
Ingegneria del Software
2019/2020



Brivio Andrea
Frighi Riccardo Maria

Gruppo – Birra – 4

SOMMARIO

PARTE 1: ANALISI	3
Diagramma dei casi d'uso	3
Casi d'uso	4
Diagramma di dominio	10
Diagramma di stato: Ingrediente	10
Diagramma di attività: Crea lotto	11
PARTE 2: PROGETTAZIONE	12
Diagramma di architettura software	12
Diagrammi classi di progetto	13
Diagramma classi di progetto: GUI	13
Diagramma classi di progetto: Gestori	14
Diagramma classi di progetto: Errori	14
Diagramma classi di progetto: Dominio	15
Diagramma classi di progetto: Dominio + Gestori	16
Diagramma classi di progetto: Database	17
Diagramma classi di progetto: Utility	17
Diagrammi di sequenza	18
Diagramma di sequenza: UC2 – Crea ricetta	18
Diagramma di sequenza: UC4 – Crea lotto	19
Diagramma di sequenza: UC9 – Aggiungi ingrediente al catalogo	20
Diagramma di sequenza: UC12 – Aggiungi ingrediente a lista spesa	21
Diagramma di sequenza: UC13 – Acquista ingrediente	22
Pattern utilizzati	23
Design Principles	23
Principi Phame	23
Persistenza	23
PARTE 3: ANALISI TECNICA	24
Antipattern	24
Analis Understand	25
Analisi SonarQube	27
ISTRUZIONI PER L'USO	28

PARTE I : ANALISI

DIAGRAMMA DEI CASI D'USO



CASO D'USO FORMATO INFORMALE UC1: Visualizza ricettario

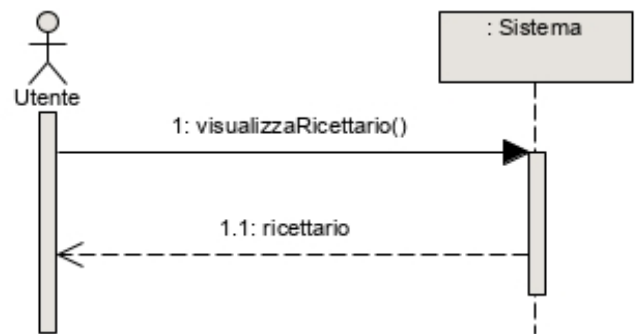
LIVELLO: Sottofunzione

SCENARIO PRINCIPALE:

1. L'utente accede al ricettario
2. Il sistema restituisce la lista delle ricette

SCENARI ALTERNATIVI:

- 1a. Nel sistema non è stata ancora inserita alcuna ricetta
 1. Il sistema restituisce all'utente un ricettario vuoto



CASO D'USO FORMATO INFORMALE UC2: Crea ricetta

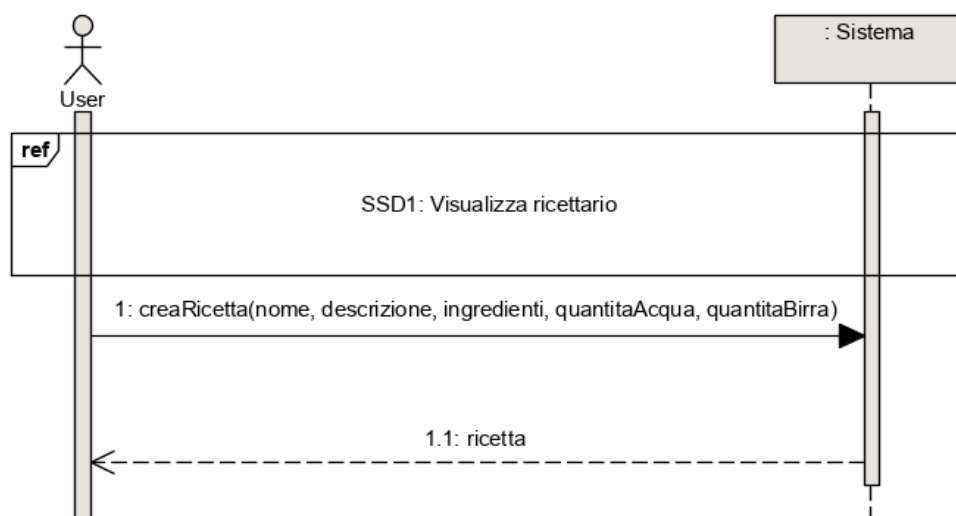
LIVELLO: Obiettivo Utente

SCENARIO PRINCIPALE:

1. L'utente accede al ricettario (rif. Caso d'Uso UC1 "Visualizza ricettario")
2. L'utente crea una nuova ricetta
3. L'utente inserisce nome della ricetta
4. (OPZIONALE) L'utente inserisce una breve descrizione della ricetta
5. L'utente inserisce i vari ingredienti specificandone nome, quantità e la categoria di appartenenza
6. (OPZIONALE) L'utente inserisce la quantità di acqua da usare
7. L'utente inserisce la quantità di birra che si può produrre utilizzando i quantitativi degli ingredienti indicati nella ricetta
8. L'utente conferma i dati della nuova ricetta
9. Il sistema controlla la correttezza dei valori inseriti
10. Il sistema converte le quantità dei vari ingredienti in valori assoluti
11. Il sistema salva la nuova ricetta nel ricettario

SCENARI ALTERNATIVI:

- 9a. Nel sistema è già presente una ricetta con lo stesso nome
 1. Il sistema invita l'utente ad inserire un nome diverso
- 9b. I valori inseriti non sono ammissibili
 1. Il sistema invita l'utente ad inserire dei valori ammissibili



CASO D'USO FORMATO INFORMALE UC3: Gestisci ricetta

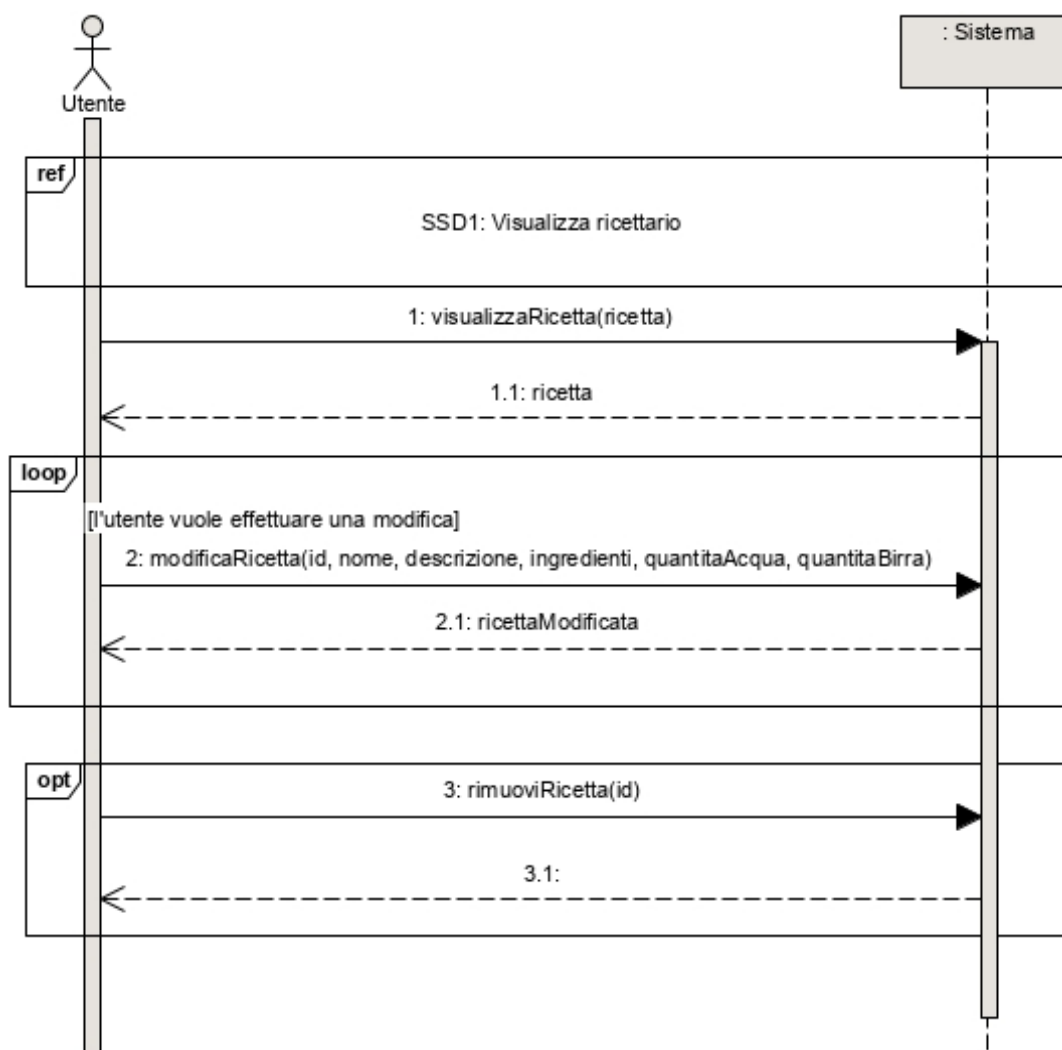
LIVELLO: Obiettivo Utente

SCENARIO PRINCIPALE:

1. L'utente accede al ricettario (rif. Caso d'Uso UC1 "Visualizza ricettario")
2. L'utente seleziona la ricetta di suo interesse
3. L'utente visualizza la ricetta interessata (nome, descrizione, ingredienti e passaggi)
4. (OPZIONALE) L'utente decide di modificare la ricetta
 - 4.1. L'utente effettua le modifiche
 - 4.1.1. L'utente modifica il nome della ricetta
 - 4.1.2. L'utente modifica la descrizione della ricetta
 - 4.1.3. L'utente modifica gli ingredienti della ricetta
 - 4.1.3.1. Il sistema converte le quantità modificate dei vari ingredienti in valori assoluti
 - 4.1.4. L'utente modifica i passaggi della ricetta
 - 4.1.5. L'utente modifica la quantità di birra che si può produrre utilizzando i quantitativi degli ingredienti indicati nella ricetta
L'utente può decidere di fare anche solo una delle quattro modifiche di cui sopra
 - 4.2. L'utente conferma le modifiche effettuate
 - 4.3. Il sistema controlla la correttezza dei dati inseriti
 - 4.4. Il sistema salva i dati della ricetta
L'utente ripete il passo 4 ogni volta che vuole modificare la ricetta
5. (OPZIONALE) L'utente decide di eliminare la ricetta
 - 5.1. L'utente elimina la ricetta
 - 5.2. Il sistema elimina la ricetta dal ricettario

SCENARI ALTERNATIVI:

- 4.3a. I valori inseriti non sono ammissibili
 1. Il sistema invita l'utente ad inserire dei valori ammissibili



CASO D'USO FORMATO INFORMALE UC4: Crea lotto

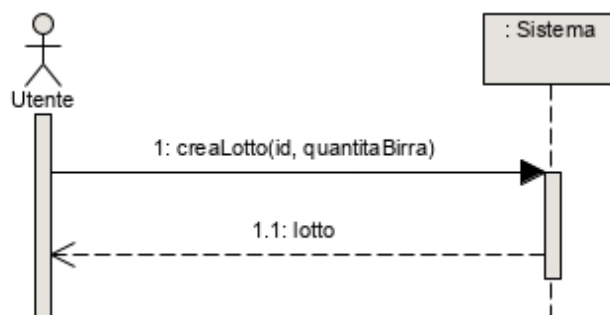
LIVELLO: Obiettivo Utente

SCENARIO PRINCIPALE:

