Dipartimento di Ingegneria e Scienza dell'Informazione
Anno formativo 2022/2023

Percorso di studio: Scienze e Tecnologie Informatiche
Attività didattica: Ingegneria del Software

Progetto: "All About Trento"
Titolo del documento: Documento di Architettura
Gruppo: T07

Allievi: Manuel Vettori, Emanuele Nardelli, Zappacosta Lorenzo

INDICE

SCOPO DEL DOCUMENTO

Il presente documento riporta la realizzazione dell'architettura del progetto "All About Trento" usando diagrammi delle classi in Unified Modelling Language (UML) e codice in Object Constraint Language(OCL).
Nel precedente documento è stato presentato il diagramma degli use case, il diagramma di contesto e quello dei componenti.
Ora, tenendo conto di questa progettazione, viene definita l'architettura del sistema dettagliando da un lato le classi che dovranno essere implementate a livello di codice e dall'altro la logica che regola il comportamento del software.
Le classi vengono rappresentate tramite un diagramma delle classi in linguaggio UML. La logica viene scritta in OCL perché tali concetti non sono esprimibili in nessun altro modo formale.

1. DIAGRAMMA DELLE CLASSI

Nel presente capitolo vengono presentate le classi previste nell'ambito del progetto. Ogni componente presente nel diagramma dei componenti diventa una o più classi. Tutte le classi individuate sono caratterizzate da un nome, una lista di attributi che identificano i dati gestiti dalla classe e una lista di metodi che definiscono le operazioni previste all' interno della classe.
Ogni classe può essere anche associata ad altre classi e tramite questa associazione, è possibile fornire informazioni su come le classi si relazionano tra loro.
Nella parte sottostante sono rappresentate le classi che sono state individuate a partire dai diagrammi di contesto e dei componenti. In questo processo abbiamo massimizzato la coesione e cercato di minimizzare l'accoppiamento tra classi.

1.1 TIPOLOGIE DI UTENTI

Analizzando il diagramma di contesto realizzato per il progetto "All About Trento" all'interno del documento D2, notiamo la presenza di ben 3 attori nello specifico: l'**utente non autenticato**, l'**utente autenticato** e il **gestore**.
L'"utente non autenticato" è colui che può accedere al sito usufruendo solamente delle funzionalità basi che il sistema offre, avendo la possibilità di registrarsi quando vuole.
L'"utente autenticato" è colui che, dopo aver effettuato l'accesso al sito, si autentica tramite login e accede a più funzionalità interattive che l'"utente non autenticato" non può utilizzare.
Infine, il "gestore" è colui che si occupa della gestione dell'intero sistema, garantendo il corretto funzionamento del servizio.

Essendo che, questi 3 attori hanno molto in comune tra di loro, sono state individuate le classi `Utente`, `Utente Autenticato` e `Gestore`, dove la classe `Utente` possiede le funzioni e gli attributi in comune e collega le altre 2 tramite una generalizzazione.
Le altre due classi, oltre ad ereditare le funzionalità e gli attributi dalla classe `Utente`, hanno delle funzioni specifiche che determinano le varie azioni che quel tipo di utente può fare.

