Configurazione del ristorante

Se non esiste alcuna configurazione del ristorante, la **fase di pre-configurazione** è il primo step obbligatorio proposto dal sistema per la creazione di un ristorante.

Questa fase include un'introduzione alla *Modalità Manager* e un campo obbligatorio in cui il *manager* imposta il nome del ristorante che intende creare, la cui lunghezza deve essere compresa tra 1 e 20 caratteri.

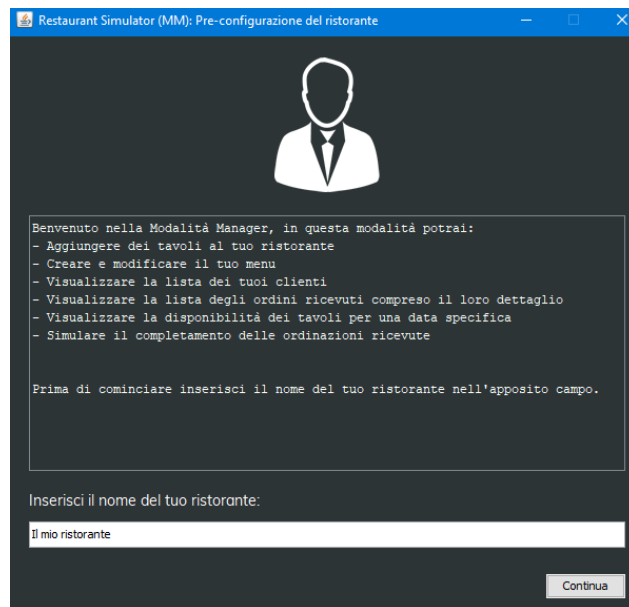


Figura 2.1: Interfaccia di pre-configurazione del ristorante.

Inserito il nome del ristorante, alla pressione del tasto *Continua*, il *sistema* verifica la correttezza del campo inserito e reindirizza alla *Home Manager* in caso di esito positivo, viceversa genera un messaggio di errore.

## Home Manager

La *Home Manager* rappresenta l'interfaccia principale della *Modalità Manager*. In questa fase del software il *manager* può:

- Visualizzare il nome del suo ristorante;
- Accedere alle impostazioni generali del ristorante tramite la voce del menu *Impostazioni ► Impostazioni generali*;
- Accedere alle impostazioni di configurazione dei tavoli del ristorante tramite la voce del menu *Impostazioni ► Impostazioni tavoli*;
- Visualizzare e configurare il menu del ristorante tramite il click sul pulsante *Menu*;
- Visualizzare lo storico degli ordini del ristorante tramite il click sul pulsante *Storico Ordini*;
- Visualizzare la lista dei clienti registrati al ristorante tramite il click sul pulsante *Clienti*;
- Visualizzare la disponibilità dei tavoli per una determinata data e per una certa ora tramite il click sul pulsante *Agenda*;
- Simulare il completamento degli ordini del ristorante antecedenti ad una data specifica tramite il click sul pulsante *Simula*.



Figura 2.2: Interfaccia *Home Manager*.

## Impostazioni generali

Selezionando la voce del menu in alto *Impostazioni* ► *Impostazioni generali* dalla *Home Manager*, il software mette a disposizione un'interfaccia grafica che permette di:

- Customizzare il nome del ristorante;
- Resetare il software alle impostazioni di fabbrica.

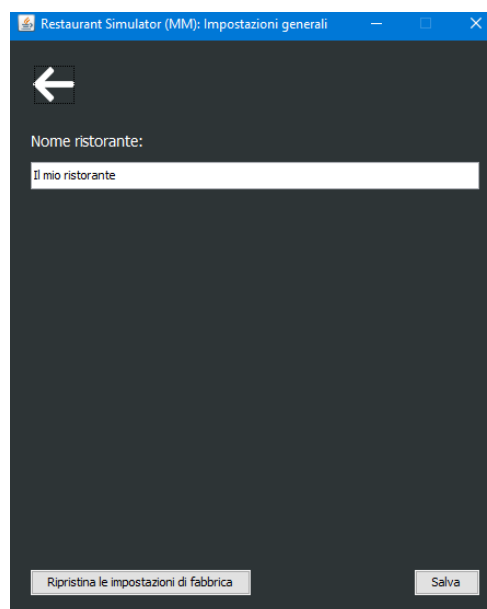


Figura 2.2: Interfaccia *Impostazioni generali*.

Il *ripristino delle impostazioni di fabbrica* consente di resettare il software ai suoi dati di origine. Questa operazione, in quanto delicata, avviene tramite il click sul relativo pulsante e la conferma sull'apposito popup.

La customizzazione del nome del ristorante avviene attraverso la ricompilazione del relativo campo con il vincolo di inserire un nome compreso tra 1 e 20 caratteri. La modifica viene confermata attraverso il click sul pulsante *Salva*. Se non vi sono errori il sistema produrrà un messaggio di avvenuto salvataggio, altrimenti un messaggio contenente gli errori relativi.

In qualsiasi momento il *manager* può ritornare alla *Home Manager* cliccando sul pulsante *Back* in alto a sinistra.

## Impostazioni tavoli

Selezionando la voce del menu in alto *Impostazioni* ► *Impostazioni tavoli* dalla *Home Manager*, il software mette a disposizione un'interfaccia grafica che permette di visualizzare i tavoli del ristorante creati in precedenza e di configurarne e crearne degli altri.

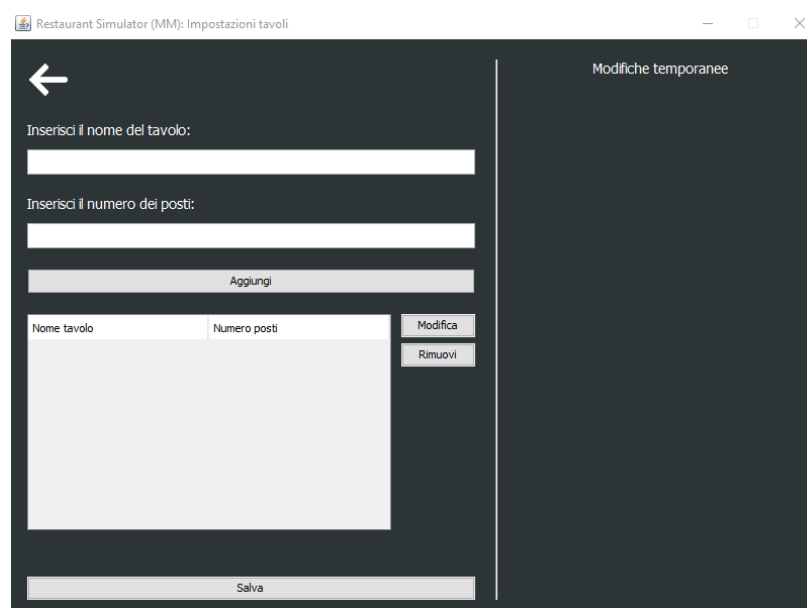


Figura 2.3: Interfaccia *Impostazioni tavoli*.

Per aggiungere un nuovo tavolo al ristorante, il *manager* effettua la compilazione dei seguenti campi:

- ***Inserisci il nome del tavolo***: il nome del tavolo che si vuole inserire con la condizione che *ciascun tavolo deve avere un nome differente*;
- ***Inserisci il numero di posti***: il numero di posti a sedere del tavolo che si vuole inserire con la condizione che *ciascun tavolo può avere un numero di posti compreso tra 1 e 10*.

Successivamente alla compilazione dei precedenti, il *manager* clicca sul pulsante *Aggiungi* per inserire il tavolo nella tabella riepilogativa dei tavoli del ristorante.

Selezionando uno dei nuovi tavoli appena aggiunti e cliccando sul pulsante *Modifica*, il sistema mostra un'interfaccia per la sua riconfigurazione (vedi Figura 2.4) nella sezione



*Modifiche temporanee.* Qui il *manager* può impostare i nuovi valori del tavolo da modificare, confermare le modifiche cliccando sul pulsante *Modifica* relativo, oppure annullarle attraverso il click sul pulsante *Annulla*. Le modifiche vengono accettate solo nel caso in cui i valori inseriti rispettano le condizioni precedentemente fissate per i tavoli.

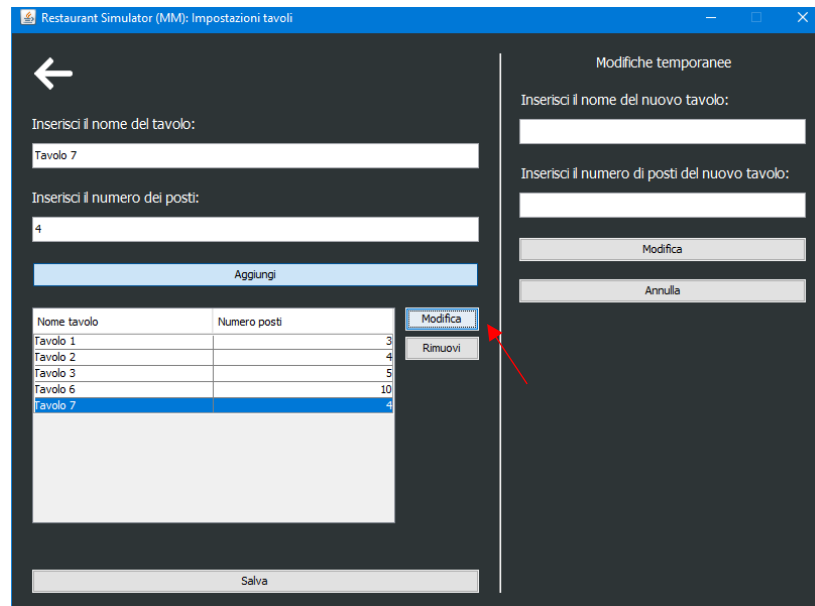


Figura 2.4: Modifica di un tavolo non ancora salvato.

Inoltre, selezionando uno dei nuovi tavoli appena aggiunti e cliccando sul pulsante *Rimuovi*, il *manager* può effettuare la rimozione.

Cliccando sul pulsante *Salva*, il *manager* aggiunge definitivamente i nuovi tavoli inseriti nel proprio ristorante. Una volta confermato il salvataggio, i tavoli inseriti non potranno più essere modificati o rimossi.

In qualsiasi momento il *manager* può ritornare alla *Home Manager* cliccando sul pulsante *Back* in alto a sinistra.

## Menu

Cliccando sul pulsante *Menu* della *Home Manager*, il software mette a disposizione un'interfaccia grafica che fornisce le seguenti funzionalità per il menu del ristorante:

- Visualizzazione;
- Configurazione;
- Stampa;
- Salvataggio su file.



Figura 2.5: Interfaccia *Menu*.

Nel dettaglio, l'area denominata *Menu* all'interno di questa interfaccia mostra la lista di tutti i piatti suddivisi per categoria (antipasti, primi, secondi, frutta, dessert e bevande) che compongono il menu del ristorante.

Il pulsante *Stampa* permette la stampa su carta del menu; il pulsante *Scarica*, invece, ne permette il salvataggio su un file di testo.

In qualsiasi momento il *manager* può ritornare alla *Home Manager* cliccando sul pulsante *Back* in alto a sinistra.

Infine, cliccando sul pulsante *Impostazioni*, il *manager* può configurare il menu del ristorante. La schermata di configurazione generata dal click sul pulsante è mostrata nella Figura 2.6.