1. L'utente seleziona una ricetta nella lista di quelle disponibili
2. L'utente inserisce la quantità di birra che intende produrre
3. L'utente crea un nuovo lotto
4. Il sistema controlla la correttezza del valore inserito
5. Il sistema inserisce il lotto nella lista dei lotti precedenti
6. Il sistema aggiorna la disponibilità ingredienti
7. Il sistema verifica disponibilità ingredienti per successive produzioni dei lotti della ricetta selezionata

SCENARI ALTERNATIVI:

- 1a. Non è presente alcuna ricetta nel ricettario
 1. Il sistema non è in grado di proseguire con la creazione
- 2a. Il valore inserito non è ammissibile
 1. Il sistema invita l'utente a inserire un valore ammissibile
- 5a. L'ingrediente non è presente o è in quantità insufficiente per la produzione corrente
 1. Il sistema notifica l'utente dell'assenza degli ingredienti necessari per la produzione corrente
- 5b. Le quantità di ingredienti presenti non sono sufficienti per la produzione successiva
 1. Il sistema notifica l'utente dell'indisponibilità degli ingredienti necessari per la produzione successiva



CASO D'USO FORMATO INFORMALE UC5: Gestisci lotto

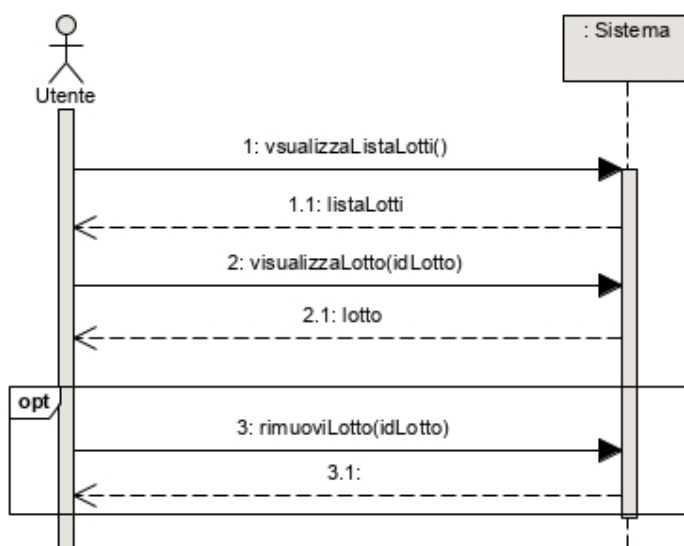
LIVELLO: Obiettivo Utente

SCENARIO PRINCIPALE:

1. L'utente visualizza l'elenco dei lotti precedenti
2. L'utente seleziona il lotto desiderato
3. (OPZIONALE) L'utente decide di eliminare il lotto
 - 3.1. L'utente elimina il lotto
 - 3.2. Il sistema elimina il lotto dalla lista dei lotti precedenti

SCENARI ALTERNATIVI:

- 1a. Non è presente alcun lotto nella lista dei lotti precedenti
 1. Il sistema informa l'utente sulla mancanza di lotti nella lista dei lotti precedenti



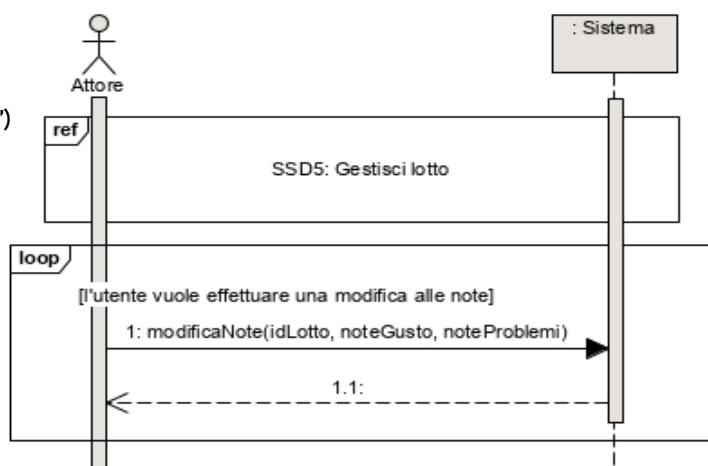
CASO D'USO FORMATO INFORMALE UC6: Gestisci note lotto

LIVELLO: Obiettivo Utente

SCENARIO PRINCIPALE:

1. L'utente visualizza il lotto (rif. Caso d'uso UC5 "Gestisci lotto")
2. L'utente decide di modificare una nota
 - 2.1. L'utente modifica il contenuto della nota
 - 2.2. L'utente conferma le modifiche effettuate
 - 2.3. Il sistema salva i dati del lotto

L'utente ripete il passo 2 ogni volta che vuole modificare una nota (dei problemi o del gusto)



CASO D'USO FORMATO INFORMALE UC7: Visualizza catalogo ingredienti

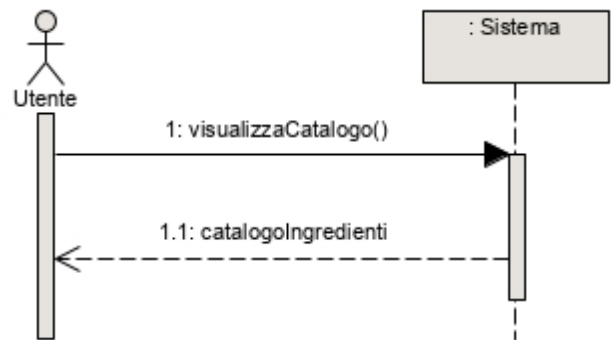
LIVELLO: Sottofunzione

SCENARIO PRINCIPALE:

1. L'utente accede al catalogo ingredienti
2. Il sistema restituisce la lista degli ingredienti con relativi quantitativi

SCENARI ALTERNATIVI:

- 1a. Nel sistema non è stato ancora inserito alcun ingrediente
1. Il sistema restituisce all'utente un catalogo vuoto



CASO D'USO FORMATO INFORMALE UC8: Gestisci catalogo ingredienti

LIVELLO: Obiettivo Utente

SCENARIO PRINCIPALE:

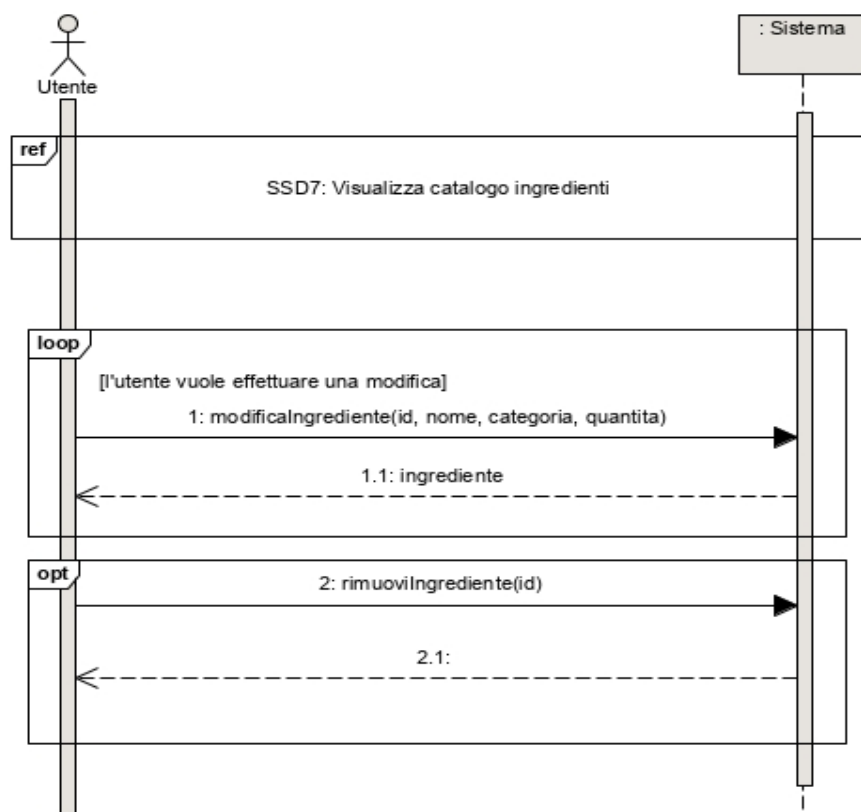
1. L'utente accede al catalogo ingredienti (rif. Caso d'Uso UC7 "Visualizza catalogo ingredienti")
2. (OPZIONALE) L'utente decide di modificare un ingrediente
 - 2.1. L'utente effettua le modifiche
 - 2.1.1. L'utente modifica il nome dell'ingrediente
 - 2.1.2. L'utente modifica la categoria dell'ingrediente
 - 2.1.3. L'utente modifica il quantitativo disponibile*L'utente può decidere di fare anche solo una delle modifiche di cui sopra*
 - 2.2. L'utente conferma le modifiche effettuate
 - 2.3. Il sistema controlla la correttezza dei valori inseriti
 - 2.4. Il sistema salva i dati dell'ingrediente

L'utente ripete il passo 2 ogni volta che vuole modificare un ingrediente

3. (OPZIONALE) L'utente decide di eliminare l'ingrediente
 - 3.1. L'utente elimina l'ingrediente
 - 3.2. Il sistema elimina l'ingrediente dal catalogo

SCENARI ALTERNATIVI:

- 2.3a. I valori inseriti non sono ammissibili
 1. Il sistema invita l'utente ad inserire dei valori ammissibili



CASO D'USO FORMATO INFORMALE UC9: Aggiungi ingrediente al catalogo ingredienti

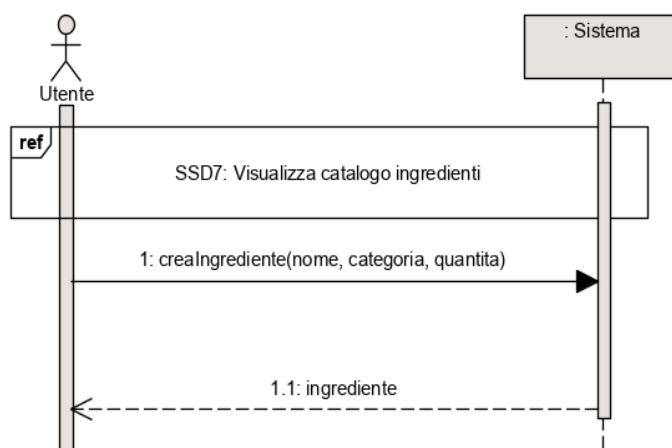
LIVELLO: Obiettivo Utente

SCENARIO PRINCIPALE:

1. L'utente accede al catalogo ingredienti (rif. Caso d'Uso UC7 "Visualizza catalogo ingredienti")
2. L'utente inserisce i dati del nuovo ingrediente specificandone nome, categoria e quantità
3. L'utente aggiunge un nuovo ingrediente
4. Il sistema controlla la correttezza dei dati inseriti
5. Il sistema aggiunge l'ingrediente al catalogo

SCENARI ALTERNATIVI:

- 4a. I valori inseriti non sono ammissibili
 1. Il sistema invita l'utente ad inserire dei valori ammissibili



CASO D'USO FORMATO INFORMALE UC10: Visualizza lista della spesa

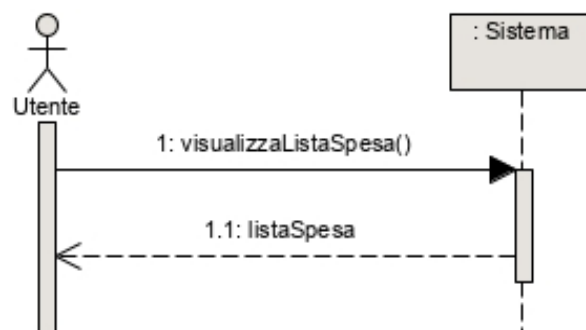
LIVELLO: Sottofunzione

SCENARIO PRINCIPALE:

1. L'utente accede alla lista della spesa
2. Il sistema restituisce la lista della spesa

SCENARI ALTERNATIVI:

- 1a. La lista della spesa è vuota
 1. Il sistema restituisce all'utente una lista della spesa vuota



CASO D'USO FORMATO INFORMALE UC11: Gestisci lista della spesa

LIVELLO: Obiettivo Utente

SCENARIO PRINCIPALE:

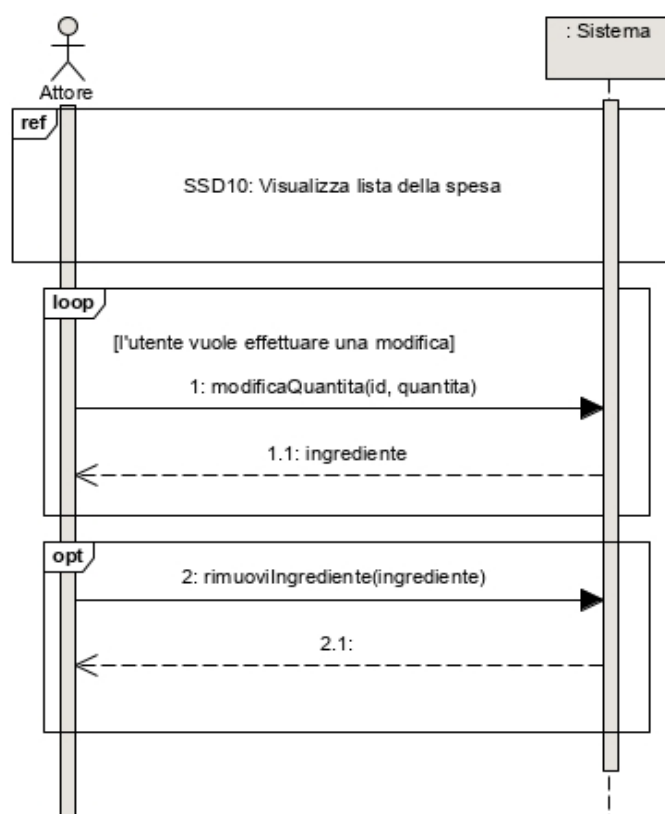
1. L'utente accede alla lista della spesa (rif. Caso d'Uso UC10 "Visualizza lista spesa")
2. (OPZIONALE) L'utente decide di modificare un ingrediente nella lista della spesa
 - 2.1. L'utente modifica il quantitativo da acquistare
 - 2.2. L'utente conferma la modifica effettuata
 - 2.3. Il sistema controlla la correttezza del valore inserito
 - 2.4. Il sistema salva i dati dell'ingrediente da acquistare

L'utente ripete il passo 2 ogni volta che vuole modificare un ingrediente da acquistare

3. (OPZIONALE) L'utente decide di eliminare l'ingrediente
 - 3.1. L'utente elimina l'ingrediente
 - 3.2. Il sistema elimina l'ingrediente dalla lista

SCENARI ALTERNATIVI:

- 2.3a. Il valore inserito non è ammissibile
 1. Il sistema invita l'utente ad inserire dei valori ammissibili



CASO D'USO FORMATO INFORMALE UC12: Aggiungi ingrediente alla lista della spesa

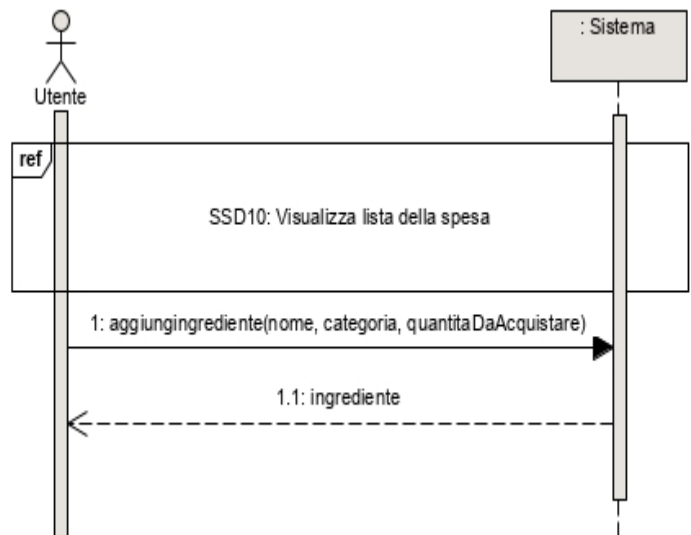
LIVELLO: Obiettivo Utente

SCENARIO PRINCIPALE:

1. L'utente accede alla lista della spesa (rif. Caso d'Uso UC10 "Visualizza lista spesa")
2. L'utente inserisce i dati dell'ingrediente da acquistare specificandone nome, categoria e quantità da acquistare
3. L'utente aggiunge l'ingrediente da acquistare alla lista della spesa
4. Il sistema controlla la correttezza dei dati inseriti
5. Il sistema aggiunge l'ingrediente alla lista della spesa

SCENARI ALTERNATIVI:

- 4a. I valori inseriti non sono ammissibili
 1. Il sistema invita l'utente ad inserire dei valori ammissibili

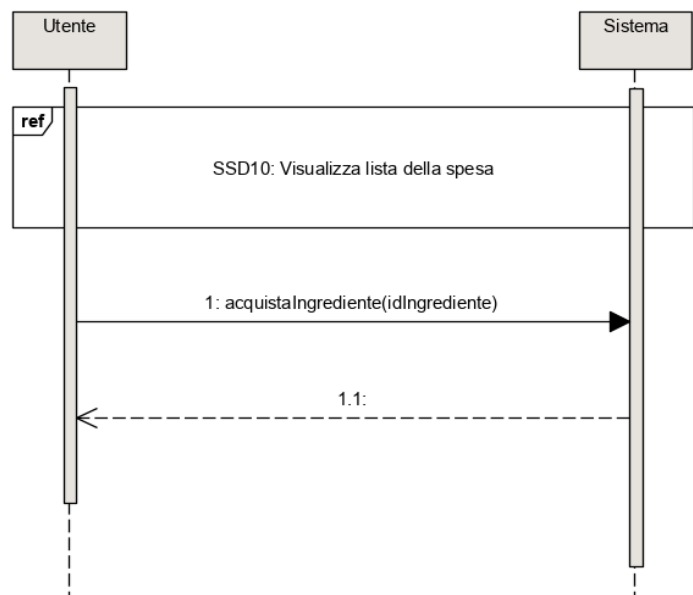


CASO D'USO FORMATO INFORMALE UC13: Acquista ingrediente

LIVELLO: Obiettivo Utente

SCENARIO PRINCIPALE:

1. L'utente accede alla lista della spesa (rif. Caso d'Uso UC10 "Visualizza lista spesa")
2. L'utente selezione l'ingrediente che ha acquistato
3. L'utente conferma di aver acquistato l'ingrediente selezionato
4. Il sistema aggiorna la quantità disponibile dell'ingrediente nel catalogo



CASO D'USO FORMATO INFORMALE UC14: Birra del giorno

LIVELLO: Obiettivo Utente

SCENARIO PRINCIPALE:

1. L'utente accede alla funzionalità
2. L'utente comunica al sistema la quantità di birra che intende produrre
3. Il sistema controlla la correttezza del valore inserito
4. Il sistema cerca la ricetta che massimizza la quantità di ingredienti disponibili
5. Il sistema restituisce la pagina della ricetta all'utente

SCENARI ALTERNATIVI:

- 3a. Il valore inserito non è ammissibile
 1. Il sistema invita l'utente a inserire valori ammissibili
- 4a. Non è presente alcuna ricetta nel ricettario
 1. Il sistema informa l'utente della mancanza di ricette nel ricettario
- 4b. Non è presente alcun ingrediente nel catalogo ingredienti
 1. Il sistema informa l'utente della mancanza di ingredienti nel catalogo ingredienti
- 5a. Nessuna ricetta tra quelle nel ricettario è producibile con il quantitativo inserito
 1. Il sistema informa l'utente della mancanza di ricette producibili