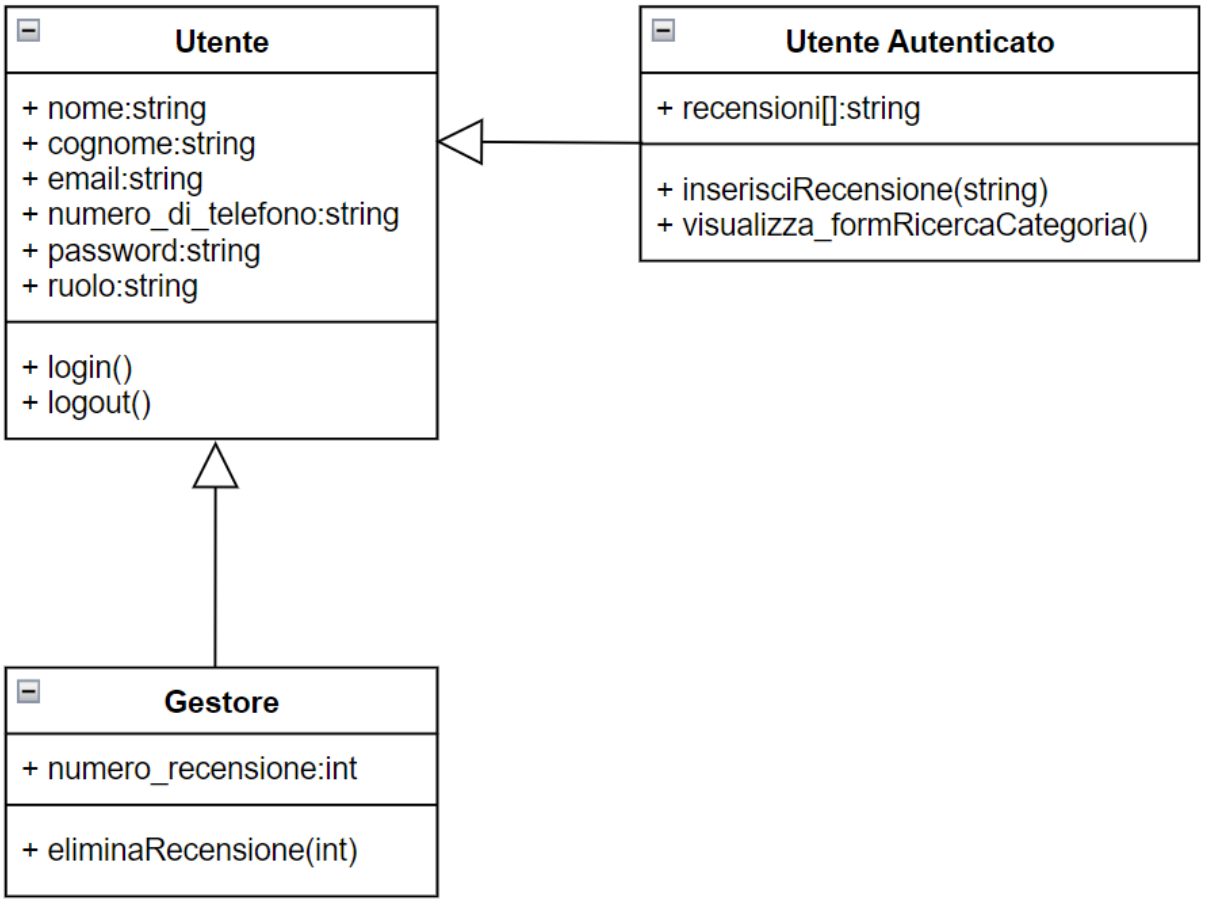

Figura 1. Classi per 'Tipologie di Utenti'

1.2 GESTIONE ACCESSO

Dal diagramma dei componenti, viene analizzato inizialmente il componente "**Accesso al sito**".
Il componente identifica come viene effettuato l'accesso al sito web e a tutti i vari servizi offerti dal sistema.
E' stata, dunque, identificata una classe `accesso`, che si interfaccerà con il sistema permettendone, appunto, l'accesso.
All'interno della classe `accesso` è stata aggiunta la funzionalità di accesso al sito tramite il **sistema esterno QR Code**, analizzato nello specifico nel

diagramma di contesto e utilizzato dal componente “**Accesso al sito**” come interfaccia nel diagramma dei componenti.

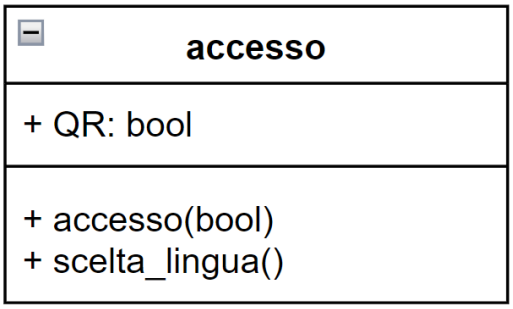

Figura 2. Classe per ‘Gestione accesso’

1.3 FORM PER LA REGISTRAZIONE

Analizzando successivamente il diagramma dei componenti, viene definito il componente “**Pagina Registrazione**”.
Da questo componente è stata identificata la classe `formRegistrazione` che gestisce l’intera procedura che l’utente effettua per registrarsi al sistema.
La classe è associata alla classe `Utente`, in quanto è “ l’utente non autenticato” che può decidere di effettuare/compilare la registrazione al sito.