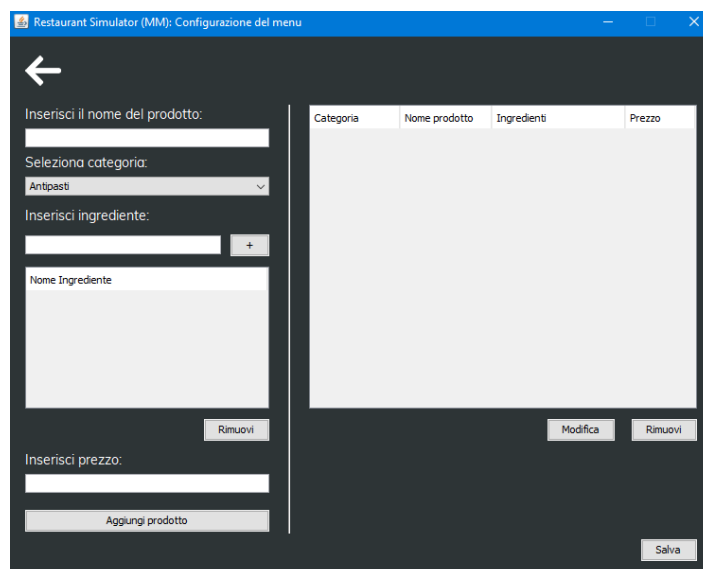


Figura 2.6: Interfaccia *Configurazione del menu*.

Da questa interfaccia, il manager può visualizzare i prodotti del menu già esistenti e inserirne di nuovi.

Per aggiungere un nuovo prodotto, il *manager* deve effettuare i seguenti passaggi:

- Inserire il nome del piatto nel campo *Inserisci il nome del prodotto*;
- Selezionare una delle categorie di default (antipasti, primi, secondi, frutta, dessert e bevande) fornite dal sistema attraverso il campo di selezione relativo;
- (opzionale) Aggiungere, uno per volta, gli ingredienti descrittivi del prodotto, specificandone il nome e cliccando sul pulsante di aggiunta raffigurato dal simbolo più (+). Inoltre è possibile effettuarne la rimozione, selezionandoli uno per volta nella *tabella degli ingredienti del prodotto* e premendo il pulsante *Rimuovi*.
- Inserire il prezzo del prodotto nel campo *Inserisci prezzo*;
- Cliccare sul pulsante *Aggiungi prodotto*.

Il software verifica, dunque, la correttezza dei dati inseriti effettuandone una validazione. Se la validazione ha esito positivo, il prodotto in questione sarà aggiunto nella *tabella dei prodotti del menu*, presente nella sezione destra di questa interfaccia.

Il sistema permette di editare e rimuovere esclusivamente i prodotti aggiunti temporaneamente e di cui non è stato effettuato per l'appunto il salvataggio.

Il *manager* può modificare un prodotto effettuandone la selezione dello stesso nella *tabella dei prodotti* e cliccando sul pulsante *Modifica*. Il sistema controlla se il prodotto selezionato è idoneo per la modifica e in caso di esito positivo, lo rimuove dalla *tabella dei prodotti* per predisporlo nei campi relativi all'aggiunta di un nuovo prodotto.

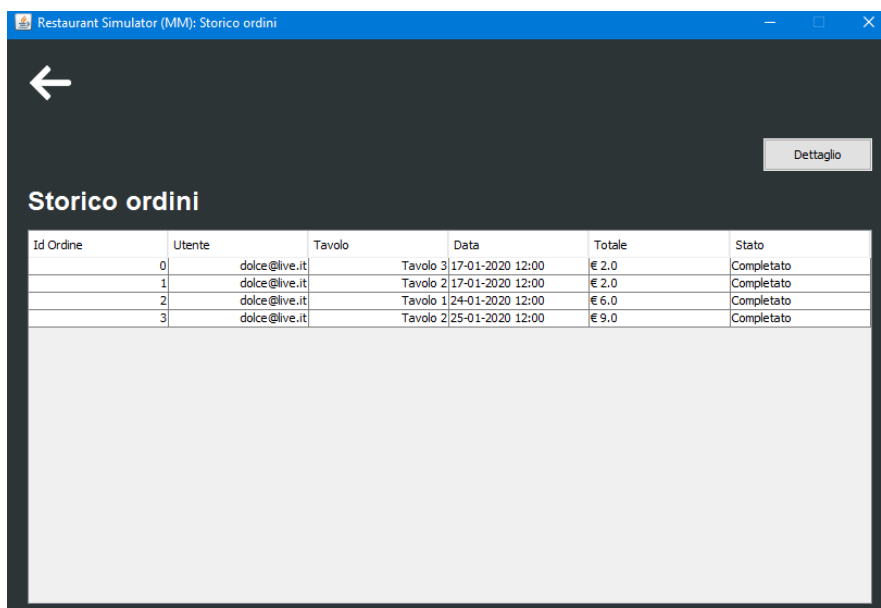
Il *manager* può effettuare la rimozione di un prodotto effettuandone la selezione dello stesso nella *tabella dei prodotti* e cliccando sul pulsante *Rimuovi*. Il sistema controlla se il prodotto selezionato è idoneo per la rimozione e in caso di esito positivo, lo rimuove dalla *tabella dei prodotti*.

Cliccando sul pulsante *Salva*, il *manager* aggiunge definitivamente i nuovi prodotti al menu del proprio ristorante. Una volta confermato il salvataggio, i prodotti inseriti non potranno più essere modificati o rimossi.

In qualsiasi momento il *manager* può ritornare alla schermata del *Menu* cliccando sul pulsante *Back* in alto a sinistra.

## Storico ordini

Cliccando sul pulsante *Storico ordini* della *Home Manager*, il software mette a disposizione un'interfaccia grafica che permette di visualizzare lo storico degli ordini del ristorante e per ciascuno di essi il relativo dettaglio.



Id Ordine	Utente	Tavolo	Data	Totale	Stato
0	dolce@live.it	Tavolo 3	17-01-2020 12:00	€ 2.0	Completato
1	dolce@live.it	Tavolo 2	17-01-2020 12:00	€ 2.0	Completato
2	dolce@live.it	Tavolo 1	24-01-2020 12:00	€ 6.0	Completato
3	dolce@live.it	Tavolo 2	25-01-2020 12:00	€ 9.0	Completato

Figura 2.7: Interfaccia *Storico ordini*.

La Figura 2.7 mostra lo storico degli ordini del ristorante. In questa interfaccia il *manager* può osservare alcune informazioni inerenti agli ordini mediante la *tabella Storico ordini*:

- *Id ordine*: l'identificativo dell'ordine;
- *Utente*: l'email dell'utente che ha effettuato l'ordine;
- *Tavolo*: il tavolo assegnato al cliente per la sua consumazione;
- *Data*: la data di consumazione dell'ordine;
- *Totale*: il totale dell'ordine;
- *Stato*: lo stato dell'ordine che può essere di due tipi:
  - **Pagato**: il cliente ha pagato l'ordine ma non ha ancora effettuato la consumazione;
  - **Completato**: il cliente ha pagato l'ordine ed ha effettuato la consumazione.

Selezionando un elemento della tabella *Storico ordini* e cliccando sul pulsante *Dettaglio*, il *manager* può visualizzare dei dettagli aggiuntivi dell'ordine e la corrispettiva ricevuta.

In qualsiasi momento il *manager* può ritornare alla *Home Manager* cliccando sul pulsante *Back* in alto a sinistra.

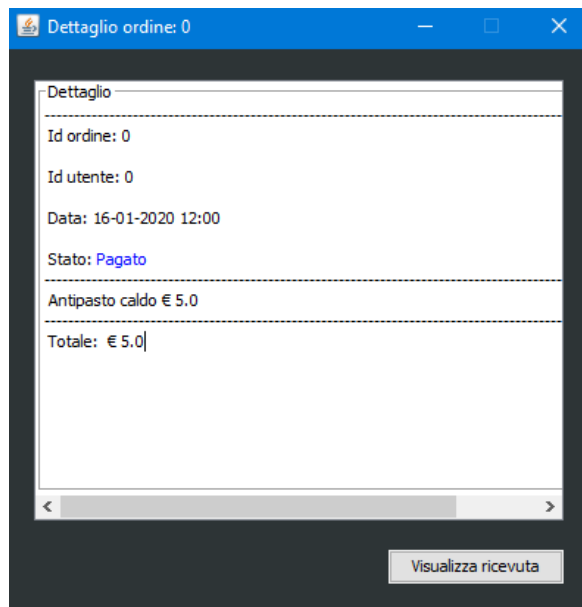


Figura 2.8: Interfaccia *Dettaglio ordine*.

La Figura 2.8, mostra il dettaglio di un ordine selezionato nella tabella *Storico ordini*. In questa interfaccia il *manager* può visualizzare, oltre alle informazioni precedenti, la lista dei prodotti con il relativo prezzo che il cliente ha scelto per il corrispettivo ordine, e cliccando sul tasto *Visualizza ricevuta* potrà (vedi Figura 2.9) *visualizzare, stampare e salvare su file* la ricevuta di quest'ultimo.

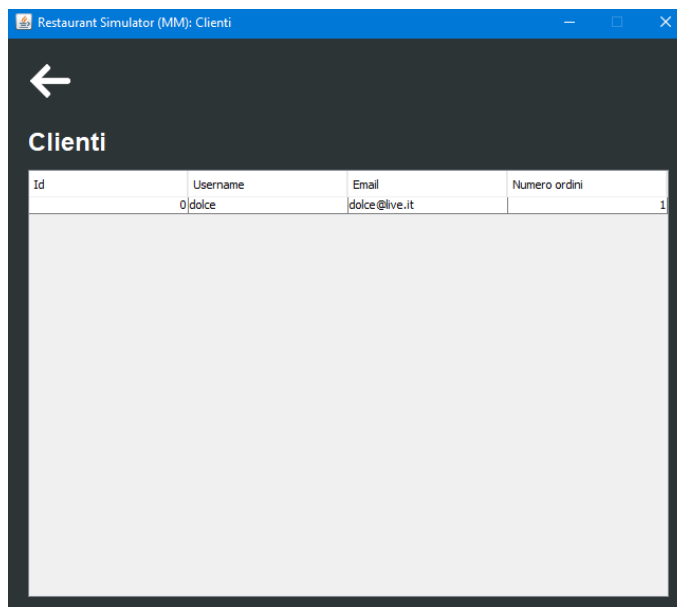


Figura 2.9: Interfaccia *Ricevuta ordine*.

## Clienti

Cliccando sul pulsante *Clienti* della *Home Manager*, il software mette a disposizione un'interfaccia grafica (vedi Figura 2.10) che permette di visualizzare la lista dei clienti del ristorante e per ciascuno di essi le seguenti informazioni:

- *Id*: l'identificativo del cliente;
- *Username*: l'username del cliente;
- *Email*: l'email del cliente;
- *Numero ordini*: il numero degli ordini effettuati dal cliente.



Id	Username	Email	Numero ordini
0	dolce	dolce@live.it	1

Figura 2.10: Interfaccia *Clienti*.

In qualsiasi momento il *manager* può ritornare alla *Home Manager* cliccando sul pulsante *Back* in alto a sinistra.

## Agenda

Cliccando sul pulsante *Agenda* della *Home Manager*, il software mette a disposizione un'interfaccia grafica che permette al *manager* di verificare la disponibilità dei tavoli del ristorante per una specifica data e per una certa ora.

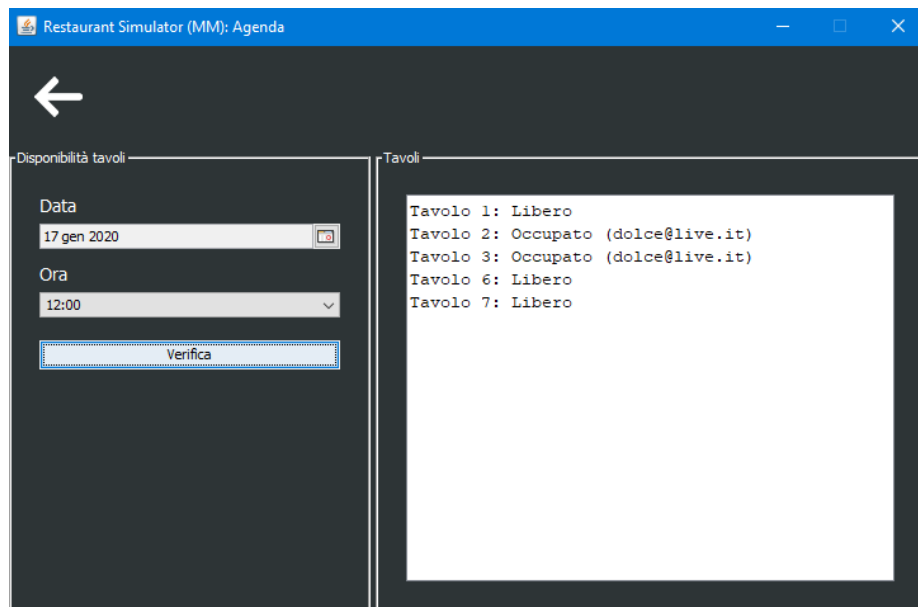


Figura 2.11: Interfaccia *Agenda*.