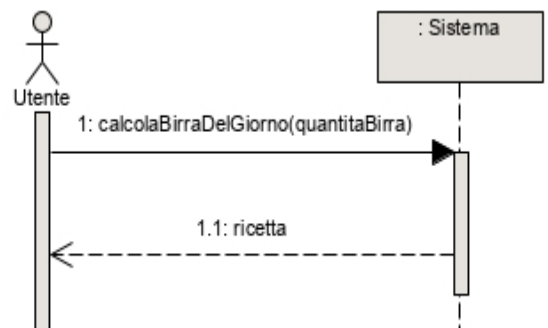


DIAGRAMMA DI DOMINIO

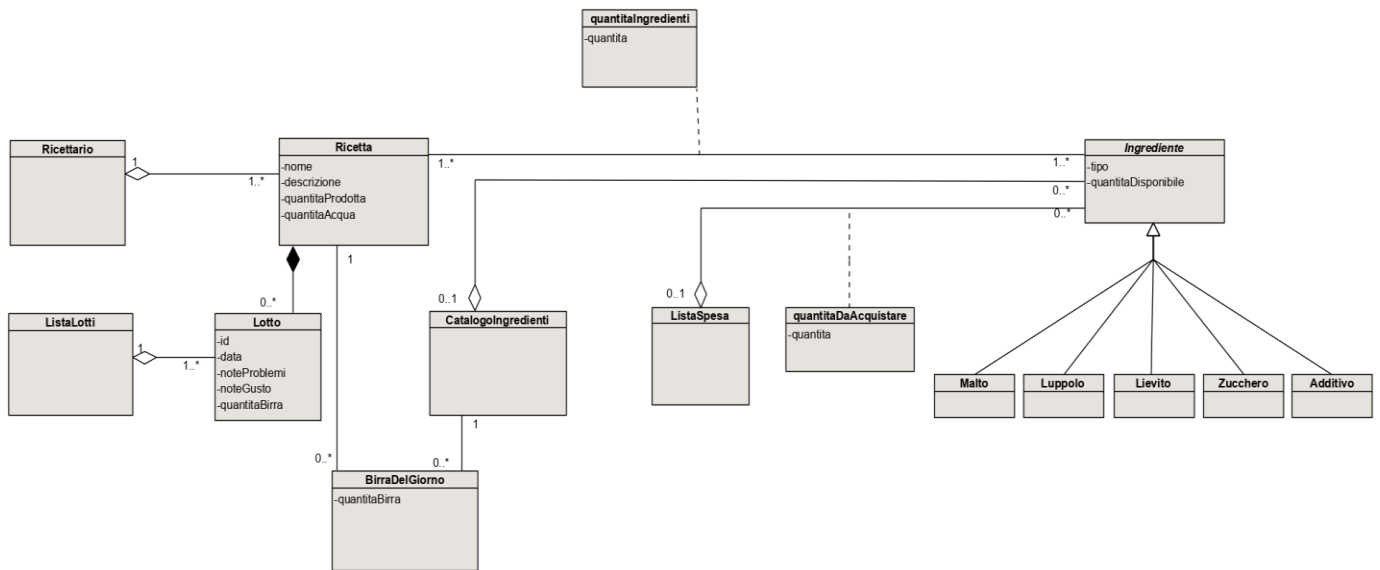


DIAGRAMMA DI STATO: Ingrediente

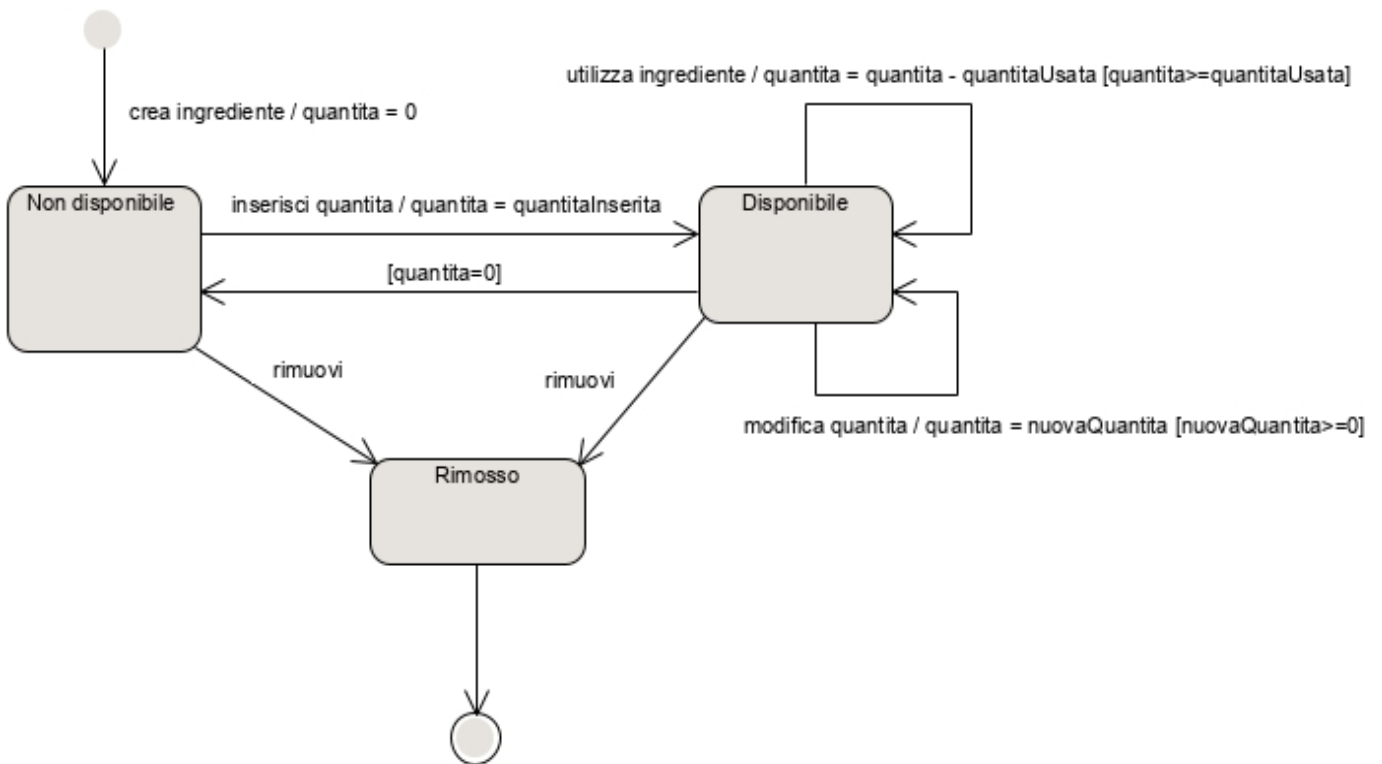
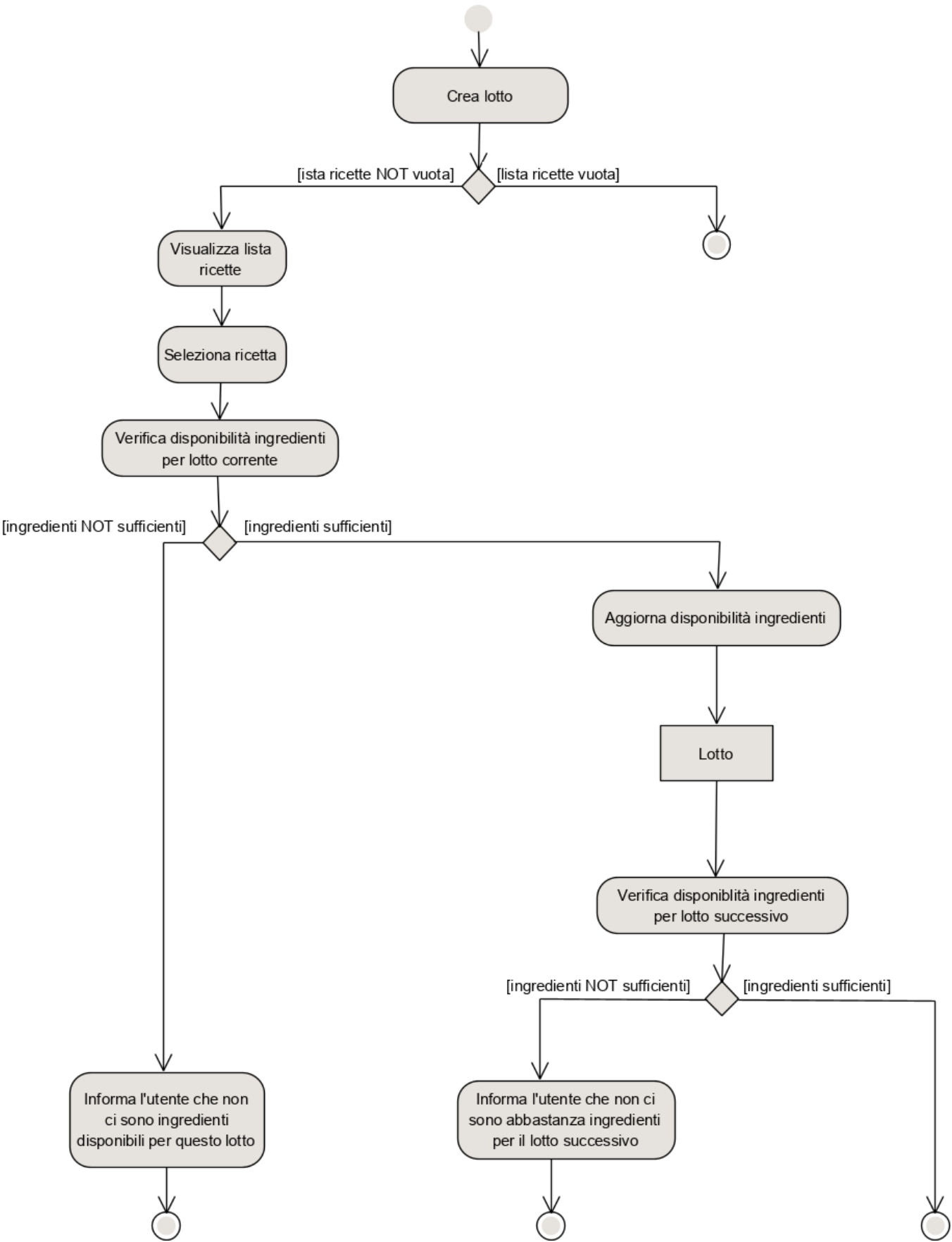