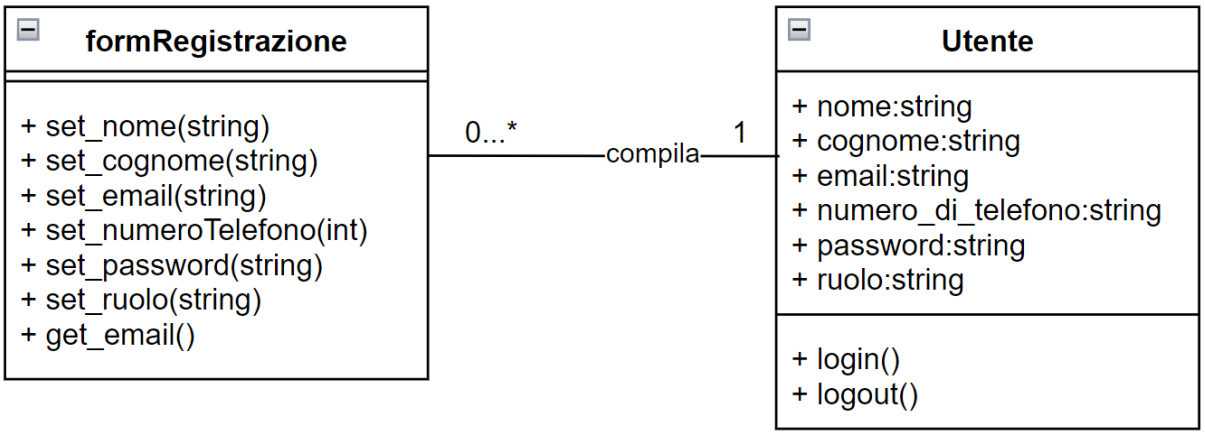

Figura 3. Classi per ‘Form per la registrazione’

1.4 PAGINA RICERCA PER CATEGORIA

Dal componente “**Pagina Ricerca per Categoria**” del diagramma dei componenti, identifichiamo la classe `formRicercaCategoria`.
All’interno della classe vengono specificate le funzionalità e i parametri che vengono utilizzati per gestire questa determinata procedura. Grazie a questa procedura possiamo gestire la ricerca dei vari POI in base alla loro categoria, parametro che viene richiesto all’utente..
Anche questa classe è associata con la classe `Utente`.