In questa interfaccia (vedi Figura 2.11), il *manager* per controllare lo stato di tutti i tavoli deve selezionare dal calendario, generato dal click sul pulsante presente nel campo *Data*, il giorno per cui intende effettuare la verifica dello stato dei tavoli, e dal campo *Ora* una delle ore indicate dal sistema, che corrispondono alle possibili ore di prenotazione. Effettuate queste scelte, l'utente deve cliccare sul pulsante *Verifica* affinché il sistema generi nell'apposita area *Tavoli*, la lista dei tavoli con rispettivo stato:

- *Libero*: tavolo disponibile;
- *Occupato*: tavolo occupato. In questo caso insieme allo stato verrà visualizzata l'email del cliente che ha occupato il tavolo.

In qualsiasi momento il *manager* può ritornare alla *Home Manager* cliccando sul pulsante *Back* in alto a sinistra.

## Simula

La *Home Manager* presenta un pulsante denominato *Simula*. Cliccando su di esso, l'utente accede al simulatore principale di *Restaurant Simulator*, che permette di effettuare la simulazione del completamento degli ordini del ristorante, cambiando lo stato di quest'ultimi da *Pagato* a *Completato*. Come già introdotto nei paragrafi precedenti, ciascun ordine infatti presenta due stati:

- *Pagato*: l'utente ha pagato l'ordine ma non ha ancora effettuato la consumazione;
- *Completato*: l'utente ha pagato l'ordine ed ha effettuato la consumazione.

In questo modo il *manager* non ha la necessità di attendere la data di consumazione affinché cambino gli stati degli ordini.

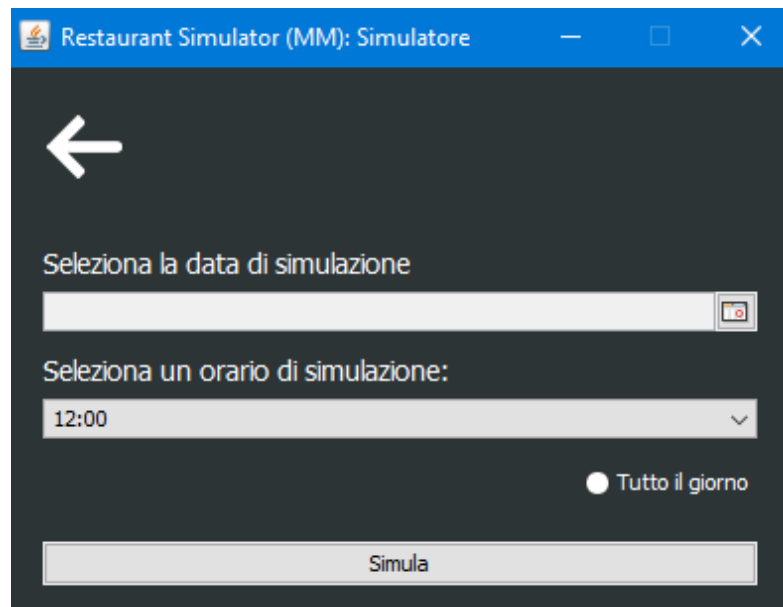


Figura 2.12: Interfaccia *Simulatore*.

La Figura 2.12 mostra l'interfaccia del simulatore principale di *Restaurant Simulator*. In questa interfaccia, il *manager* per effettuare la simulazione deve selezionare dal calendario, generato dal click sul pulsante presente nel campo *Seleziona la data di simulazione*, il giorno per cui intende effettuare la simulazione del completamento degli ordini, e dal campo *Seleziona un orario di simulazione* una delle ore indicate dal sistema, che permettono di effettuare il completamento di tutti gli ordini antecedenti a quella data e a quell'ora. In alternativa, il *manager* può spuntare l'opzione *Tutto il giorno* per considerare la simulazione dell'intera giornata e delle date ad essa precedenti. Effettuate queste scelte, il *manager* deve cliccare sul pulsante *Simula* affinché il sistema provveda al completamento di tutti gli ordini interessati.

In qualsiasi momento il *manager* può ritornare alla *Home Manager* cliccando sul pulsante *Back* in alto a sinistra.

### 2.1.2 Modalità Cliente

La Modalità Cliente rappresenta quella modalità in cui l'utente si immedesima nella figura di *Cliente* del ristorante creato in *Modalità Manager* per poter effettuare le seguenti operazioni:

- Prenotazione di un tavolo con rispettiva ordinazione e simulazione del pagamento;
- Visualizzazione degli ordini effettuati;
- Rilascio dei feedback per ordini completati.

Questa modalità è accessibile previa configurazione del ristorante effettuabile nella *Modalità Manager*.



## Autenticazione

La fase di autenticazione, come mostrato nella Figura 2.13, rappresenta il punto d'ingresso della *Modalità Cliente*. Qui l'utente può effettuare il login (se già cliente del ristorante) o la registrazione (se potenziale cliente del ristorante) e visualizzare, se presenti, la media dei feedback del ristorante suddivisi per categoria (Servizio, Rapporto qualità/prezzo, Ambiente) rilasciati dai clienti fino a quel momento.

The screenshot shows a web application window titled "Restaurant Simulator: Autenticazione". The interface is dark-themed. At the top, there's a login section with "Email" and "Password" input fields and a green "Login" button. Below the login section, on the left, is a section titled "Il mio ristorante" with a chef icon. Underneath, a bar chart titled "Il giudizio dei nostri clienti" displays three categories: "Servizio" with a score of 8.0, "Rapporto qualità/prezzo" with a score of 6.0, and "Ambiente" with a score of 10.0. On the right side, there's a "Crea un account" section with input fields for "Username", "Email", "Password", and "Conferma password", and a "Crea un account" button at the bottom.

Figura 2.13: Interfaccia *Autenticazione*.

Se l'utente è un potenziale cliente, può registrarsi compilando i seguenti form e premendo infine sul pulsante *Crea un account*.

- *Username*: identificativo dell'utente la cui lunghezza deve essere compresa tra 4 e 16 caratteri e il cui primo di questi non inizi con una cifra;
- *Email*: l'indirizzo di posta elettronica dell'utente;
- *Password*: la password di accesso al ristorante la cui lunghezza deve essere maggiore di 6 caratteri;
- *Conferma password*: la stessa password inserita nel campo *Password*.

Verificata la validità dei campi inseriti dall'utente, quest'ultimo riceve un messaggio di avvenuta registrazione e viene inserito nella lista dei clienti del ristorante. Viceversa, l'utente viene notificato del mancato successo della registrazione se:

- I campi inseriti non risultano essere validi;
- L'username è già presente nel sistema;
- L'email è già presente nel sistema.

Se l'utente è già un cliente del ristorante, per accedere alla *Home Cliente* deve inserire la propria email e password negli appositi campi in alto a destra e cliccare sul pulsante *Login*. Il sistema verificherà, dunque, la loro corrispondenza nel database e indirizzerà il cliente nella *Home Cliente*, se confermata, altrimenti notificherà l'utente con un messaggio di errore.

## Home Cliente

La *Home Cliente* rappresenta l'interfaccia principale della *Modalità Cliente*. In questa fase del software il *cliente* può:

- Visualizzare un messaggio di benvenuto con il suo username;
- Effettuare una prenotazione accendendo nell'apposita interfaccia tramite il click sul pulsante *Ordinazione*;
- Visualizzare la lista dei propri ordini, il relativo dettaglio con la ricevuta e rilasciare i feedback per gli ordini completati tramite il click sul pulsante *I miei ordini*.



Figura 2.14: Interfaccia *Home Cliente*.

## Prenotazione

Cliccando sul pulsante *Ordinazione* della *Home Cliente*, il software mette a disposizione un'interfaccia grafica che permette al cliente di effettuare una prenotazione<sup>1</sup>.

Questa interfaccia si suddivide nelle seguenti tre sezioni, ciascuna delle quali rappresenta uno step della prenotazione e il cui accesso dipende dal completamento del precedente:

1. Verifica disponibilità del tavolo;
2. Ordinazione;
3. Pagamento e conferma dell'ordine.

<sup>1</sup> Per convenzione, ogni prenotazione effettuata, ha un tempo limite di un'ora per la relativa consumazione.

Restaurant Simulator (MC): Prenotazione

←

Verifica disponibilità tavolo

Data:  
24 gen 2020

Numero di persone:  
3

Ora:  
12:00

Verifica disponibilità

Stato disponibilità  
È possibile effettuare la prenotazione per 3 persone  
in data: 24-01-2020 12:00

Ordinazione

Pagamento e conferma dell'ordine

Continua

Figura 2.15: Interfaccia *Prenotazione* che mostra il primo step di una prenotazione.

La Figura 2.15, mostra il primo step della prenotazione, ovvero la verifica della disponibilità di un tavolo per una specifica data e una certa ora.

Per effettuare questa verifica, il cliente deve:

- Selezionare dal calendario, generato dal click sul pulsante presente nel campo *Data*, il giorno per cui intende effettuare la verifica della disponibilità di un tavolo;
- Inserire il numero di persone per cui intende effettuare la prenotazione nel campo *Numero di persone*;
- Selezionare l'ora per la quale intende effettuare la consumazione.
- Cliccare sul pulsante *Verifica disponibilità*.

Il sistema verifica, quindi, che tutti i campi inseriti siano validi e in caso di esito positivo, mostra nell'area *Stato disponibilità tavoli*, un messaggio che indica la possibilità o meno di poter effettuare la prenotazione secondo i parametri digitati.

Cliccando sul pulsante *Continua* della relativa sezione, qualora il sistema verifichi la disponibilità di un tavolo, il cliente passa alla fase successiva della prenotazione (vedi Figura 2.16).

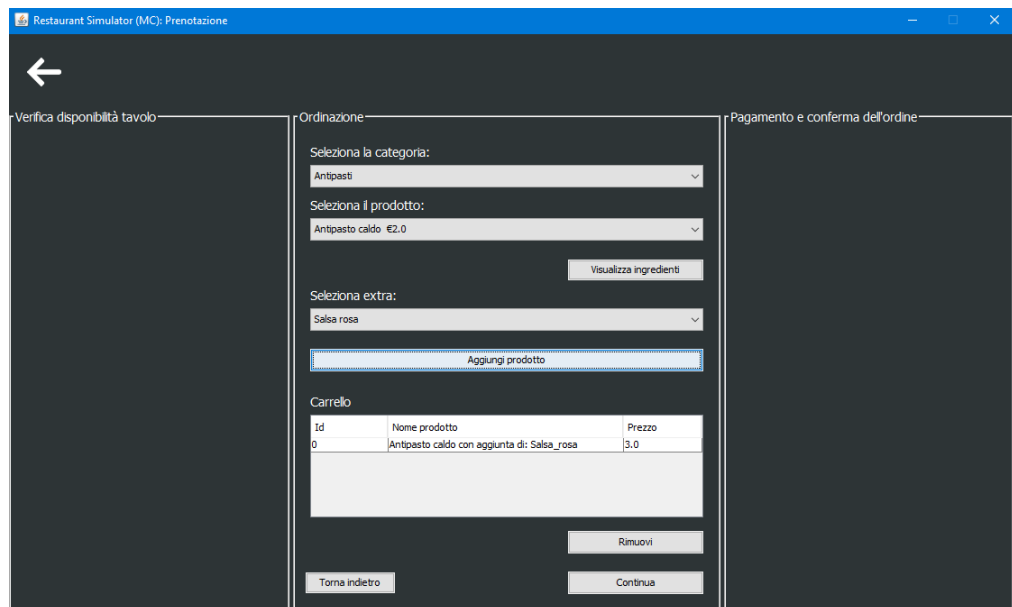


Figura 2.16: Interfaccia *Prenotazione* che mostra il secondo step di una prenotazione.