Diagramma di Attività: Crea lotto



PARTE 2: PROGETTAZIONE

DIAGRAMMA DI ARCHITETTURA SOFTWARE

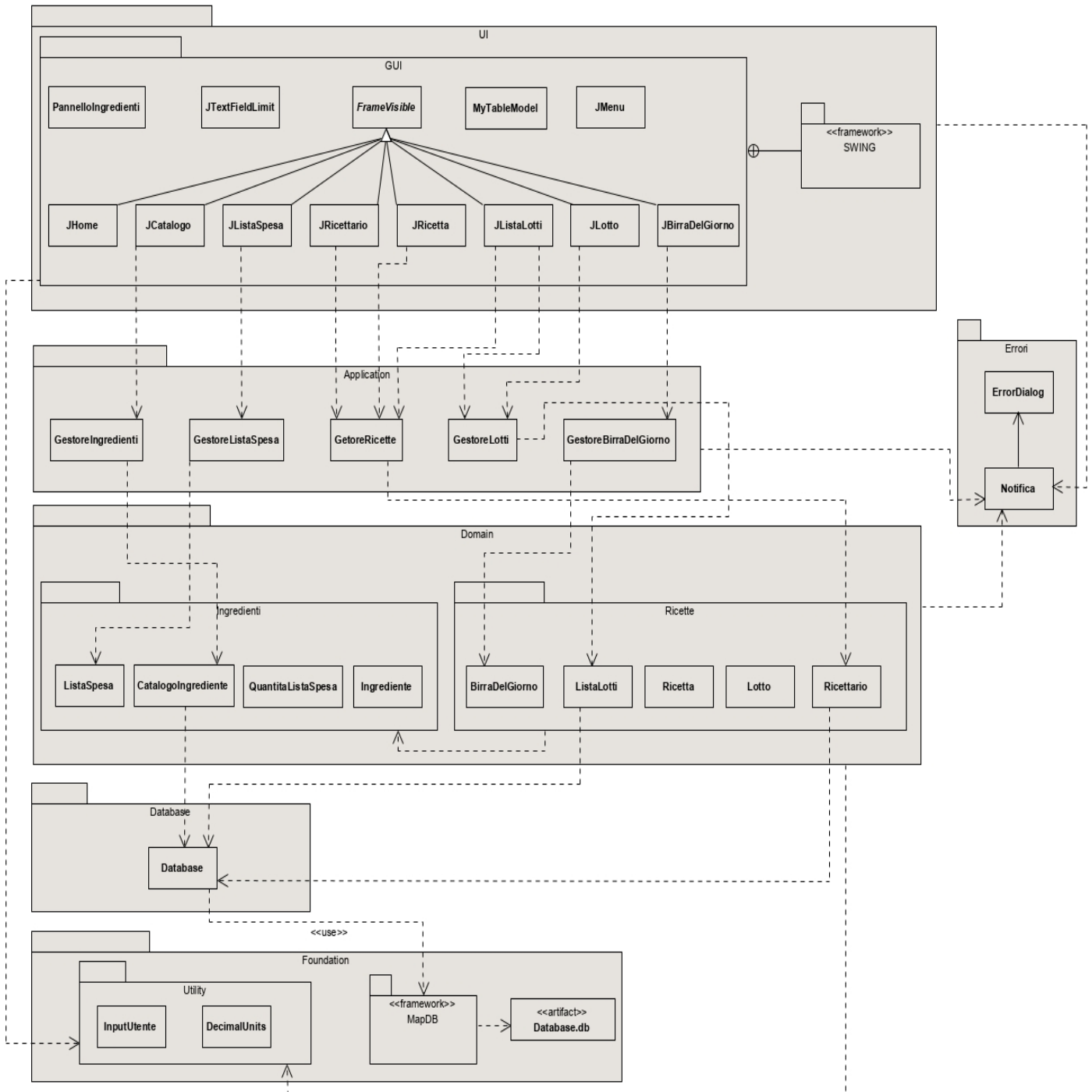




DIAGRAMMA CLASSI DI PROGETTO: Gestori

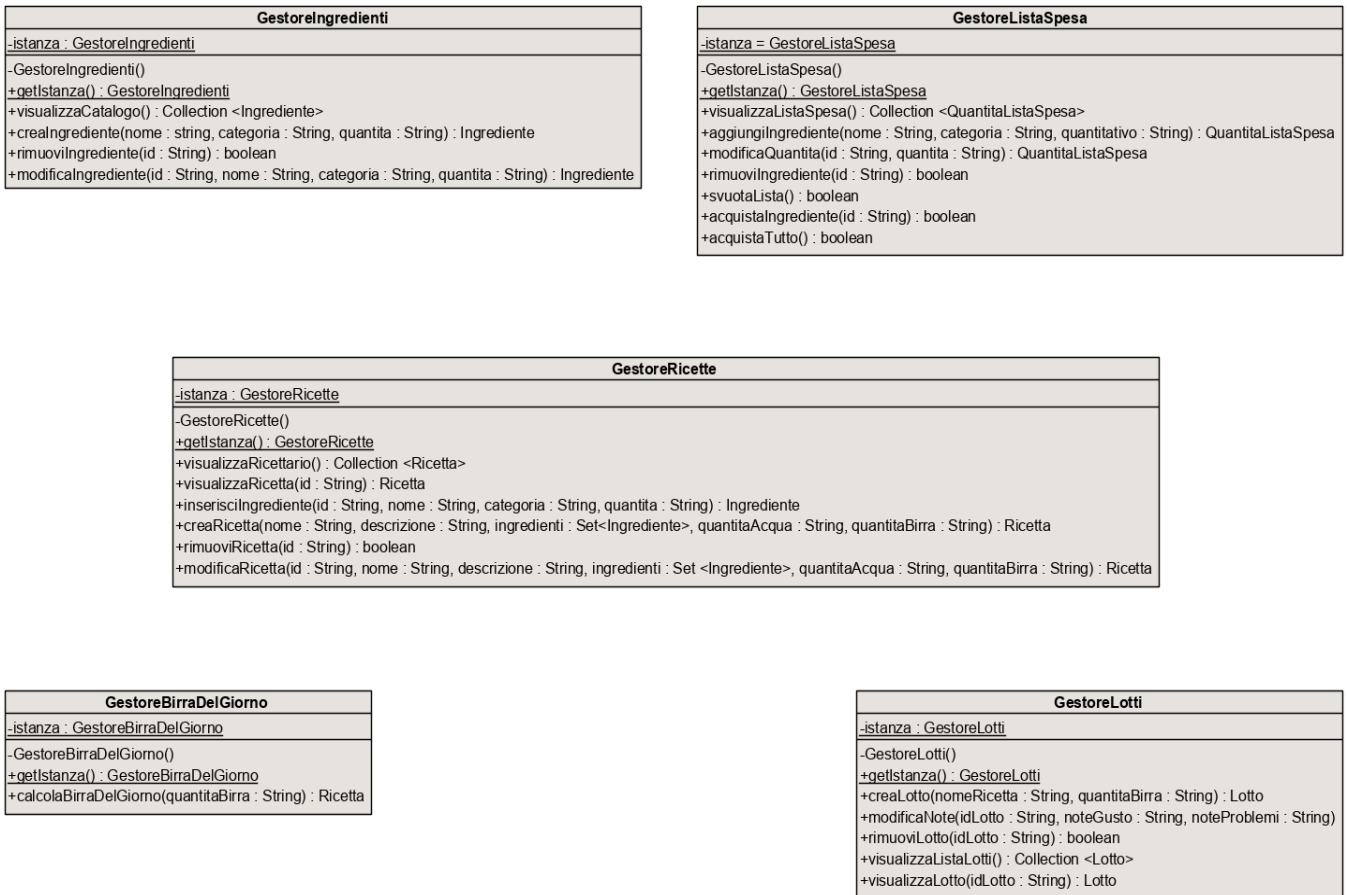


DIAGRAMMA CLASSI DI PROGETTO: Errori

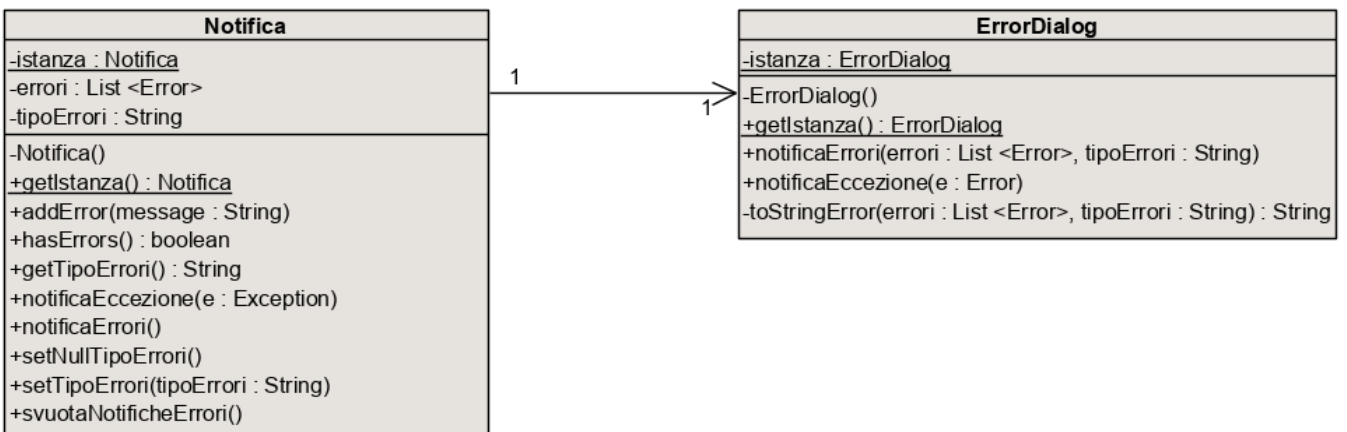






DIAGRAMMA CLASSI DI PROGETTO: Database

Database
-db : DB
-istanza : Database
-Database(path : String)
+getIstanza() : Database
+openMapDB(nomeMappa : String) : HTreeSet<?, ?>
+closeDB()
+getDB() : DB
-setDBNull()

DIAGRAMMA CLASSI DI PROGETTO: Utility

DecimalUtils
-DecimalUtils()
+round(value : double, numberOfDigitsAfterDecimalPoint : int) : double

InputUtente
-CAMPO : String = "Il campo \"
-InputUtente()
+isStringaVuota(str : String, field : String) : boolean
+isPositive(str : String, field : String) : boolean
+isNumber(str : String, field : String) : boolean
+rimuoviWhiteSpaces(str : String) : String
+convertToNumber(str : String) : double

DIAGRAMMA DI SEQUENZA: UC2 - Crea ricetta

Per semplicità e per una maggiore comprensione si è omesso di mettere la gestione degli errori e il controllo dell'input nei diagrammi di sequenza.

