SETTEMBRE 2022

SAVE THE MEAL

Alessia Laghezza - Anna Metelli - Alessio Novel

UNIVERSITÀ DEGLI STUDI
DI TRENTO

Titolo del documento:

Documento di Architettura

Indice generale

Scopo del documento

Il documento presenta la definizione dell'architettura dell'applicazione web "Save The Meal" utilizzando diagrammi delle classi descritti in Unified Modeling Language (UML) e codice in Object Constraint Laguage (OCL).
Basandosi sul documento precedente che ha definito i diagrammi degli use case, di contesto e dei componenti sarà definita l'architettura del progetto descrivendo le classi che verranno implementate a livello di codice e la logica che regolerà il comportamento del software.
Le classi verranno rappresentate con un diagramma delle classi in linguaggio UML, mentre la logica del comportamento del software sarà rappresentata in linguaggio OCL.

Diagramma delle classi

In questa sezione del documento vengono riportare le classi previste nello sviluppo dell'applicazione web "Save The Meal". Le classi descritte sono caratterizzate da un nome, una lista di attributi che identificano i dati gestiti dalla classe e una lista di metodi che definiscono il comportamento e le operazioni della classe. Tramite le associazioni tra classi è possibile ricavare ulteriori informazioni sulle relazioni tra di esse.

Utenti

Nell'applicazione web "Save The Meal" si sono delineati nei documenti precedenti diverse tipologie di utente (utente anonimo, generico e autenticato). Questi condividono dei metodi e attributi specifici. Sono quindi state individuate tre sottoclassi (Utente anonimo, Utente generico e Utente autenticato) della superclasse Utente che specificheranno la tipologia dell'utente che utilizza il sistema e di conseguenza i metodi e gli attributi che caratterizzano la sua esperienza.

Nella classe Utente anonimo sono specificate delle operazioni verso il sistema esterno Gmail presente nel diagramma di contesto.

La classe Coordinata permetterà di localizzare la posizione dell'utente che verrà poi utilizzata all'interno del sistema per visualizzare la mappa e la lista dei fornitori.

Nella superclasse Utente sono presenti operazioni verso i sistemi esterni GPS e Google Maps già presenti nel diagramma di contesto.