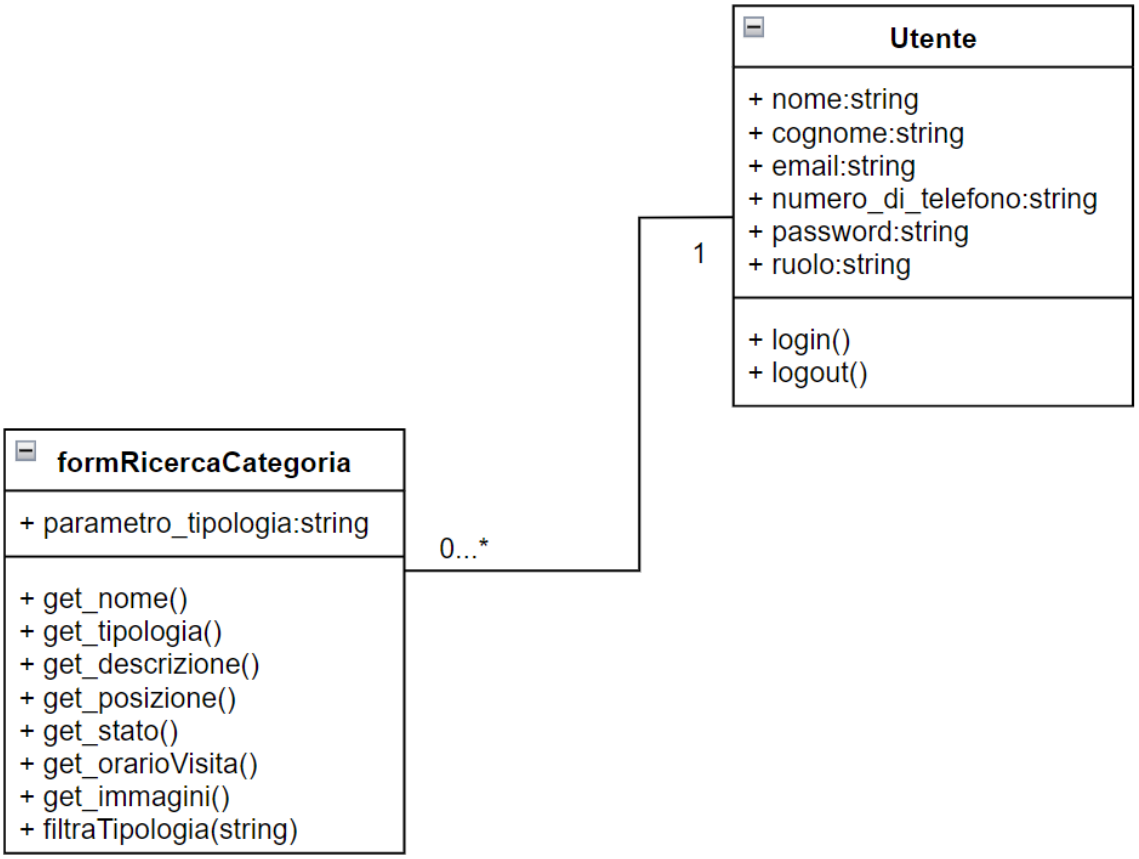

Figura 4. Classi per 'Pagina Ricerca per Categoria'

1.5 PAGINA PERCORSI PRESTABILITI

Viene identificata la classe `formPercorsiPrestabiliti` dal componente "**Pagina Percorsi Prestabiliti**".

La classe specifica le funzionalità per la ricerca, all'interno del sistema, di un percorso creato appositamente per avere un servizio più interattivo per l'utente.

Questa classe è associata con la classe `Utente Autenticato`, perchè il componente "**Pagina Percorsi Prestabiliti**" è accessibile soltanto se l'utente si autentica, ovvero se effettua il login dopo essersi registrato al sito.

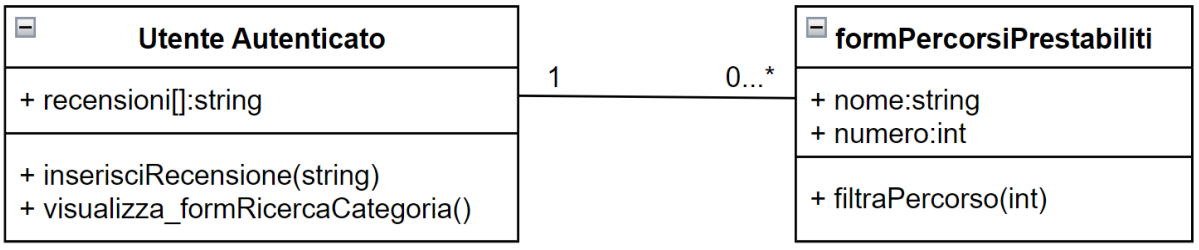

Figura 5. Classi per 'Pagina Percorsi Prestabiliti'

1.6 GESTIONE DEL PROFILO PERSONALE

Dal diagramma dei componenti, analizziamo il componente “**Visualizzazione Profilo Personale**”.

Questo componente permette all’”utente autenticato” di visualizzare i propri dati personali e, eventualmente, modificarli.

Per questo , all’interno del diagramma delle classi, viene definita la classe `formProfiloPersonale` che gestirà le funzionalità di visualizzazione e modifica dei dati.

Essendo che a queste funzionalità può accedere soltanto l’”utente autenticato”, la classe è associata con `Utente Autenticato`.