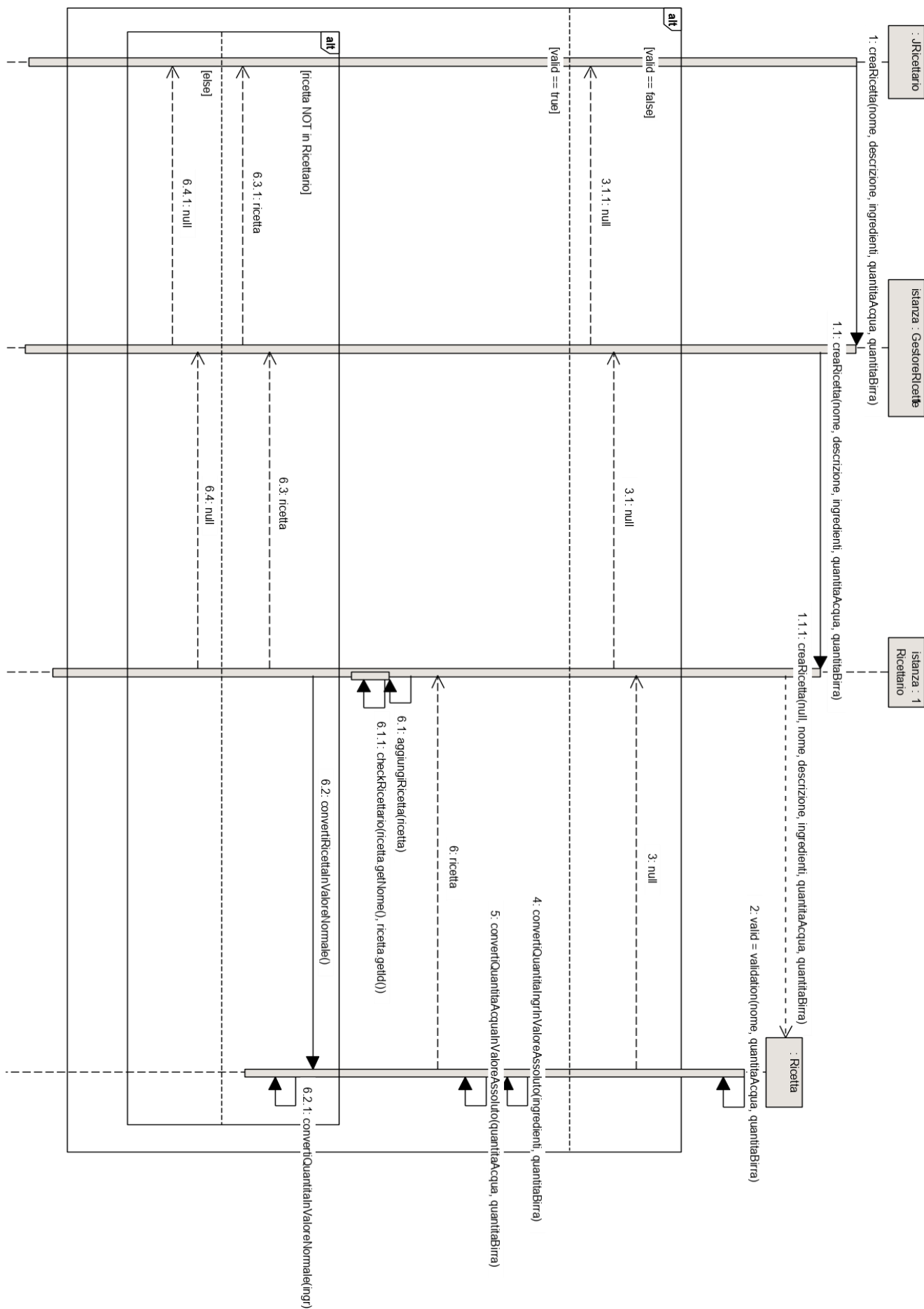


DIAGRAMMA DI SEQUENZA: UC4 - Crea lotto

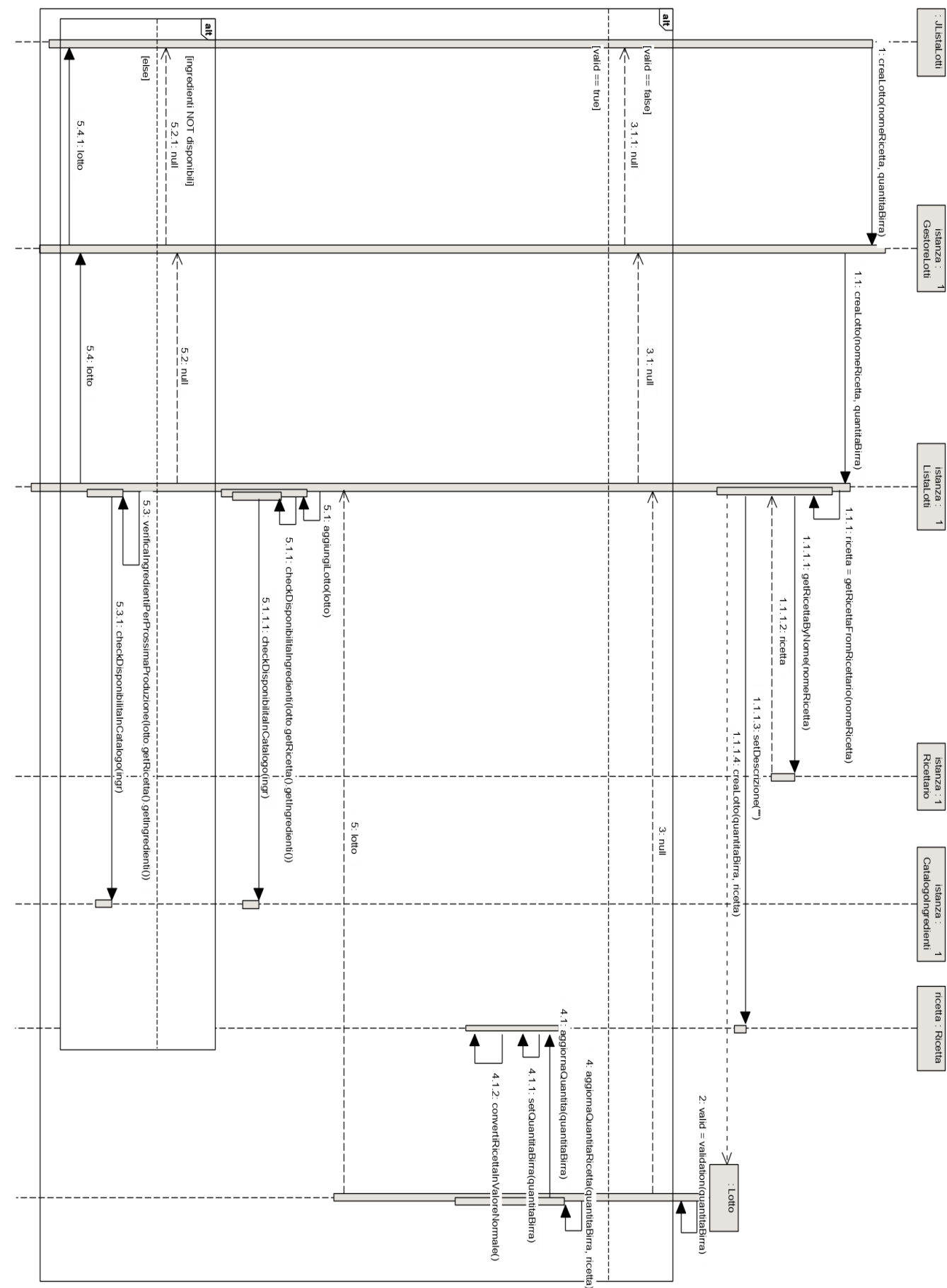


DIAGRAMMA DI SEQUENZA: UC9 - Aggiungi ingrediente al catalogo

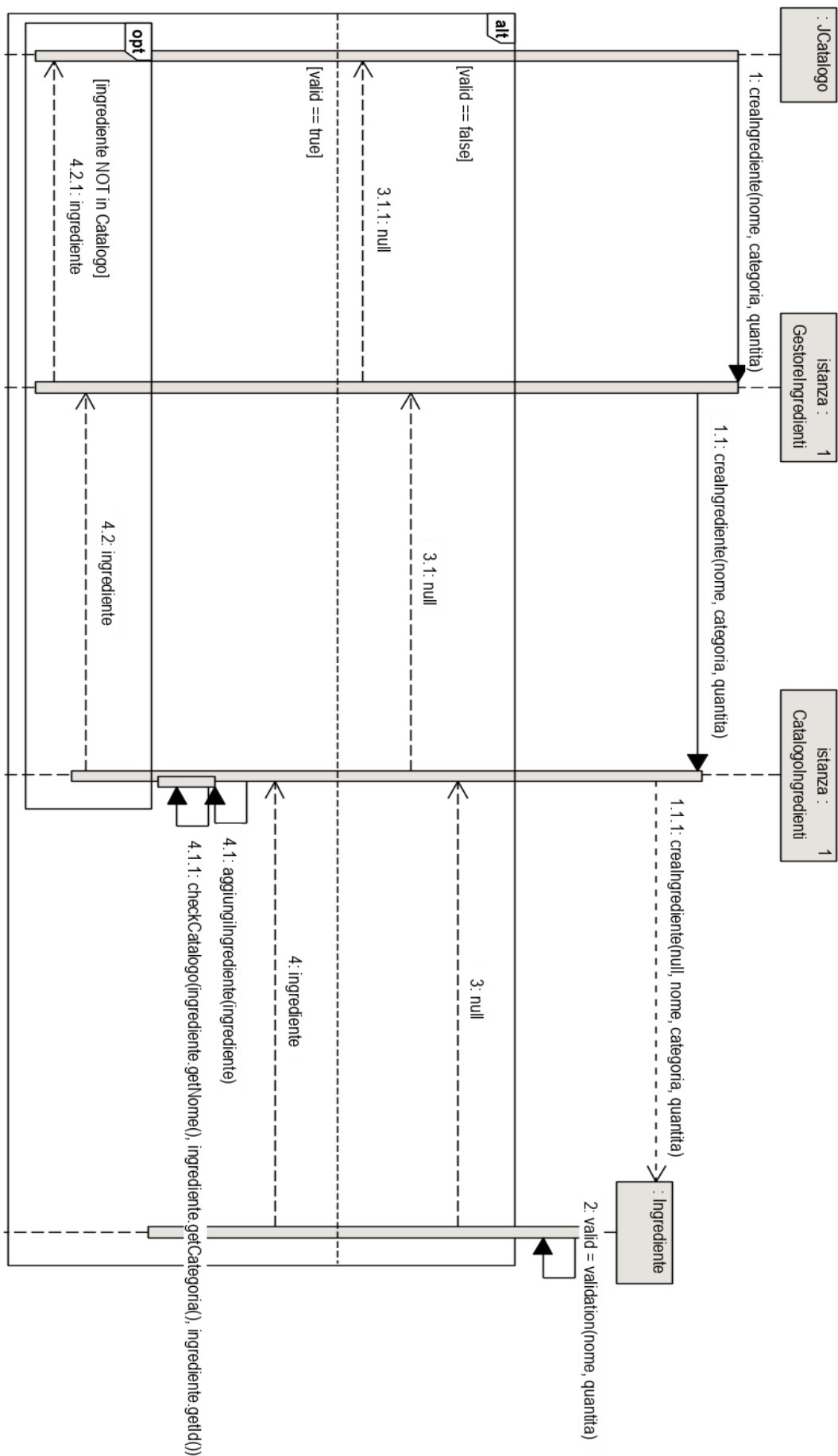


DIAGRAMMA DI SEQUENZA: UC12 - Aggiungi ingrediente a lista spesa

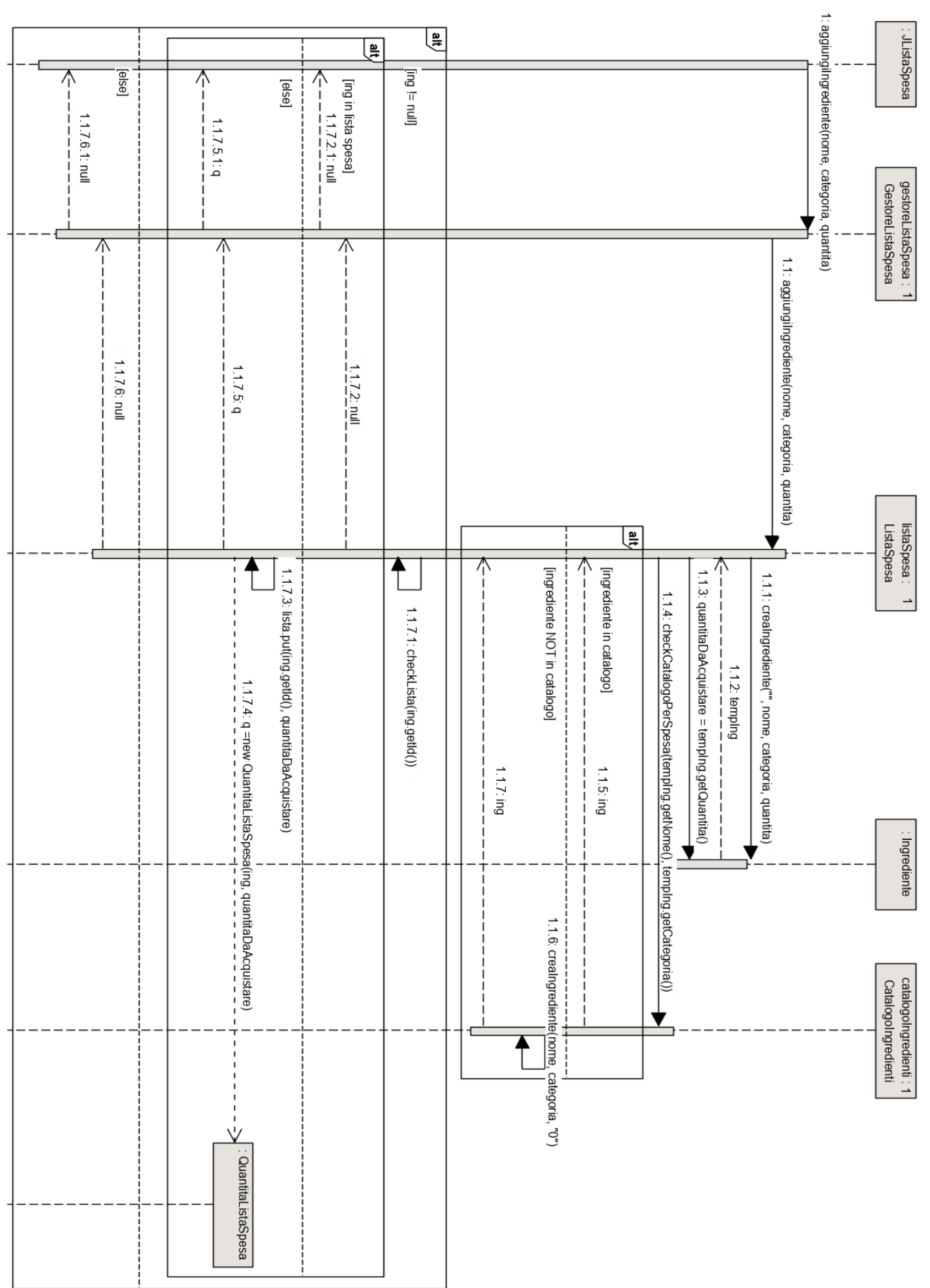
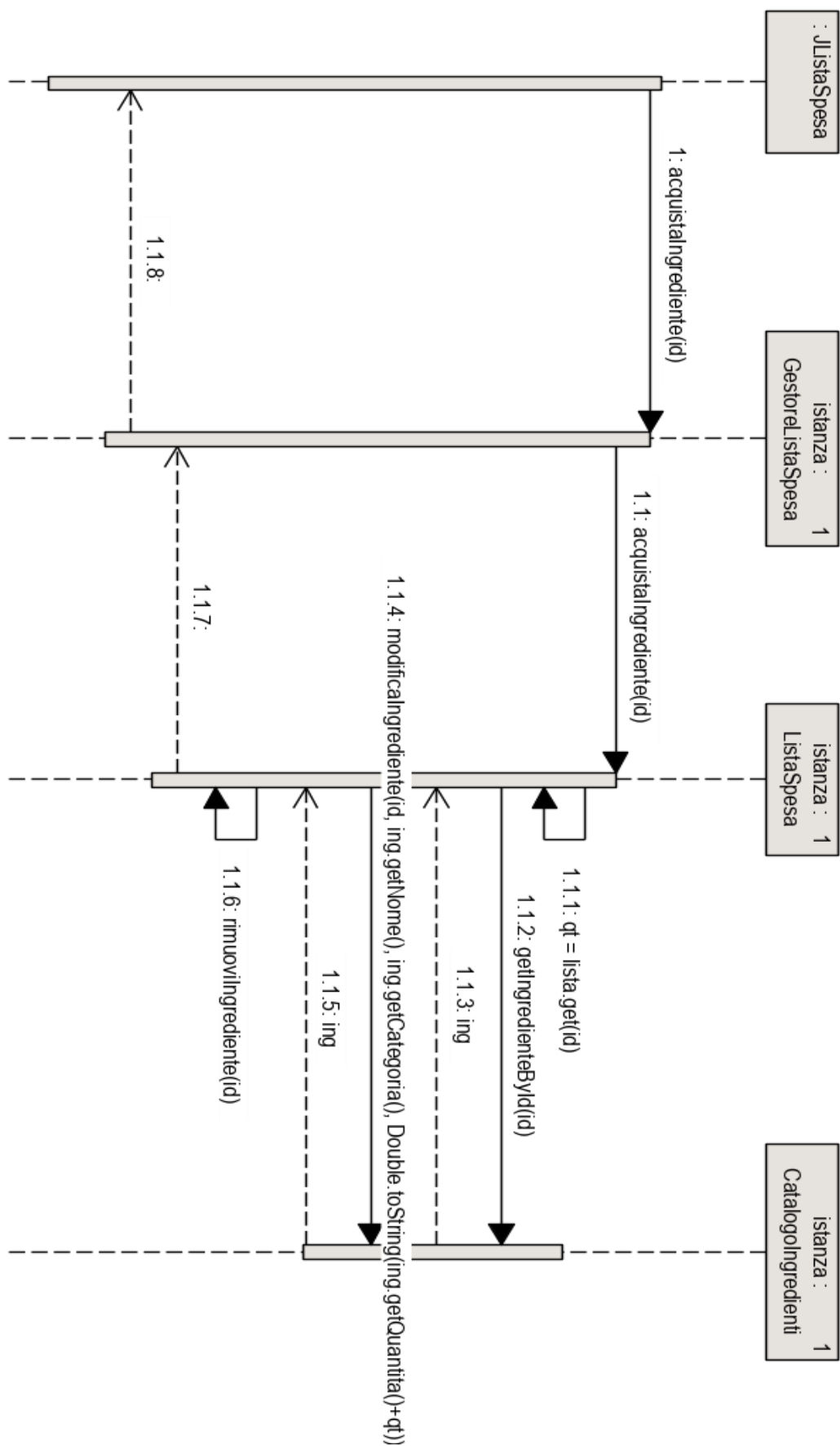