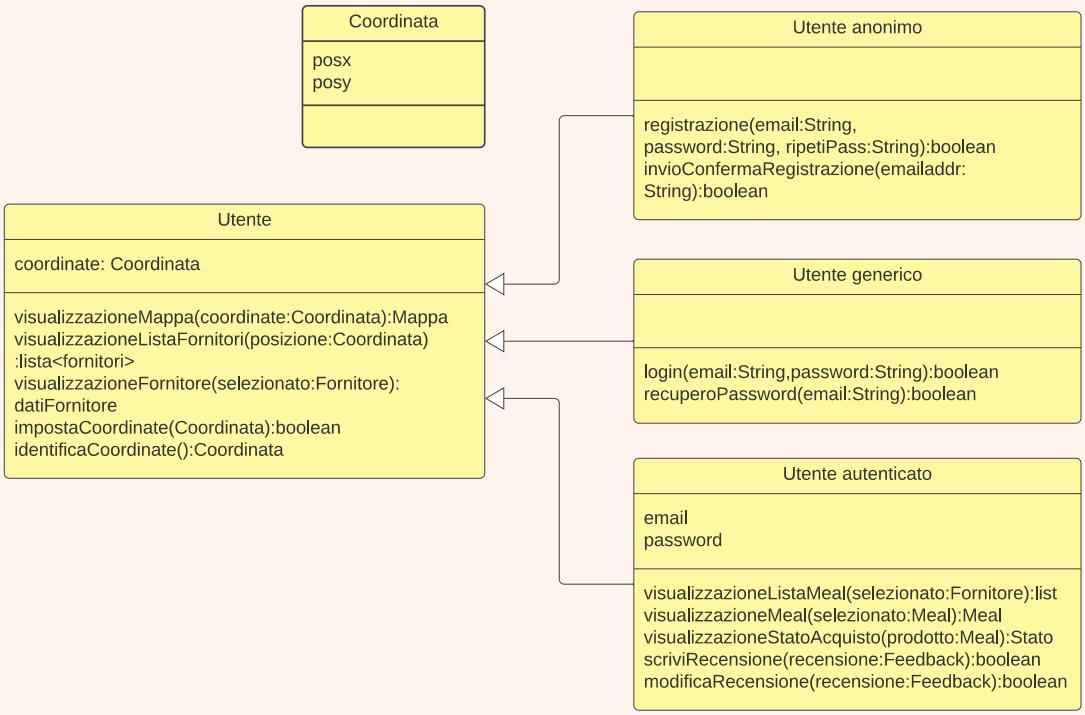


Figura 1. Classi per utenti

Acquisto di Meal

Il processo di acquisto di un Meal è caratterizzato da una serie di classi collegate tra loro. L'Utente autenticato potrà effettuare un Acquisto di uno specifico Meal, il quale sarà stato reso disponibile tra quelli in Vendita dal Fornitore.

All'interno della classe Acquisto sono presenti delle operazioni verso i sistemi esterni Gmail e Paypal per effettuare il pagamento. La classe Fornitore effettua delle operazioni verso i sistemi esterni Maps, Gmail e Paypal.

La classe Indirizzo conterrà le informazioni del fornitore riguardanti la posizione della sua attività.

La classe Stato specificherà lo stato del Meal che apparirà all'utente nello storico dei suoi acquisti.

La classe Dimensione specificherà la dimensione del Meal tra le tre esistenti.

La classe Data verrà utilizzata per registrare i dati delle avvenute operazioni all'interno del sistema e per porre dei limiti ad alcune operazioni.

La classe FasciaOraria identifica la fascia del ritiro dei Meal decisa dal fornitore ed è identificata da due date (una di inizio e una di fine).

La classe Coordinata permetterà di localizzare la posizione del fornitore sulla mappa che gli utenti visualizzeranno.