Nella sezione *Ordinazione*, il *cliente* effettua l'aggiunta dei prodotti con eventuali personalizzazioni nel *Carrello*.

Per aggiungere un prodotto deve:

- Selezionare una delle categorie proposte dal sistema nel campo *Seleziona la categoria*;
- Selezionare uno dei prodotti relativi alla categoria scelta nel campo *Seleziona il prodotto*. Inoltre, premendo sul pulsante *Visualizza ingredienti*, il software permette di visualizzare, se presenti, la lista degli ingredienti associati al prodotto;
- (Opzionale) Personalizzare il prodotto scelto con l'aggiunta di un ingrediente tra quelli proposti dal sistema, selezionandolo nel campo *Seleziona Extra*;
- Cliccare sul pulsante *Aggiungi prodotto*.

Il cliente può effettuare la rimozione di un prodotto aggiunto, selezionandolo dal carrello e cliccando sul tasto *Rimuovi*.

Il cliente può tornare in qualunque momento nella sezione precedente di *verifica disponibilità tavolo*, senza perdere eventuali prodotti aggiunti al carrello, cliccando sul pulsante *Torna indietro*.

In alternativa, se ha inserito almeno un prodotto nel carrello, può proseguire nella terza ed ultima fase di *Pagamento e conferma dell'ordine* cliccando sul tasto *Continua*. Il sistema prima di reindirizzare in quest'ultima fase, verifica se la data di prenotazione scelta è ancora valida. Questo perché il cliente potrebbe aver trascorso troppo tempo nella sezione di ordinazione e qualcun altro potrebbe aver prenotato per quella data al suo posto. In quest'ultimo caso il cliente viene avvertito con un apposito messaggio di conferma che chiede al cliente quale delle seguenti azioni intraprendere:

- Tornare nella sezione *Verifica disponibilità tavolo*, senza perdere i dati finora inseriti;
- Tornare alla *Home Cliente*.

Infine, nell'ultima sezione *Pagamento e conferma dell'ordine* (vedi Figura 2.17), il software mostra:

- Il riepilogo dell'ordine, contenete la data di prenotazione, la lista dei prodotti scelti con relativo prezzo e il totale da pagare per concludere e confermare la prenotazione.
- Una sezione che permette di effettuare il pagamento in contante. Con quest'ultimo si vuole simulare l'azione del pagamento in contante al gestore, una volta terminata la consumazione.

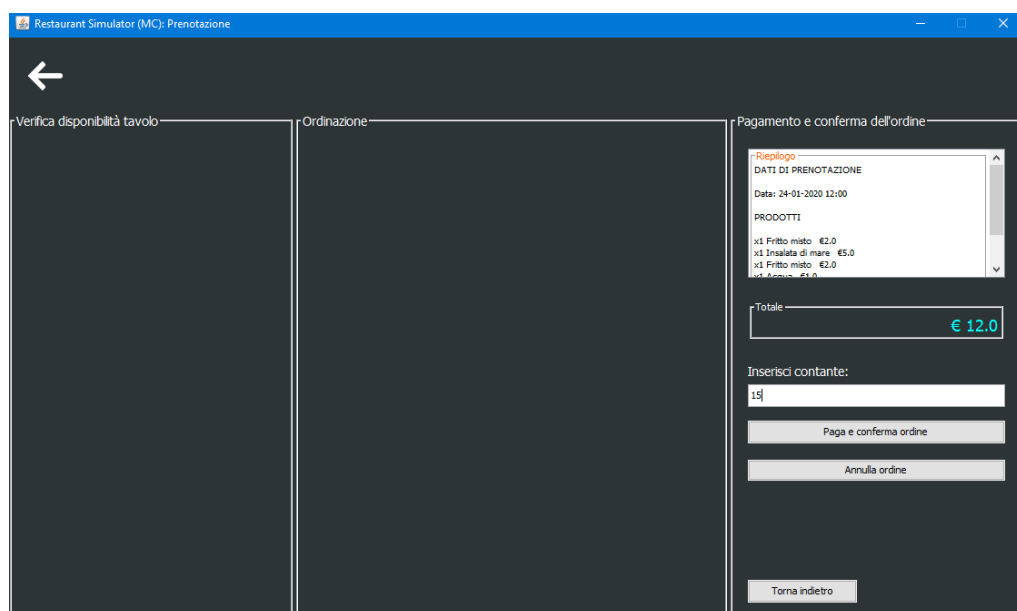


Figura 2.17: Interfaccia *Prenotazione* che mostra il terzo step di una prenotazione.

Per effettuare il pagamento, il cliente deve inserire nel campo *Inserisci contante*, l'importo in euro con cui intende pagare la somma dovuta, e cliccare sul tasto *Paga e conferma ordine*. Il sistema controlla, dapprima, che la data di prenotazione scelta sia ancora valida e, in caso di esito positivo, che la cifra inserita sia maggiore o uguale al totale richiesto. Se quest'ultima risulta valida per il sistema, esso notifica al cliente il successo della prenotazione con l'eventuale resto erogato (vedi Figura 2.18). In quest'ultimo caso il sistema dà possibilità al cliente di tornare alla Home o di tornare nell'interfaccia di selezione modalità del software.

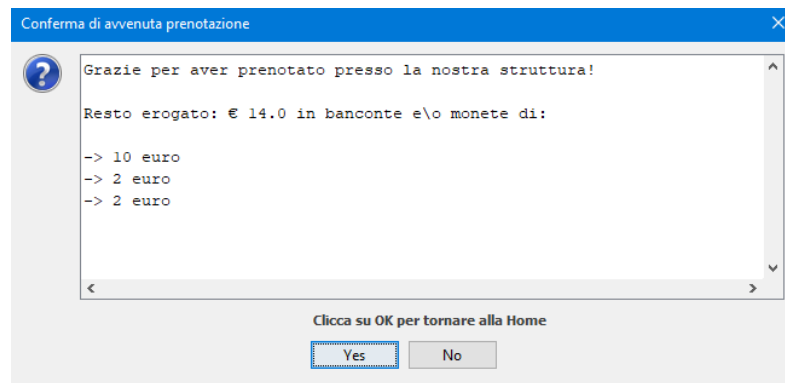


Figura 2.18: Messaggio di conferma prenotazione con visualizzazione del resto erogato.

In alternativa, il cliente può cliccare sul tasto *Torna indietro* per tornare nella sezione ordinazione, oppure cliccare sul tasto *Annulla ordine* per annullare la prenotazione e di conseguenza perdere tutti i dati inseriti fino a quel momento. Occorre notare che il cliente può effettuare l'annullamento dell'ordine in qualsiasi momento cliccando sul pulsante *Back* in alto a sinistra per ritornare alla *Home Cliente*.

## I miei ordini

Cliccando sul pulsante *I miei ordini* della *Home Cliente*, il software mette a disposizione un'interfaccia grafica (vedi Figura 2.19) che permette al *cliente* di:

- Visualizzare la lista dei suoi ordini e il relativo dettaglio.
- Visualizzare la ricevuta erogata dal *manager* con la possibilità di stamparla su carta o di effettuarne il salvataggio su file;
- Rilasciare un feedback per gli ordini completati.

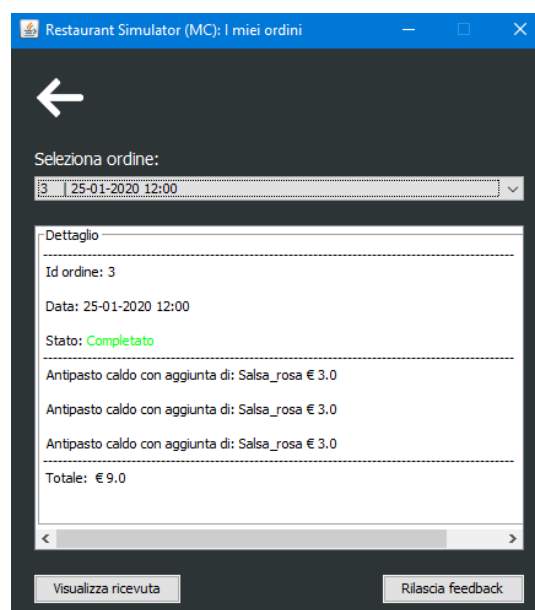


Figura 2.19: Interfaccia *I miei ordini*.

Per visualizzare il dettaglio di uno specifico ordine, il *cliente* deve semplicemente selezionarlo dal campo *Seleziona ordine*. Il sistema, dunque, produrrà le informazioni relative all'ordine selezionato nell'area *Dettaglio*.

Per visualizzare e accedere alle relative funzioni disponibili per la ricevuta, il *cliente* deve cliccare sul pulsante *Visualizza ricevuta*. La Figura 2.20 mostra l'interfaccia corrispondente.



Figura 2.20: Interfaccia *Ricevuta ordine*.

Il rilascio del feedback per un ordine richiede che lo stato dell'ordine sia impostato su completato. Questa informazione è anche visibile nel dettaglio dell'ordine selezionato. Un ordine completato è un ordine per cui il cliente abbia già effettuato la consumazione, che per convenzione, termina un'ora dopo l'ora di prenotazione. Il cliente, inoltre, può rilasciare un solo feedback per ordine.

In qualsiasi momento il *cliente* può ritornare alla *Home Manager* cliccando sul pulsante *Back* in alto a sinistra.

Per rilasciare il feedback, il cliente deve, dapprima, cliccare sul tasto *Rilascia feedback* presente in basso a destra dell'interfaccia *I miei ordini*. Se è possibile rilasciarlo, il sistema indirizza il cliente nell'interfaccia per il rilascio del feedback relativa a quell'ordine (vedi Figura 2.21). In quest'ultima, il cliente sceglie per ciascuna categoria di valutazione (Servizio, Rapporto qualità\prezzo e ambiente), il metro giudizio che intende attribuire ad ognuna di essa.

I criteri di valutazione per ogni categoria sono:

- Pessimo;
- Insufficiente;
- Sufficiente;
- Buono;
- Eccellente

A ciascuna valutazione corrisponde un punteggio che sarà utilizzato nel calcolo della media relativa a tutti i feedback rilasciati per quella categoria. La media di ciascuna categoria di valutazione è visibile nell'interfaccia di autenticazione del cliente.

Il rilascio effettivo del feedback avviene alla pressione del tasto *Rilascia* della medesima interfaccia.

Il *cliente* può ritornare all'interfaccia precedente cliccando sul pulsante *Back* in alto a sinistra.

The image shows a software window titled "Rilascia feedback". Inside, there is a back arrow in the top left corner. Below it, there are three sections for rating: "Servizio", "Rapporto Qualità\Prezzo", and "Ambiente". Each section has five radio buttons corresponding to the rating levels: "Pessimo", "Insufficiente", "Sufficiente", "Buono", and "Eccellente". At the bottom right, there is a button labeled "Rilascia".

Figura 2.21: Interfaccia *Rilascia feedback*.



## 2.2 Raccolta dei requisiti: casi d'uso

Il nostro lavoro di progettazione parte con la raccolta dei requisiti. In questa sezione proponiamo i diagrammi dei casi d'uso dei due attori principali di *Restaurant Simulator*: il *manager* e il *cliente*.