DIAGRAMMA DI SEQUENZA: UC13 - Acquista ingrediente



PATTERN UTILIZZATI

Nell'applicazione sono stati utilizzati quattro pattern:

State pattern: è stato applicato il pattern state per tenere traccia della pagina attualmente in uso, affinché la *JMenu* possa chiuderla e aprire quella richiesta dall'utente.

La classe *JMenu* ha un attributo `frameVisibile` che indica lo stato corrente dell'applicazione, ossia qual è la pagina attualmente visualizzata dall'utente. Ogni volta che l'utente vuole cambiare pagina la *JMenu* chiude la pagina corrente attraverso l'attributo `frameVisibile` e apre quella richiesta. Il nuovo frame, quindi, andrà a impostare il nuovo stato (cioè la nuova pagina aperta) con la chiamata `menu.setFrameVisibile(frameCorrente)` di *JMenu*.

Singleton pattern: è stato applicato in *JMenu* perché la stessa istanza è utilizzata da tutti i frame della GUI ad eccezione di *JHome*, in tutti i gestori (*GestoreBirraDelGiorno*, *GestoreIngredienti*, *GestoreListaSpesa*, *GestoreLotti* e *GestoreRicette*), in *CatalogoIngredienti*, *ListaSpesa*, *ListaLotti* e *Ricettario* (poiché è sufficiente solamente un'istanza), in *Database*, in *Notifica* ed in *ErrorDialog*.

Notification pattern (usato con i pattern *Error Dialog* e *Singleton*): una notifica è un oggetto (della classe *Notifica*) che il dominio, ma non solo, usa per raccogliere informazioni sugli errori durante la convalida dei dati acquisiti dall'utente o su eventuali eccezioni sollevate e li notifica a un oggetto *ErrorDialog* che provvederà a notificarli allo stato di presentazione.

Error Dialog pattern (usato con i pattern *Notification* e *Singleton*): nei gestori le eccezioni sollevate o gli eventuali errori di convalida dell'input dell'utente vengono notificati dall'oggetto *Notifica* all'oggetto *ErrorDialog* che li notificherà a sua volta a una finestra di dialogo della GUI.

DESIGN PRINCIPLES

Sono stati applicati tre principi **SOLID**: **Single Responsibility Principle (SRP)**, **Liskov Substitution Principle (LSP)**, **Dependency Inversion Principle (DIP)**.

I principi LSP e DIP sono stati usati nella GUI dove ci sono le classi *JHome*, *JCatalogo*, *JListaSpesa*, *JRicettario*, *JRicetta*, *JListaLotti*, *JLotto*, *JBirraDelGiorno* che estendono la classe astratta *FrameVisibile*

Come principi a livello di package di coesione è stato utilizzato il **Release Reuse Equivalency Principle (REP)** nel caso del package *Utility* che contiene le classi *InputUtente* e *DecimalUnits* che vengono usate dalle classi di dominio e dalla GUI.

Mentre per quelli a livello di package di accoppiamento è stato applicato l'**Acyclic Dependencies Principle (ADP)**, in quanto dal diagramma dell'architettura software non risultano cicli tra package.

PRINCIPI PHAME

Nell'applicazione è stato applicato il principio dell'**incapsulamento** grazie all'inserimento dei gestori che nascondono all'utente i dettagli implementativi dei componenti interni al sistema

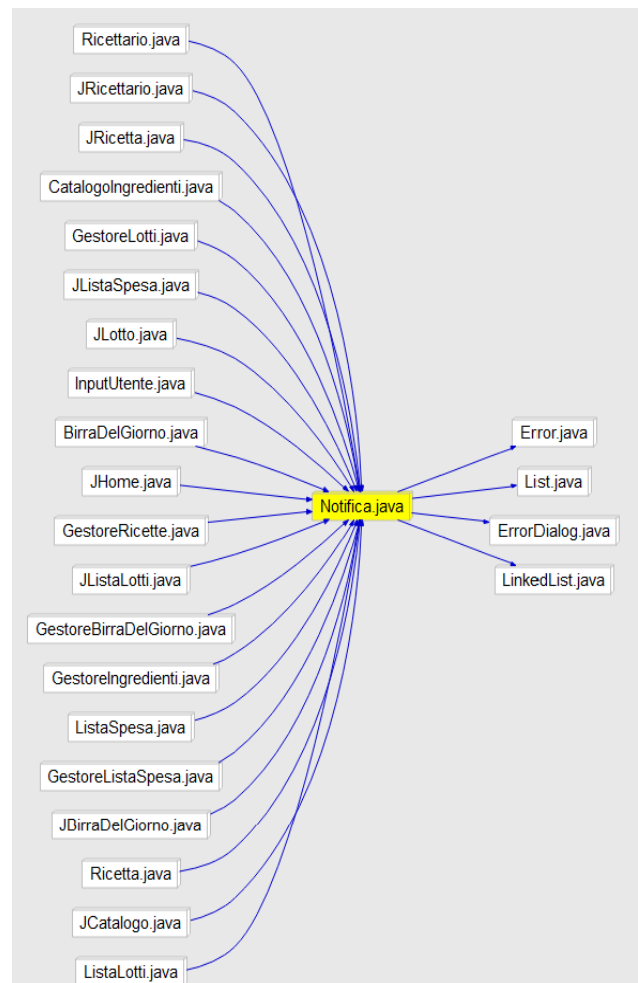
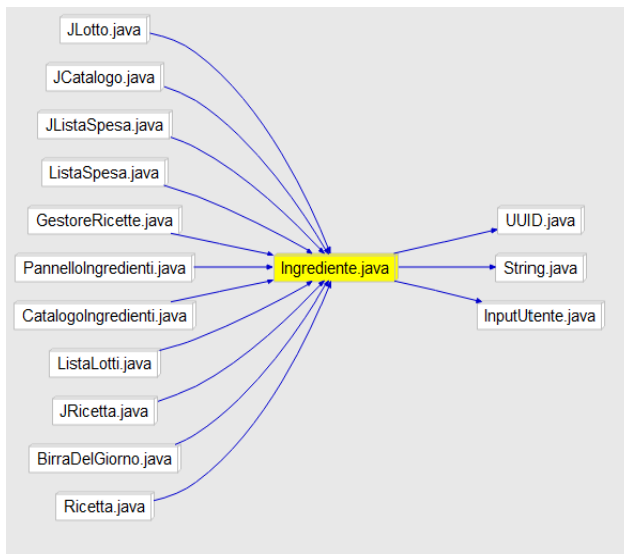
PERSISTENZA