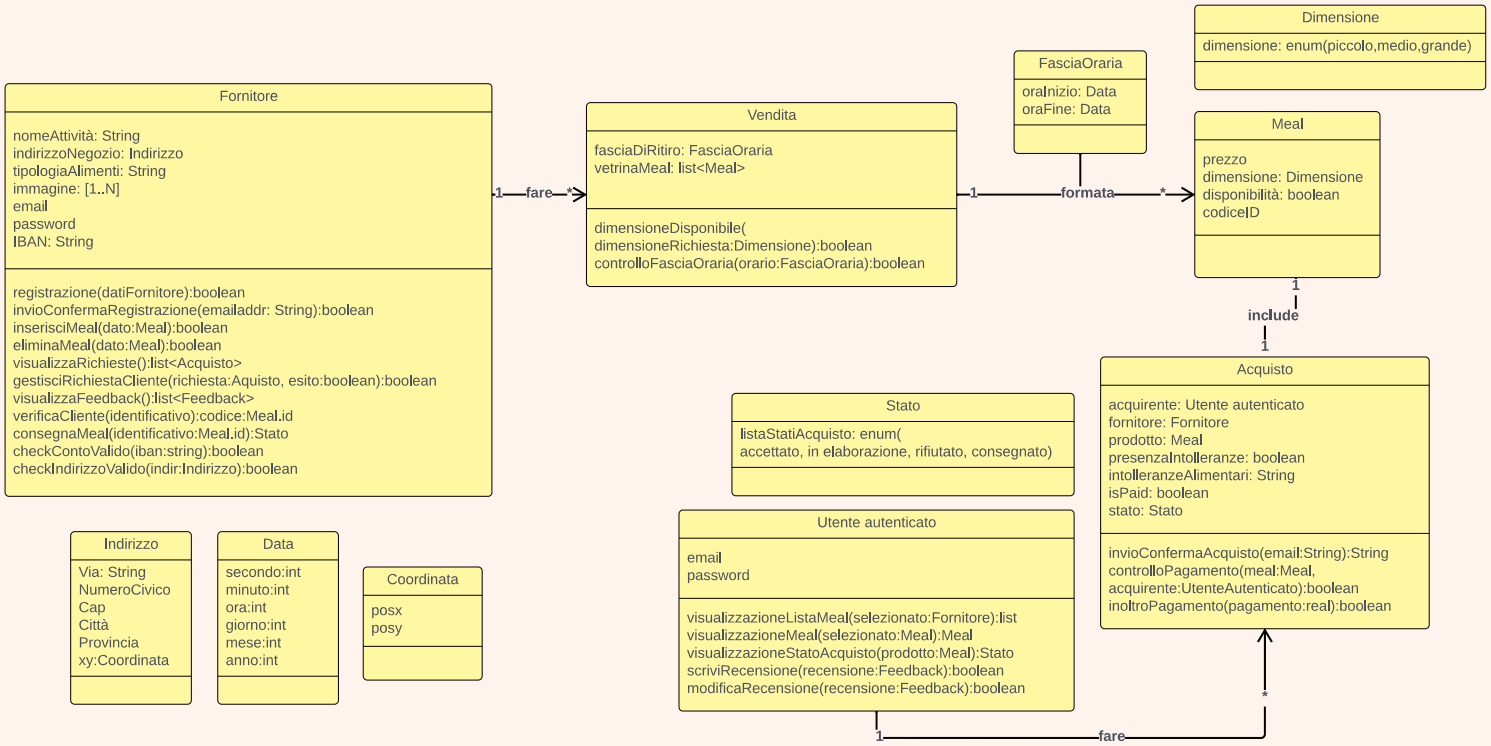


Figura 2. Classi per l'acquisto di un Meal

Feedback

Analizzando le componenti che gestiscono i feedback e la loro scrittura e visualizzazione è stata definita la classe Feeback che permette agli utenti autenticati di lasciare un feedback a uno specifico fornitore. Un feedback vive ed esiste se e solo se si è fatto un acquisto.

La classe Aggettivo specifica i "punti di forza" che l'utente potrà selezionare nella sua recensione tra quelli elencati.