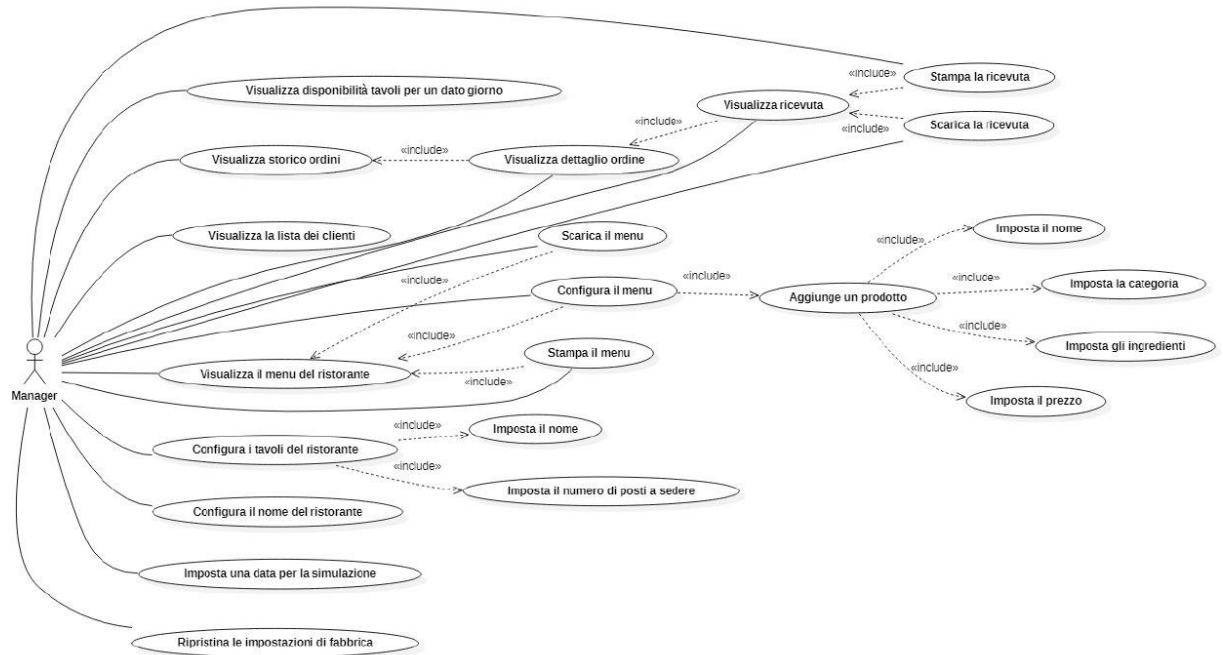


Figura 2.22: Diagramma dei casi d'uso dell'attore *manager*.

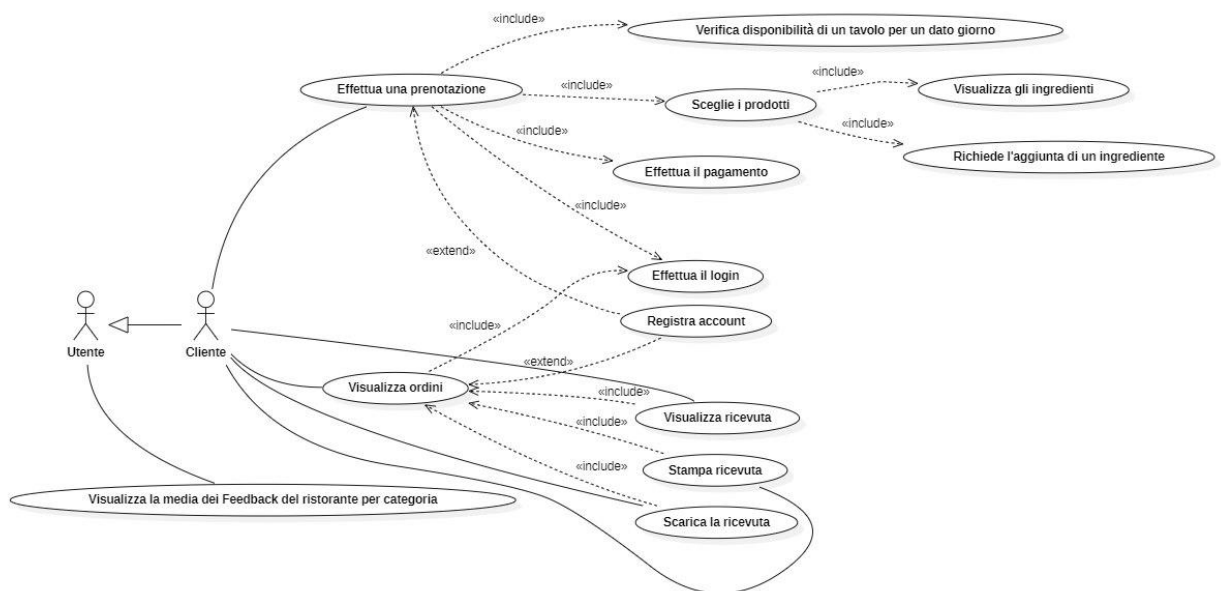


Figura 2.23: Diagramma dei casi d'uso dell'attore *cliente*.

### 2.3 Diagramma delle classi e motivazione delle scelte

In questa sezione presentiamo il diagramma delle classi che abbiamo progettato per lo sviluppo del nostro software, insieme ai design pattern utilizzati con le relative motivazioni.

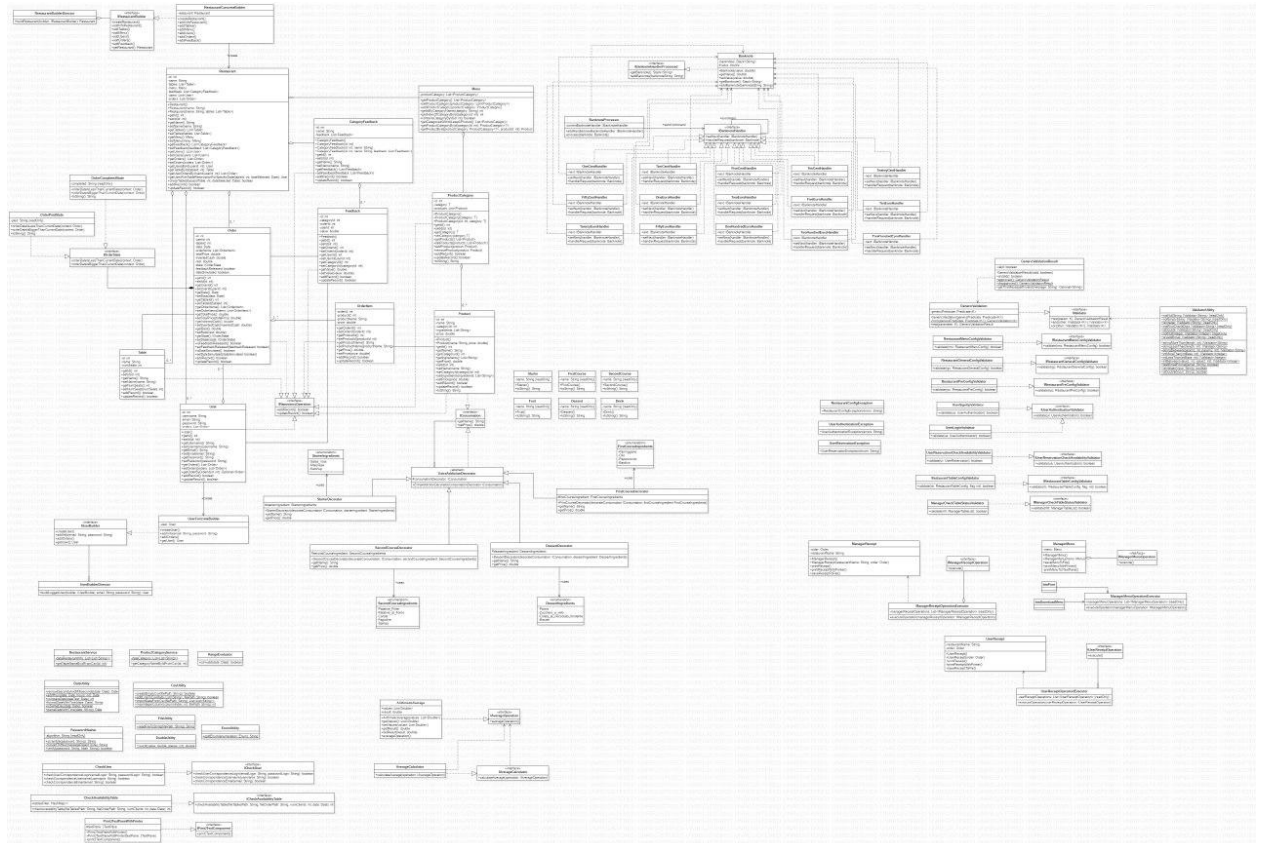


Figura 2.24: Diagramma delle classi di *Restaurant Simulator*.

La Figura 2.24 mostra il diagramma delle classi di *Restaurant Simulator*, realizzato con il software di progettazione *StarUML*.

I design pattern che abbiamo deciso di adottare, e che descriveremo nelle sottosezioni relative, sono i seguenti:

- *Pattern creazionali*: Builder;
- *Pattern comportamentali*: Chain of Responsibility, State, Command;
- *Pattern strutturali*: Decorator.

### 2.3.1 Builder

Il *Builder* è un design pattern di tipo *creazionale* usato per creare istanze di oggetti molto complessi. I suoi componenti sono:

- *Product*: definisce il tipo di oggetto complesso che sarà generato dal Builder;
- *Builder*: è l'interfaccia o classe astratta che dichiara le parti di cui è composto l'oggetto da costruire. Le implementazioni di ogni metodo sono fornite dalle sottoclassi concrete;
- *Concrete Builder*: concretizzazione del Builder. Contiene l'implementazione di tutti i metodi per la creazione di oggetti complessi;
- *Director*: classe che dirige la costruzione.

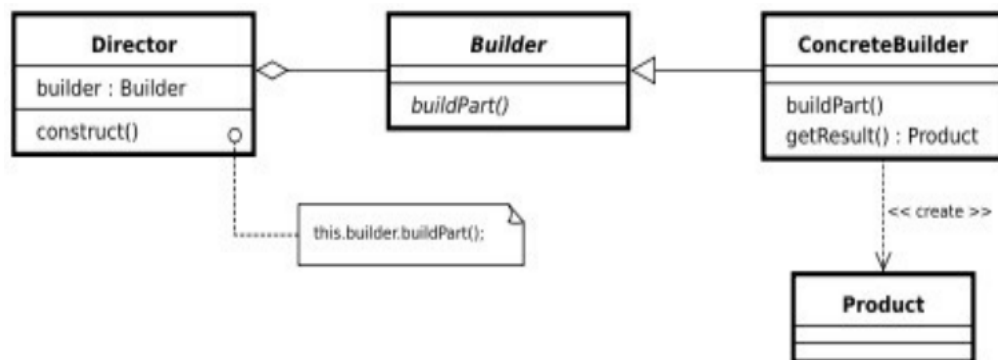


Figura 2.25: Diagramma delle classi del pattern builder.

Abbiamo optato per l'utilizzo di tale pattern creazionale per la creazione degli oggetti *Restaurant* e *User*, dal momento che sono entità complesse con molti attributi e metodi. Infatti, l'uso del *pattern builder* è consigliato nei casi in cui le classi presentano diversi attributi e pertanto, costruttori con molti parametri. Inoltre con l'applicazione di questo pattern abbiamo potuto mantenere distinte le parti costituenti degli oggetti in modo da consentire un migliore controllo del processo di costruzione dei singoli isolandoli dal resto del codice.

La Figura 2.26 mostra l'applicazione del pattern builder per la costruzione dell'oggetto *Restaurant*.