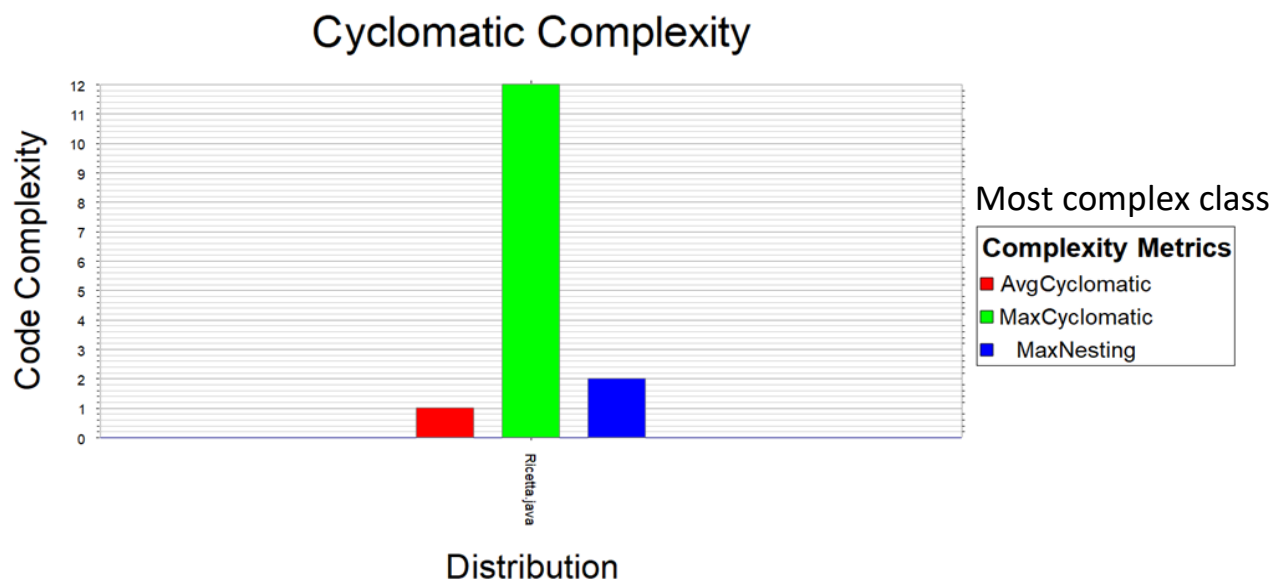
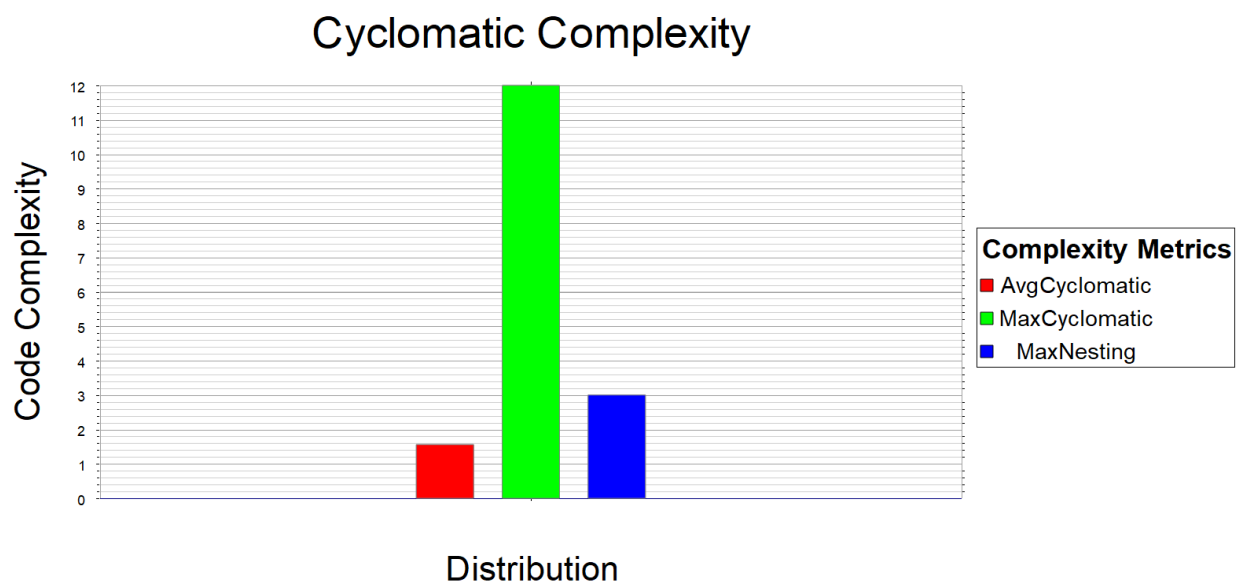
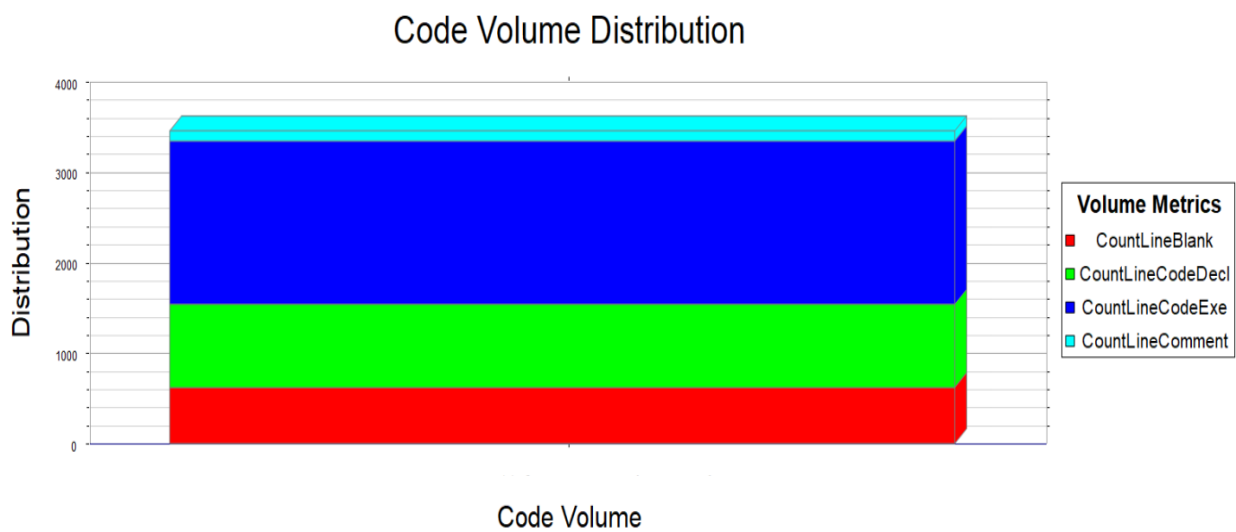
Per la gestione dei dati persistenti si è utilizzata la libreria *MapDB* che permette di creare un database locale accessibile attraverso delle strutture dati simili alle collections di java.

PARTE 3: ANALISI TECNICA

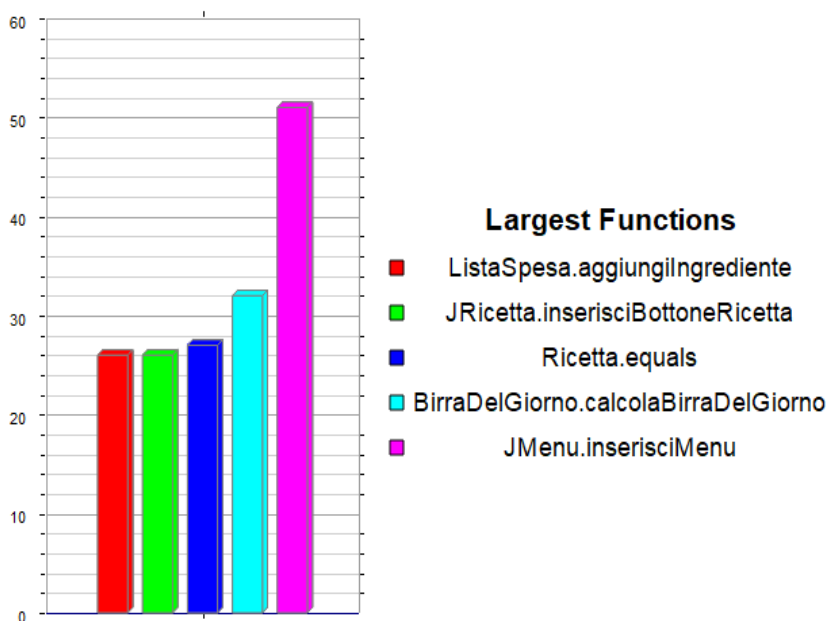
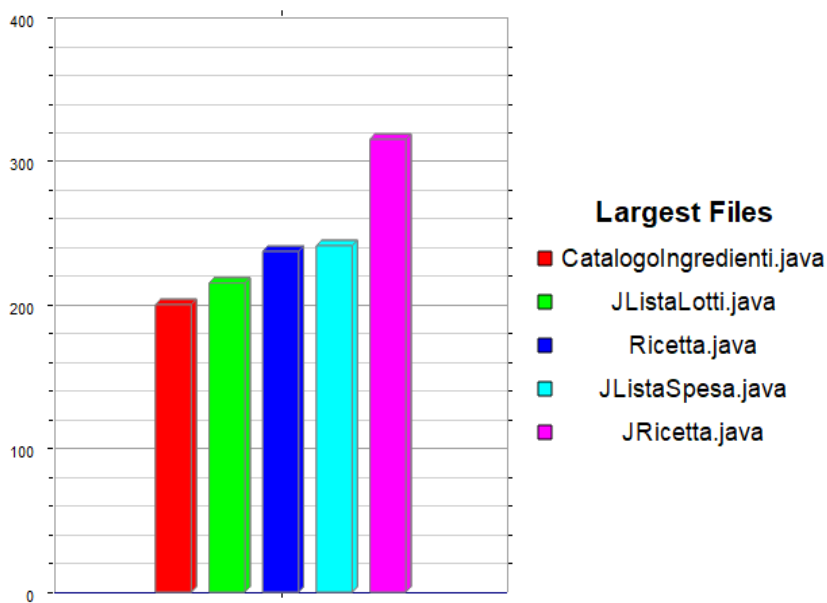
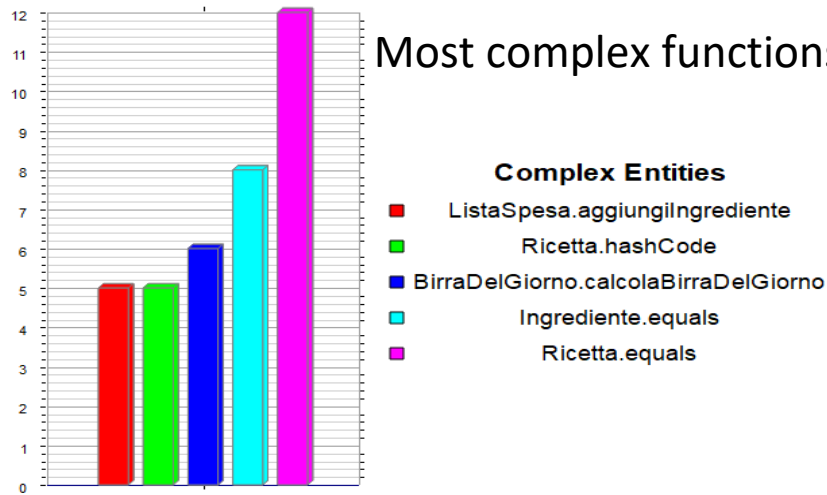
ANTIPATTERN

Da un'analisi sul tool Understand è stato individuato, per quanto riguarda le dipendenze tra le sole classi dell'applicazione ad eccezione di quelle appartenenti alle librerie java e ai vari framework utilizzati, l'antipattern **global butterfly** nelle classi *Ingrediente* e *Notifica*, anche se in quest'ultima è giustificato per il fatto che tutti gli oggetti si rivolgono alla classe *Notifica* in caso di errori.





Most complex functions



ANALISI SONARQUBE

BrewDay

Passed

Last analysis: February 15, 2020, 4:07 PM

0 **A**
Bugs



0 **A**
Vulnerabilities

3 **A**
Code Smells

0.0%
Coverage


2.4%
Duplications

3.1k
Java, XML

Bugs  Vulnerabilities 

0 **A**
Bugs


0 **A**
Vulnerabilities

Code Smells 

1h **A**
Debt


started 6 minutes ago

3
Code Smells

Coverage 

0.0%
Coverage

13
Unit Tests

Duplications 

2.4%
Duplications

5
Duplicated Blocks

Sono stati contrassegnati come “falsi positivi” per l'analisi con SonarQube i controlli nelle `validation()` con l'operatore logico `and` e “&” in *Ricetta* e *Ingrediente* così da effettuare sempre i controlli, anche nel caso quelli precedenti falliscano, in modo da poter restituire l'errore all'utente, i metodi `equals()` di *Ricetta* e *Ingrediente* perché considerati troppo complessi, la dichiarazione di una variabile `d` non utilizzata nel metodo `isNumber()` di *InputUtente* usata per controllare se la stringa passata come parametro è un double, la firma del metodo `openDB()` di *Database* che restituisce una `HTreeMap` di tipo generico, e infine i nomi dei package che cominciano con *gruppobirra4* (perché contenenti un numero).

ISTRUZIONI PER L'USO

Istruzioni per avviare l'applicazione:

- importare il progetto come maven
- OPZIONALE: creare un database di prova eseguendo la classe InserimentoDBdiProva.java in src/test/java nel package gruppobirra4.brewday
- avviare la classe JHome.java in src/main/java nel package gruppobirra4.brewday.gui

Istruzioni per l'utilizzo

- HOME: pagina d'avvio dell'applicazione, comprende una serie di pulsanti che permettono di passare alla pagina desiderata
- CATALOGO INGREDIENTI: pagina contenente tutti gli ingredienti inseriti nell'applicazione con relativa quantità disponibile, permette di aggiungere un nuovo ingrediente, modificare, ed eliminare un ingrediente selezionato
- RICETTARIO: pagina contenente tutte le ricette presenti nell'applicazione, permette di creare, aprire una ricetta che si vuole modificare o eliminare una ricetta selezionata
- LISTA DELLA SPESA: pagina contenente, come anticipato dal nome, una lista della spesa per gli ingredienti che l'utente vuole acquistare, all'utente è permesso aggiungere un nuovo ingrediente, modificarne la quantità da acquistare ed eliminare un ingrediente selezionato nella lista o svuotare completamente la lista, permette poi di acquistare un singolo ingrediente oppure di acquistare tutti gli ingredienti presenti
- LISTA LOTTI: pagina contenente tutti i lotti precedentemente prodotti, permette di aprire un lotto in modo da poterne modificare le note oppure di eliminarlo
- BIRRA DEL GIORNO: pagina contenente una funziona che restituisce, in base alla quantità di birra da produrre inserita, la pagina della ricetta che massimizza l'utilizzo degli ingredienti presenti nel catalogo.