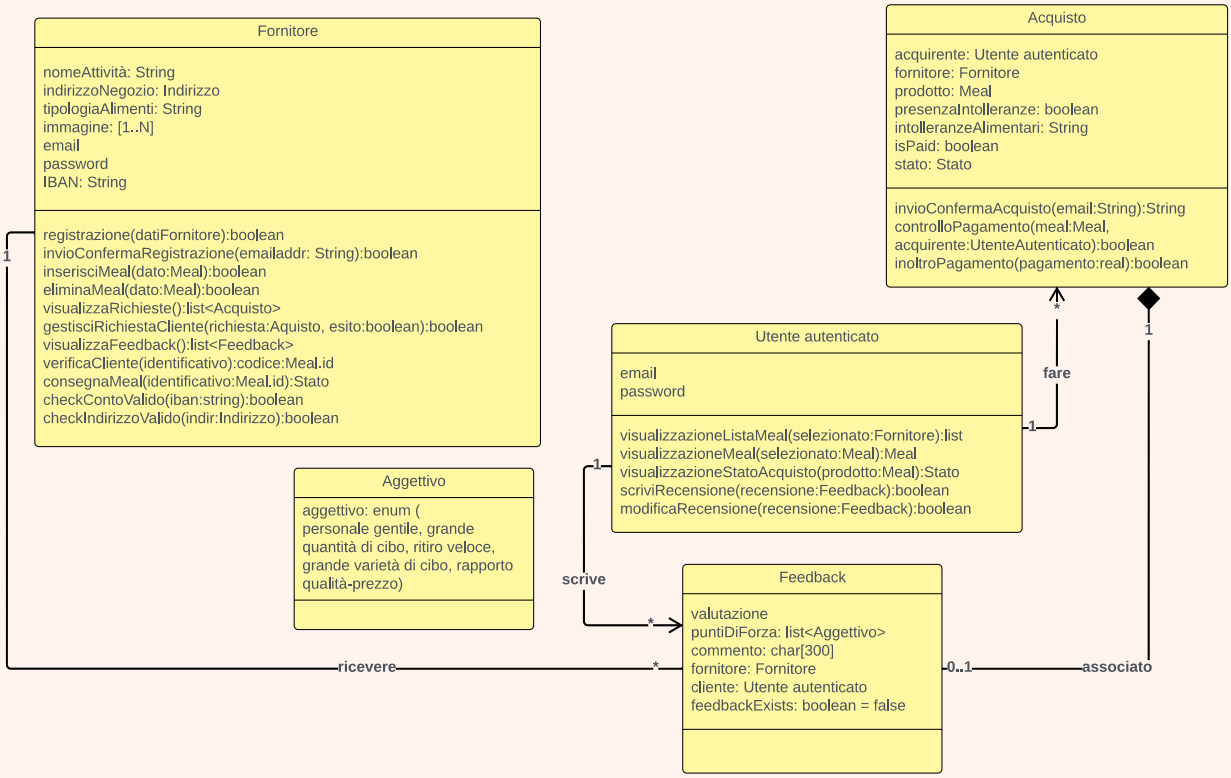
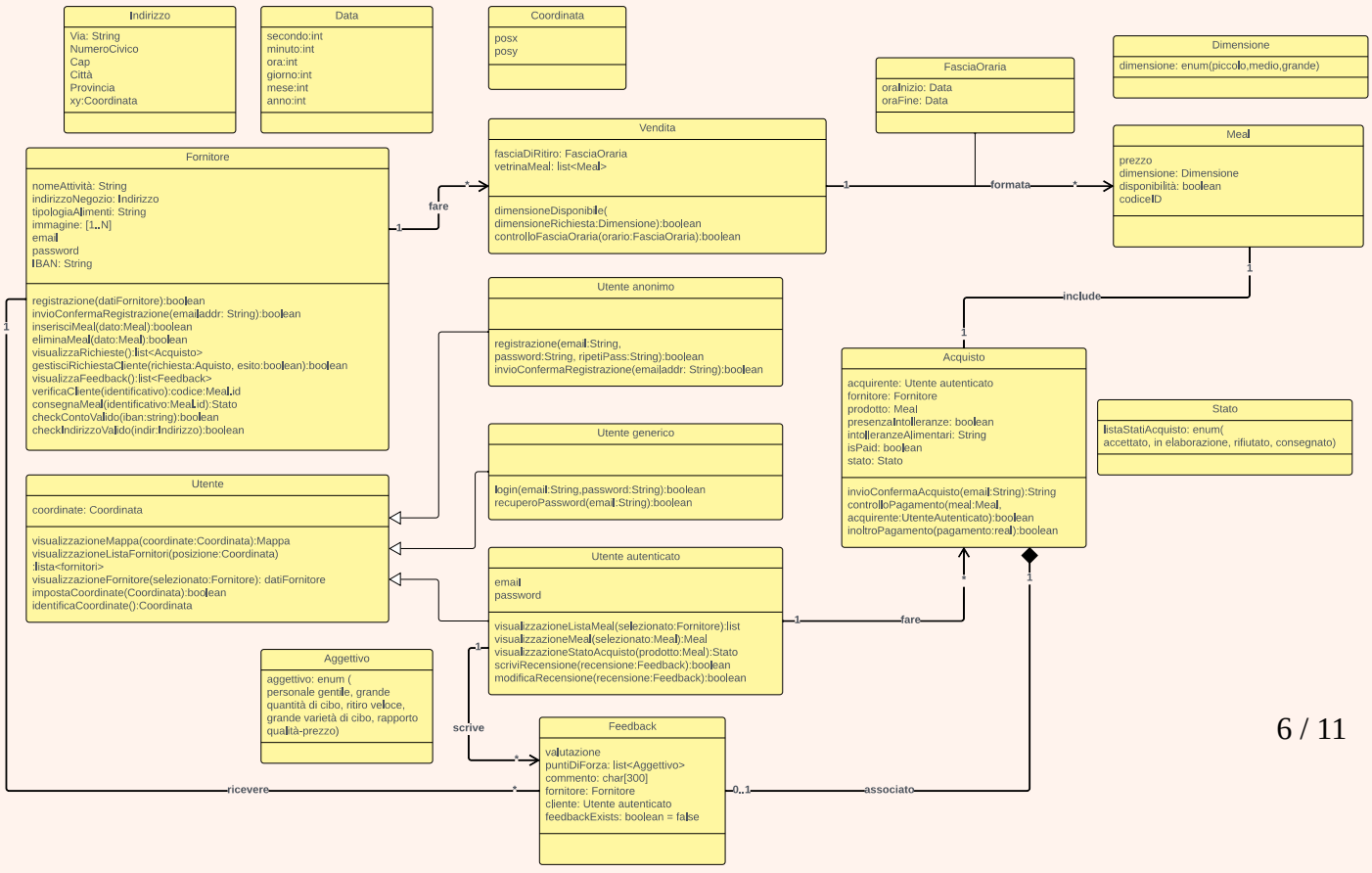