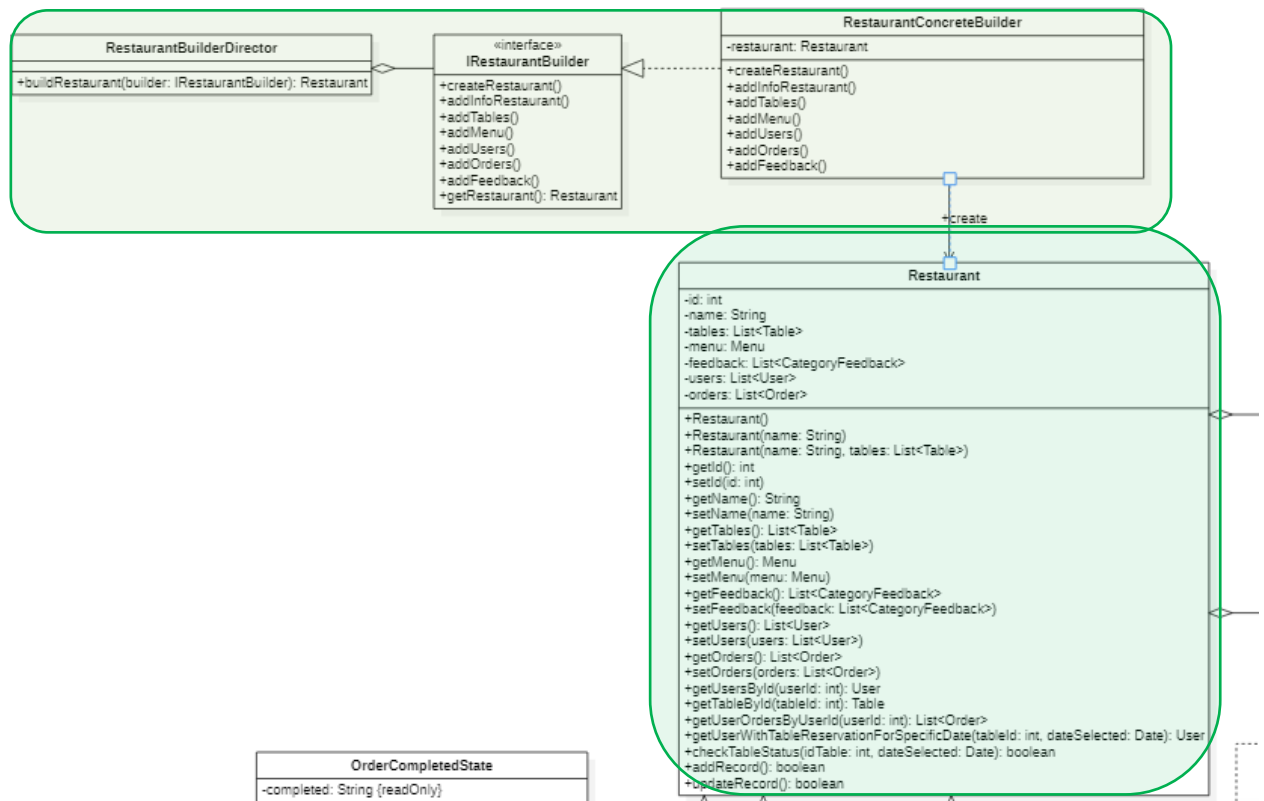


Figura 2.26: Diagramma delle classi del pattern builder per la costruzione dell'oggetto *Restaurant*.

Come possiamo osservare dalla Figura 2.26, le classi coinvolte nella costruzione dell'oggetto *Restaurant* sono:

- *Restaurant*: è il Product;
- *IRestaurantBuilder*: è il Builder;
- *RestaurantConcreteBuilder*: è il Concrete Builder;
- *RestaurantBuilderDirector*: è il Director.

La Figura 2.27 mostra l'applicazione del pattern builder per la costruzione dell'oggetto *User*.

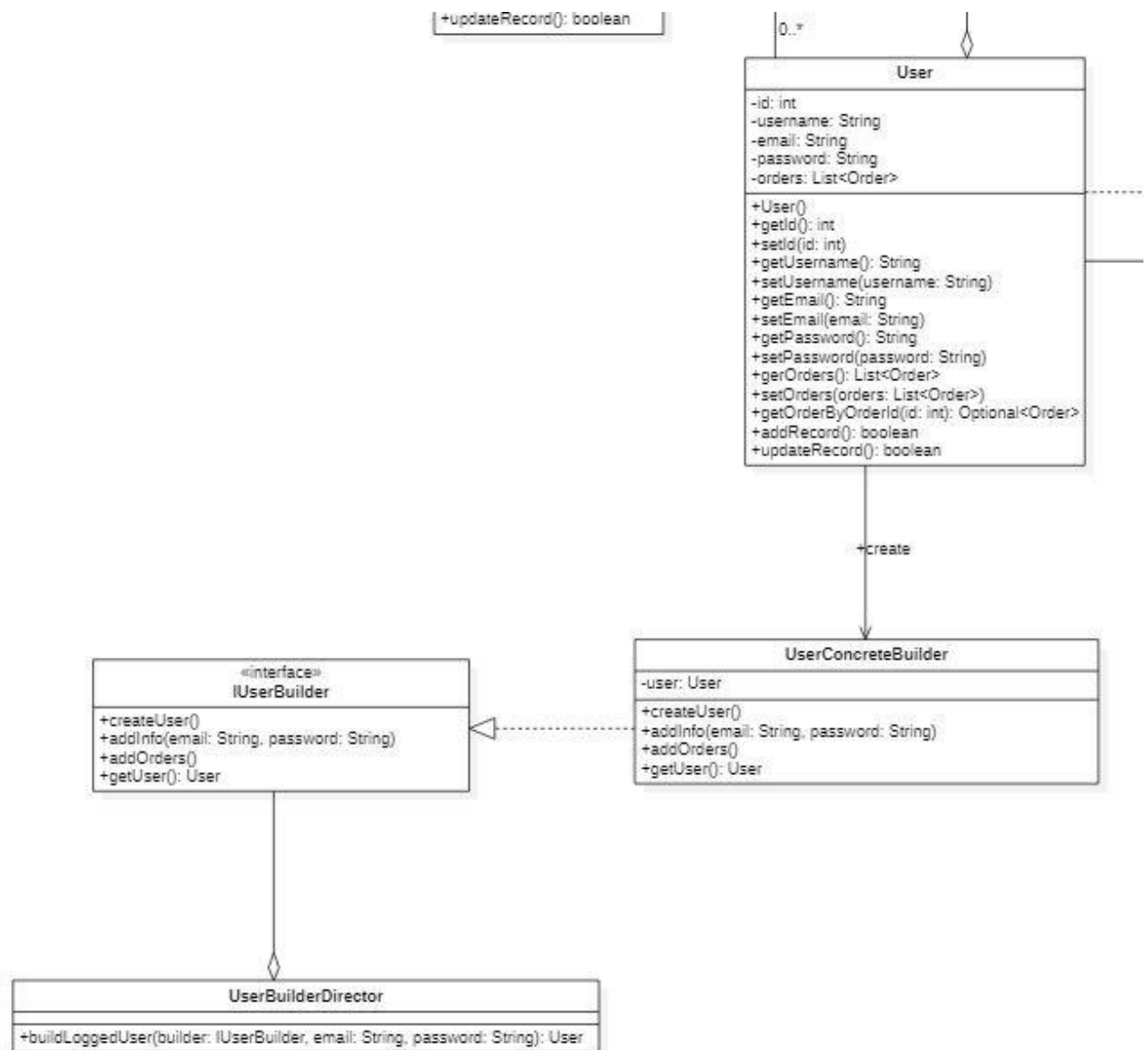


Figura 2.27: Diagramma delle classi del pattern builder per la costruzione dell'oggetto *User*.

Come possiamo osservare dalla Figura 2.27, le classi coinvolte nella costruzione dell'oggetto *User* sono:

- *User*: è il Product;
- *IUserBuilder*: è il Builder;
- *UserConcreteBuilder*: è il Concrete Builder;
- *UserBuilderDirector*: è il Director.

### 2.3.2 Chain of Responsibility

Il *Chain of Responsibility* è un design pattern di tipo *comportamentale* che si usa nelle situazioni in cui un client richiede un'operazione che varia in un insieme di operazioni possibili per lo stesso insieme di dati.

Seguendo il principio di responsabilità singola, ogni operazione sarà gestita da una specifica classe.

La richiesta del client viene gestita da un handler, che chiama a catena le classi responsabili delle singole operazioni, fino a trovare quella richiesta.

Questo pattern è composto dalle seguenti classi:

- *Client*: è responsabile dell'istanziatura della catena di handler e dell'invocazione del metodo *handleRequest* sul primo oggetto;
- *Handler*: definisce l'interfaccia per gestire le richieste e implementa il collegamento successivo alla gerarchia;
- *ConcreteHandler*: ognuna di queste classi implementa il metodo *handleRequest* e mantiene un riferimento al successivo elemento della catena. Verifica se è in grado di gestire la richiesta, altrimenti la passa all'elemento successivo nella catena.

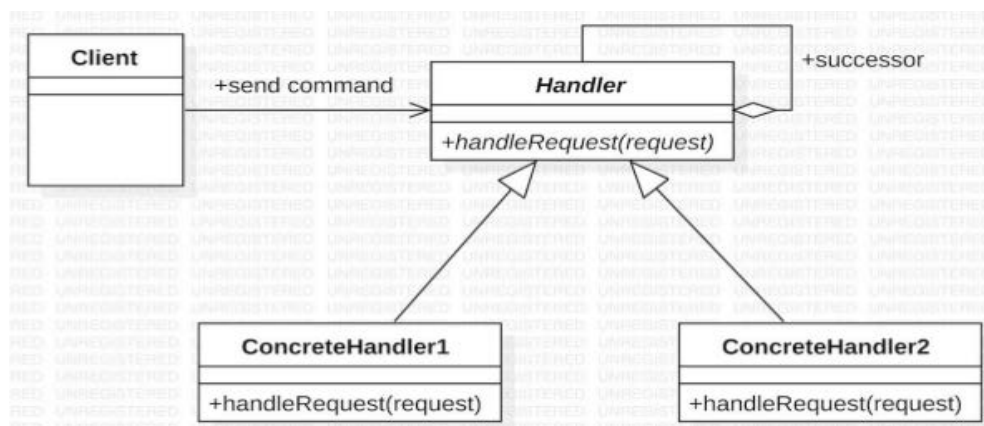


Figura 2.28: Diagramma delle classi del pattern chain of responsibility.

Abbiamo scelto di utilizzare questo pattern per la simulazione del pagamento in contante dell'ordine da parte del cliente. In particolare, grazie a questo pattern possiamo processare tutte le banconote e le monete richieste per erogare il resto al cliente. Ad esempio se il resto dovuto al cliente è pari ad un importo di euro venticinque, tutti gli handler coinvolti saranno quelli il cui valore della banconota sia minore o uguale a euro venti.

La Figura 2.29 mostra l'applicazione del pattern chain of responsibility per il processamento dell'eventuale resto da erogare al pagamento effettuato dal cliente per l'ordine.

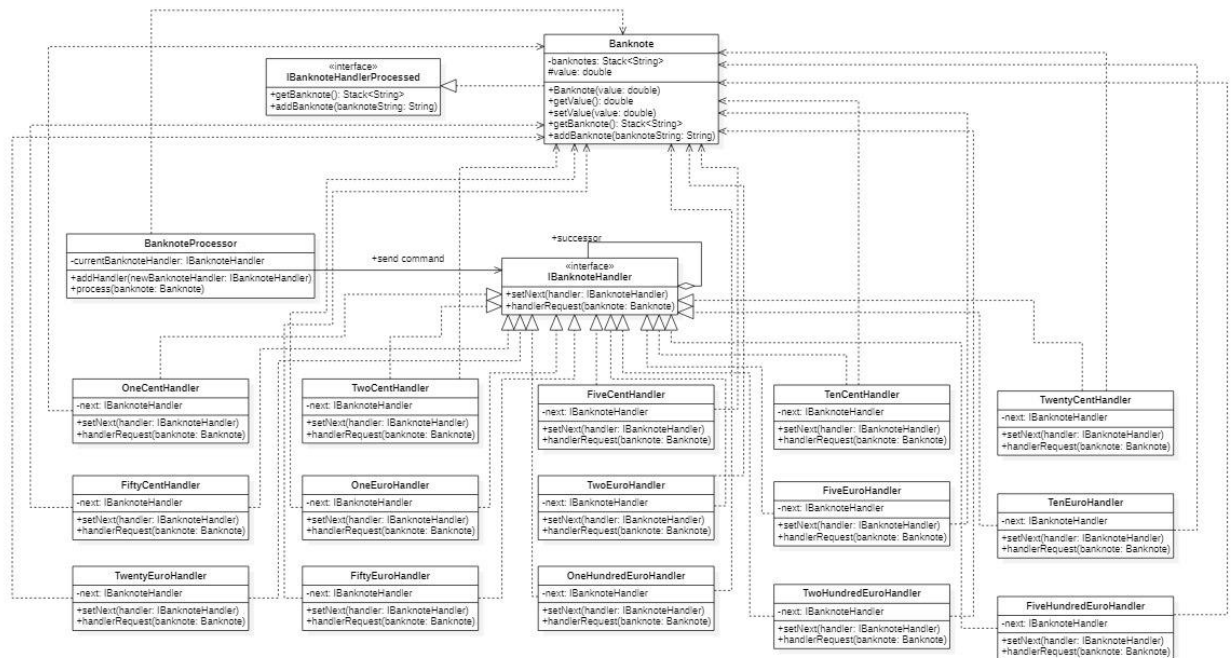


Figura 2.29: Diagramma delle classi del pattern chain of responsibility per il processamento del resto in *Restaurant Simulator*.