Figura 3. Classi che identificano un feedback

Diagramma delle classi complessivo

È riportato di seguito il diagramma delle classi nella sua interezza per fornire una visualizzazione d'insieme del sistema di "Save The Meal".

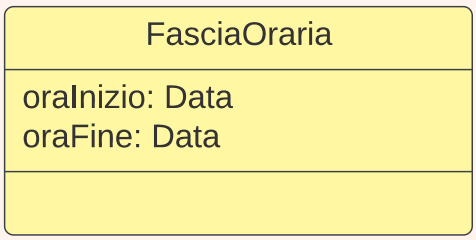


Codice in Object Constraint Language

In questo capitolo del documento viene approfondita la logica del comportamento del software del sistema “Save The Meal” in alcune operazioni tra classe. Verrà utilizzato il linguaggio Object Constraint Language per esprimere in modo chiaro le restrizioni che verranno applicate nel sistema.

Limiti temporali

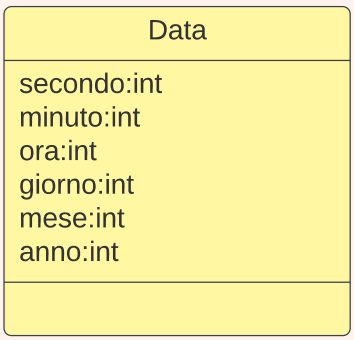
La fascia oraria nella quale è possibile ritirare il Meal acquistato è definita da due date (una di inizio e una di fine). Non sarà possibile avere una fascia oraria dove l’orario di inizio è successivo all’orario di fine.



Questa condizione è espressa in OCL attraverso un’invariante con il codice:

```
context FasciaOraria inv:
oraInizio < oraFine
```

La data sarà espressa seguendo dei vincoli sugli interi che la rappresentano.



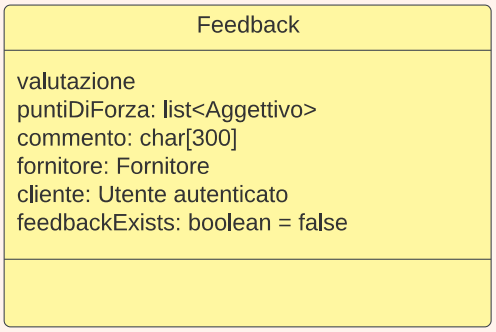
Questa condizione è espressa in OCL attraverso un’invariante con il codice:

```
context Data inv:
0 <= secondo < 60
0 <= minuto < 60
0 <= ora < 24
```

```
0 <= giorno <= 31
0 <= mese <= 12
```

Limiti sul feedback

Un feedback è caratterizzato anche dalla sua valutazione, che sarà visualizzata in “stelle” a livello di front-end. Il valore dell’attributo valutazione sarà quindi limitato tra i valori 0 e 5 compresi.

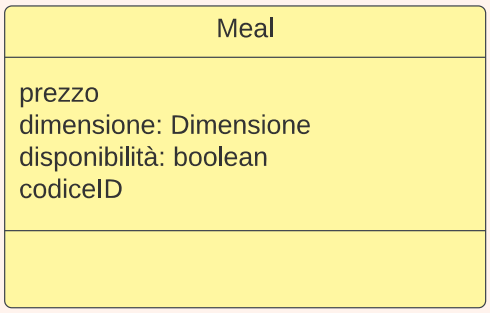


Questa condizione è espressa in OCL attraverso un’invariante con il codice:

```
context Feedback inv:
0 <= valutazione <= 5
```

Meal

Il prezzo di un Meal dovrà essere superiore agli 0€.



Questa condizione è espressa in OCL attraverso un’invariante con il codice:

```
context Meal inv:
prezzo > 0
```

Acquisto

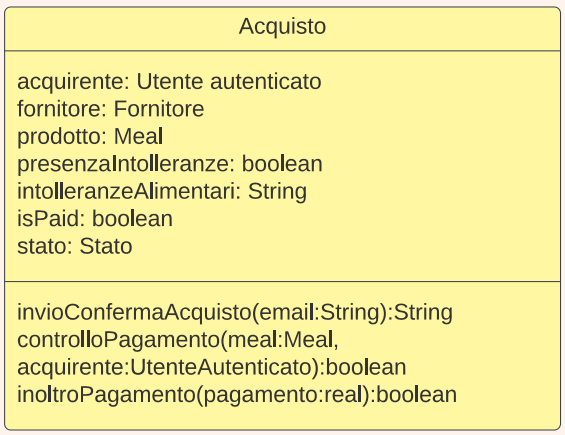
Un utente riceverà una mail di conferma dell'acquisto solo se il fornitore (in presenza di intolleranze) o il sistema (in mancanza) avranno accettato la richiesta di acquisto del Meal.

Lo stato del Meal è aggiornato nella variabile "stato" e la condizione sarà espressa in OCL con il codice:

```
context Acquisto::invioConfermaAcquisto(email:String)
pre: self.stato = accettato
```

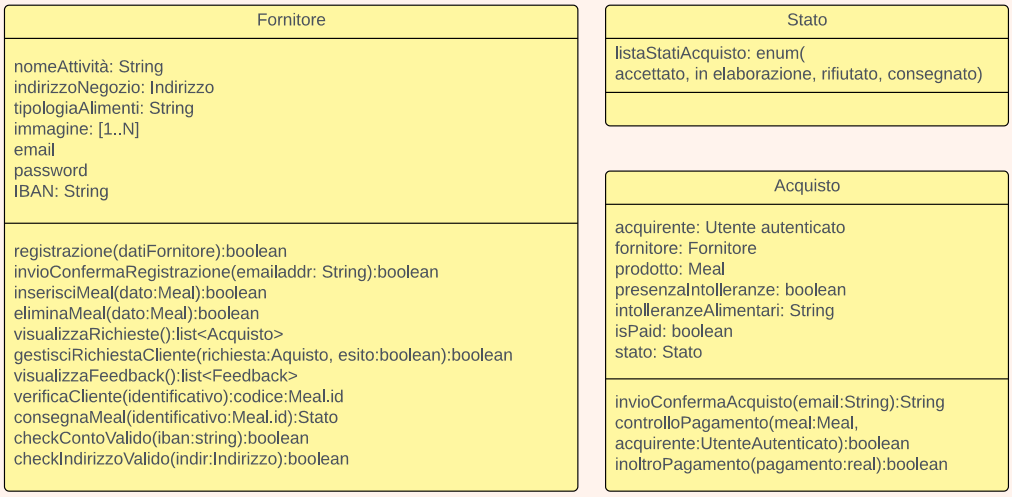
Un utente potrà procedere al pagamento del Meal attraverso il sistema esterno Paypal a condizione che la richiesta di acquisto sia stata accettata dal fornitore (in presenza di intolleranze) o dal sistema (in mancanza) e che il pagamento non sia già stato effettuato. Una volta effettuato il pagamento, il sistema registrerà l'esito positivo dell'operazione nella variabile "isPaid". Questa condizione sarà espressa in OCL con il codice:

```
context Acquisto::inoltroPagamento(pagamento:real)
pre: self.stato = accettato AND self.isPaid = false
post: self.isPaid = true
```



Richieste con intolleranze

Un fornitore potrà gestire una richiesta di acquisto di un utente se in essa sono presenti specifiche riguardanti allergie o intolleranze alimentari. L'acquisto dovrà essere ancora in fase di elaborazione e non essere stato gestito in precedenza.