Come possiamo osservare dalla Figura 2.29, le classi coinvolte nell'applicazione del pattern chain of responsibility sono:

- *BanknoteProcessor*: è il Client;
- *IBanknoteHandler*: è l'Handler;
- *OneCentHandler*: è un ConcreteHandler;
- *TwoCentHandler*: è un ConcreteHandler;
- *FiveCentHandler*: è un ConcreteHandler;
- *TenCentHandler*: è un ConcreteHandler;
- *TwentyCentHandler*: è un ConcreteHandler;
- *FifthCentHandler*: è un ConcreteHandler;
- *OneEuroHandler*: è un ConcreteHandler;
- *TwoEuroHandler*: è un ConcreteHandler;
- *FiveEuroHandler*: è un ConcreteHandler;
- *TenEuroHandler*: è un ConcreteHandler;
- *TwentyEuroHandler*: è un ConcreteHandler;
- *FifthEuroHandler*: è un ConcreteHandler;
- *OneHundredEuroHandler*: è un ConcreteHandler;
- *TwoHundredEuroHandler*: è un ConcreteHandler;
- *FiveHundredEuroHandler*: è un ConcreteHandler;

### 2.3.3 State

*State* è un design pattern di tipo *comportamentale* che consente ad un oggetto di cambiare il proprio comportamento durante l'esecuzione del software in funzione dello stato in cui si trova. Le classi coinvolte sono le seguenti:

- *Context*: definisce la classe del client e mantiene una istanza della *ConcreteState* che definisce lo stato corrente;
- *State*: definisce un'interfaccia per incapsulare il comportamento associato ad un determinato stato;
- *ConcreteState*: implementa il comportamento associato ad un particolare stato.

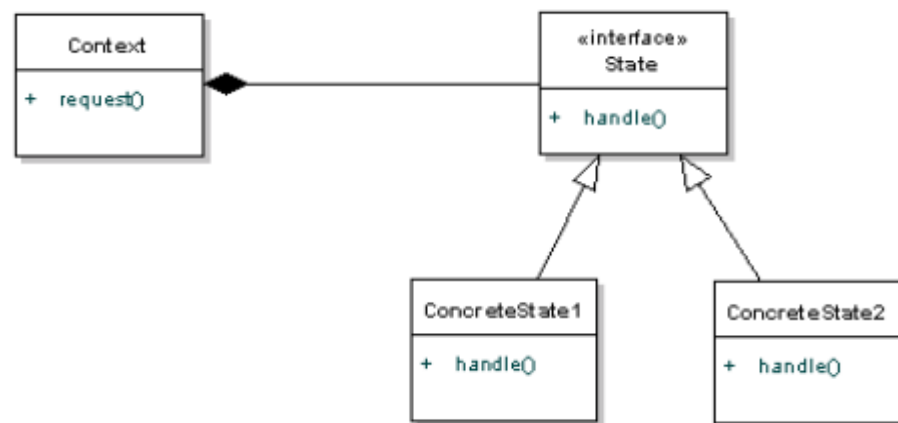


Figura 2.30: Diagramma delle classi del pattern state.

La scelta di questo pattern è dovuta alla gestione dello stato degli ordini effettuati dai clienti del ristorante.

In *Restaurant Simulator*, infatti ciascun ordine presenta due stati:

- *Pagato*: l'utente ha pagato l'ordine ma non ha ancora effettuato la consumazione;
- *Completato*: l'utente ha pagato l'ordine ed ha effettuato la consumazione.

Ricordiamo inoltre che, per convenzione, la consumazione ha una durata di un'ora a partire dalla data di prenotazione.

L'applicazione di questo pattern, quindi, ci ha permesso di cambiare a runtime lo stato degli ordini evitando l'uso eccessivo di istruzioni condizionali nella classe che ne implementa il comportamento.



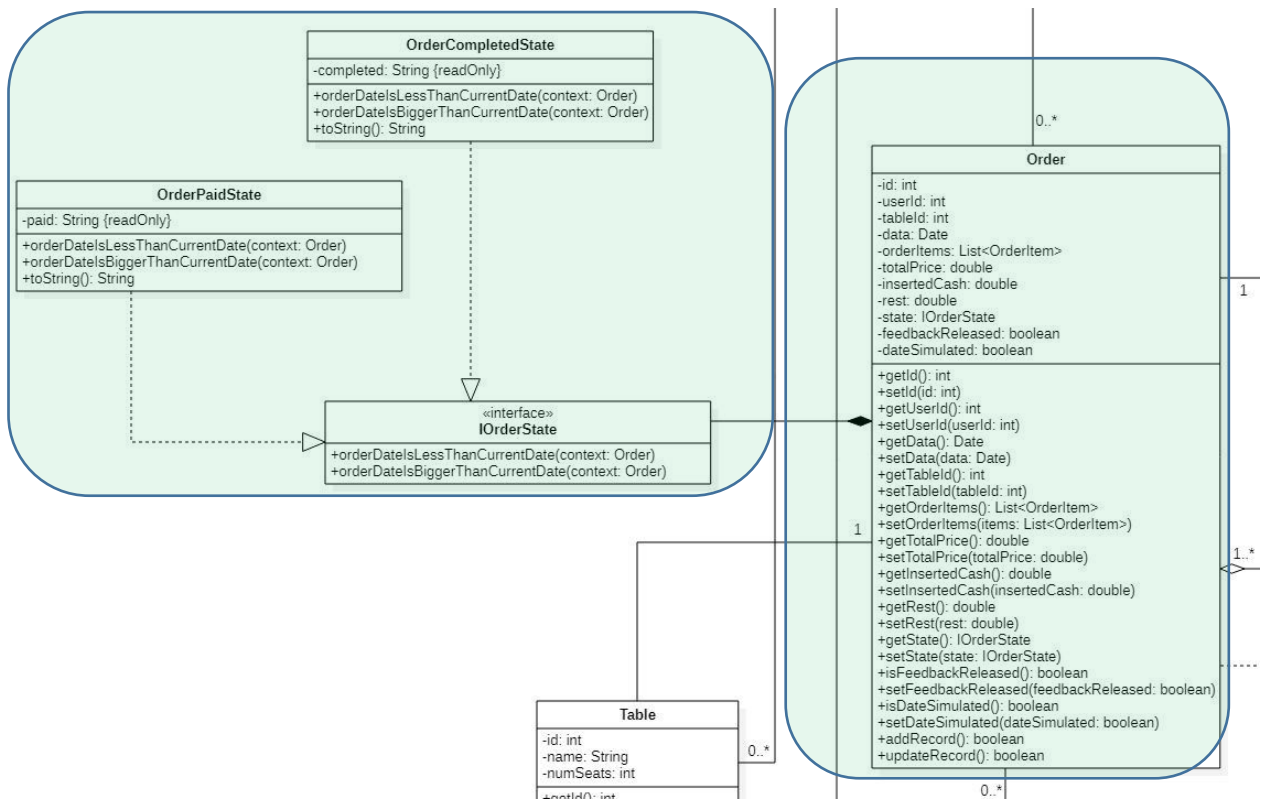


Figura 2.31: Diagramma delle classi del pattern state per gestire lo stato degli ordini in *Restaurant Simulator*.

Come possiamo osservare dalla Figura 2.31, le classi coinvolte nell'applicazione del pattern state sono:

- *Order*: è il Context;
- *IOrderState*: è lo State;
- *OrderPaidState*: è un ConcreteState;
- *OrderCompletedState*: è un ConcreteState.

### 2.3.4 Command

Il *Command* è un design pattern di tipo *comportamentale* che si realizza costruendo oggetti che rappresentano azioni o eventi, incapsulando eventuali parametri per l'esecuzione di un'operazione. Le classi coinvolte sono:

- *Client*: classe che richiede il comando ed imposta il Receiver;
- *Receiver*: classe responsabile dell'esecuzione dell'azione associata al comando;
- *Command*: astrazione del comando;
- *ConcreteCommand*: concretizzazione del comando. Esegue il comando delegando al ricevitore;
- *Invoker*: classe che lancia il comando.

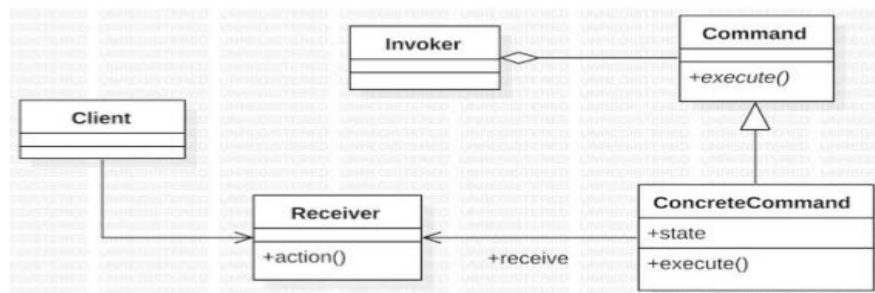


Figura 2.31: Diagramma delle classi del pattern command.

La scelta del pattern command è nata dall'esigenza di avere un design appropriato per la gestione dei comandi relativi alla stampa e al salvataggio del menu del ristorante e delle ricevute degli ordini. Nella prima fase di sviluppo, ci siamo basati sulla classica definizione del pattern, in un secondo momento abbiamo rifattorizzato il tutto applicando delle tecniche di programmazione avanzata fornite a partire dalla versione 8 del linguaggio Java, ottenendo così un codice più pulito e snello.

La Figura 2.32 mostra la progettazione del design pattern command prima della rifattorizzazione, in relazione alla stampa e al salvataggio su file del menu del ristorante.

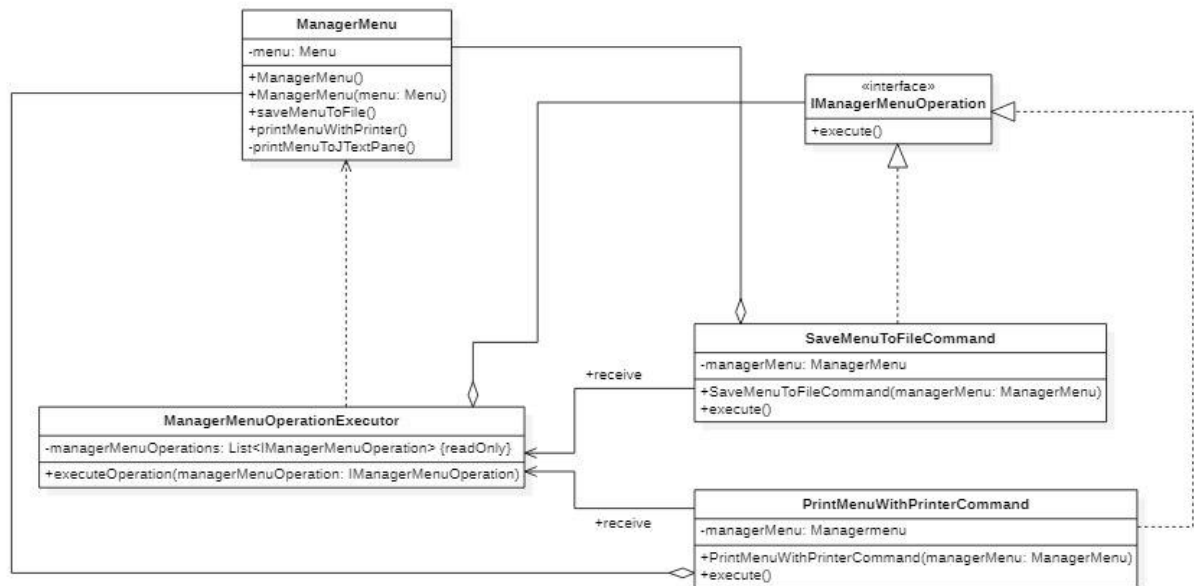


Figura 2.32: Diagramma delle classi del pattern command in relazione alle operazioni del menu prima della rifattorizzazione.