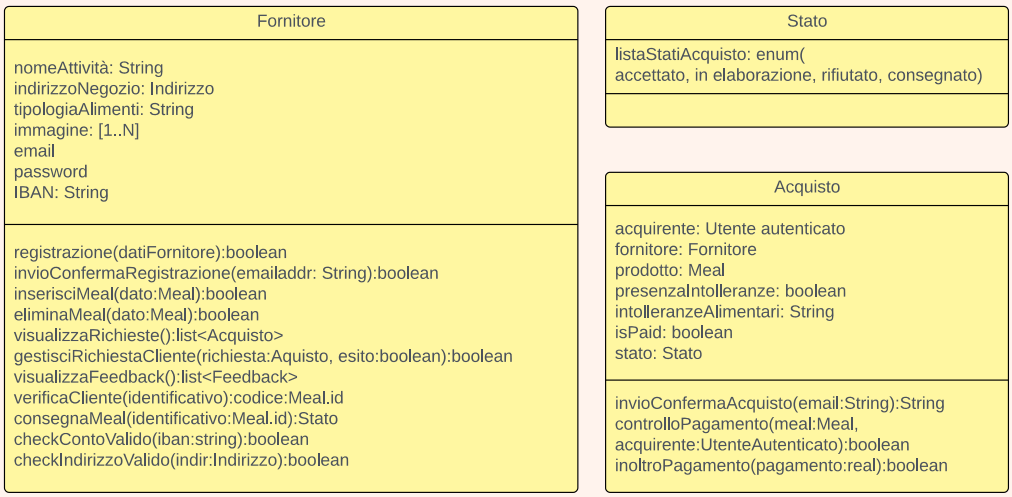


Questa condizione è espressa in OCL attraverso una pre-condizione con il codice:

```
context Fornitore::gestisciRichiestaCliente(richiesta:Acquisto, esito:boolean):
pre: Acquisto.presenzaIntolleranze=true AND Acquisto.stato=in elaborazione
```

Consegna Meal

Perché la consegna del Meal acquistato avvenga correttamente, quando il fornitore visualizzerà le caratteristiche del Meal grazie al codice identificativo presentato dall'utente l'acquisto dovrà risultare "accettato". Dopo aver consegnato il Meal correttamente l'acquisto dovrà risultare "consegnato".



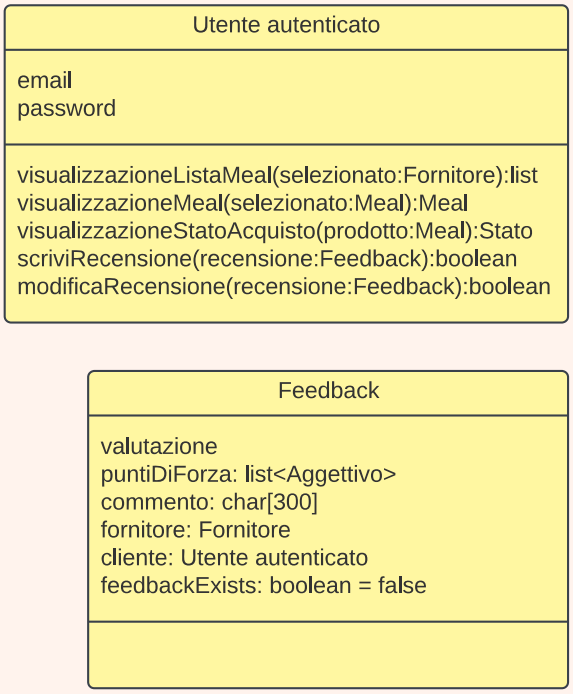
Questa condizione è espressa in OCL con il codice:

```
context Fornitore::consegnaMeal(identificativo:Meal.id):
```

pre: Acquisto.stato = accettato
post: Acquisto.stato = consegnato

Scrittura feedback

Sarà possibile scrivere un feedback a uno specifico fornitore esclusivamente se non è già stata lasciata una recensione in precedenza. Una volta salvato un feedback a uno specifico fornitore, non sarà possibile lasciarne un'altro ma solo modificare quello precedente.



Questa condizione è espressa in OCL con il codice:

Context UtenteAutenticato::scriviRecensione(recensione:Feedback):
Pre: Feedback.feedbackExists = false
Post: Feedback.feedbackExists = true