Come possiamo osservare dalla Figura 2.32, nella progettazione del pattern command in relazione ai comandi del menu del ristorante, le classi coinvolte prima della rifattorizzazione erano:

- *ManagerMenuOperationExecutor*: il Receiver;
- *IManagerMenuOperation*: il Command;
- *SaveMenuToFileCommand*: un ConcreteCommand;
- *PrintMenuWithPrinterCommand*: un ConcreteCommand;
- *ManagerMenu*: l'Invoker.

La Figura 2.33 mostra il diagramma delle classi rifattorizzato che rappresenta l'applicazione del pattern command per la gestione dei comandi relativi alla stampa e al salvataggio del menu del ristorante e delle ricevute degli ordini.

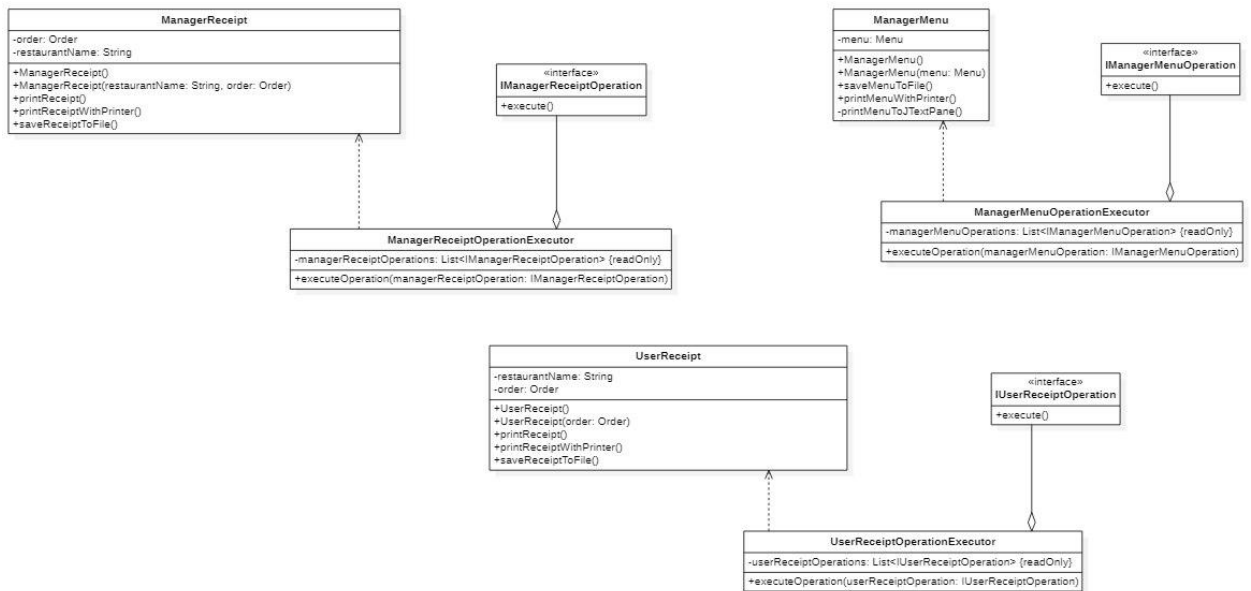


Figura 2.33: Diagramma delle classi del pattern command in *Restaurant Simulator*.

Come possiamo notare dalla Figura 2.33, grazie all'uso delle tecniche avanzate del linguaggio Java 8 è stato possibile rimuovere le classi relative ai *Concrete Command*.

### 2.3.5 Decorator

Il *Decorator* è un design pattern di tipo *strutturale* che permette di modificare, durante l'esecuzione del programma, il comportamento di un oggetto senza l'utilizzo dell'ereditarietà. La sua struttura è la seguente:

- *Component*: è l'interfaccia dell'oggetto a cui verranno aggiunte nuove funzionalità;
- *ConcreteComponent*: definisce un oggetto al quale verrà aggiunta una nuova funzionalità;
- *Decorator*: rappresenta l'interfaccia tra il Component e i ConcreteDecorator; possiede un riferimento al Component e un'interfaccia a esso conforme;
- *ConcreteDecorator*: rappresentano gli oggetti che aggiungono le nuove funzionalità ai ConcreteComponent.

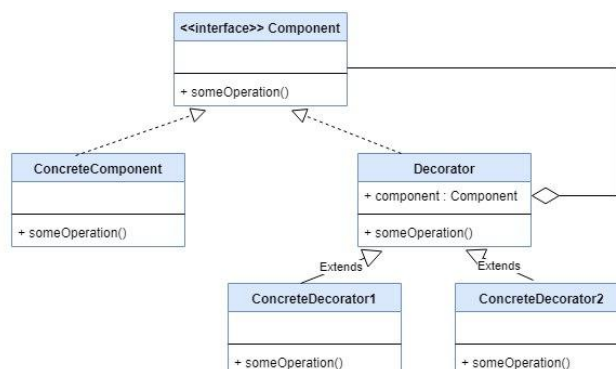


Figura 2.34: Diagramma delle classi del pattern decorator.

La motivazione che ci ha spinti verso l'utilizzo di questo pattern è stata dettata dall'esigenza di poter consentire al cliente, l'aggiunzione a run-time, di un ingrediente extra al prodotto che sta ordinando senza la necessità di doverci avvalere dell'uso dell'ereditarietà singola o multipla.

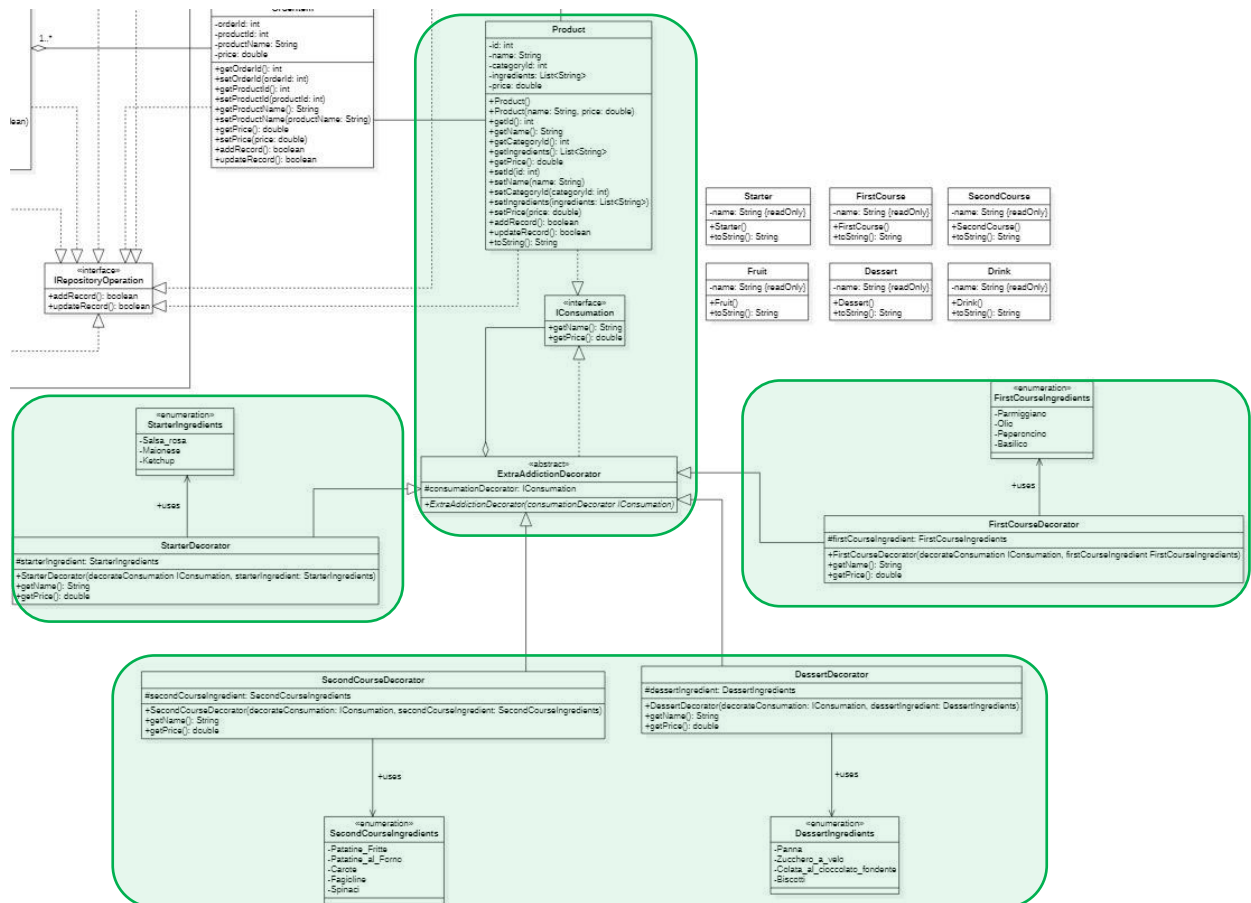


Figura 2.35: Diagramma delle classi del pattern decorator in *Restaurant Simulator*.

Come possiamo osservare dalla Figura 2.35, le classi coinvolte nell'applicazione del pattern decorator in *Restaurant Simulator* sono le seguenti:

- *IConsumation*: è il Component;
- *Product*: è il ConcreteComponent;
- *ExtraAddictionDecorator*: è il Decorator;
- *StarterDecorator*: è un ConcreteDecorator;
- *FirstCourseDecorator*: è un ConcreteDecorator;
- *SecondCourseDecorator*: è un ConcreteDecorator;
- *DessertDecorator*: è un ConcreteDecorator.

### 3. Conclusione

Con *Restaurant Simulator*, siamo riusciti a creare un software di simulazione capace di emulare due tra le principali figure di un ristorante: il *gestore* (qui anche chiamato *manager*) e il *cliente*. In questo software, infatti, l'utente ha la possibilità di immedesimarsi nei panni del proprietario di un ristorante, effettuando tutte quelle operazioni che vanno dalla configurazione alla gestione dello stesso, o nei panni di un suo cliente, per effettuare delle prenotazioni, gestire i propri ordini e rilasciare dei feedback per ciascuno di questi.

L'utilizzo dei design pattern e delle tecniche avanzate del linguaggio Java 8 ci ha permesso di produrre un software più compatto e funzionale. Questo agevolerà di molto una sua rifattorizzazione o manutenzione futura, concetti che al giorno d'oggi possono definirsi fondamentali nella progettazione e realizzazione di un software.

Realizzare un simulatore di ristorante ci ha dato numerose idee che potrebbero arricchire il software nel rilascio di possibili aggiornamenti. Dare la possibilità di creare diversi ristoranti nella *Modalità Manager* e di gestirne le entrate economiche sono solo alcuni esempi di come il software possa essere ampliato in futuro.

#### 3.1 Riferimenti e strumenti di sviluppo utilizzati

Per la progettazione e l'implementazione del nostro software abbiamo utilizzato i seguenti strumenti:

- StarUML per la progettazione del diagramma delle classi e dei casi d'uso;
- Apache Netbeans come ide di sviluppo;
- Swing come framework per le interfacce grafiche;
- JCalendar come framework per la visualizzazione del calendario. Quest'ultimo è disponibile al seguente link: <https://toedter.com/jcalendar/>;
- Google drive come strumento di condivisione del materiale di sviluppo.