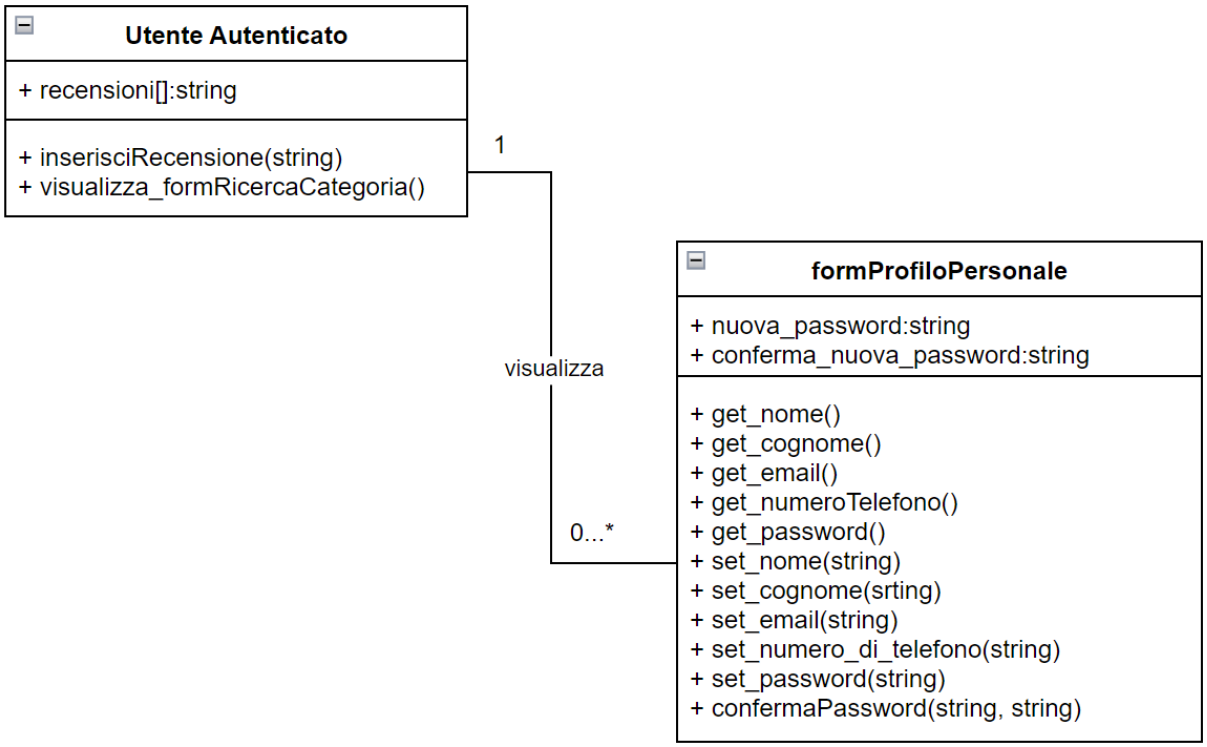

Figura 6. Classi per ‘Gestione del Profilo Personale’

1.7 RECENSIONI

Viene identificata la classe `recensione`.

Questa classe identifica ogni recensione effettuata dall’”utente autenticato”, infatti è associata proprio alla classe `Utente Autenticato`.

Ogni recensione, inoltre, viene gestita dall'utente “gestore”, anche se non è associata alla sua classe.

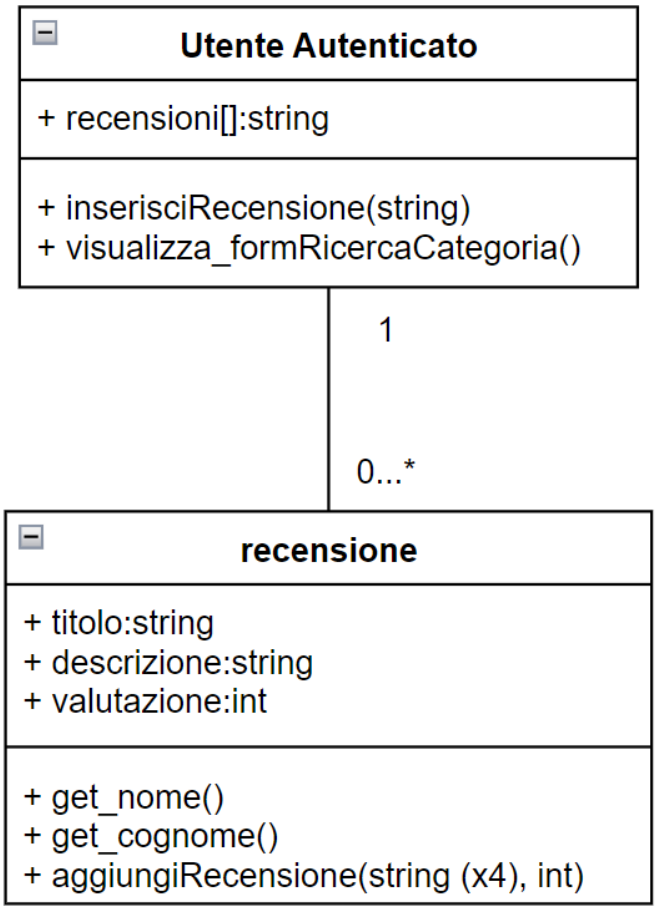

Figura 7. Classi per 'Recensioni'

1.8 PUNTI D'INTERESSE (POI)

L'intero progetto si basa sul divulgare le informazioni per i turisti riguardanti la città di Trento.
Perciò, viene definita una classe `POI`.
Questa classe specifica in particolare gli attributi di ogni punto d'interesse della città, e poi definisce le funzioni che il sistema utilizza per poter divulgare tutte le informazioni ad ogni servizio offerto.

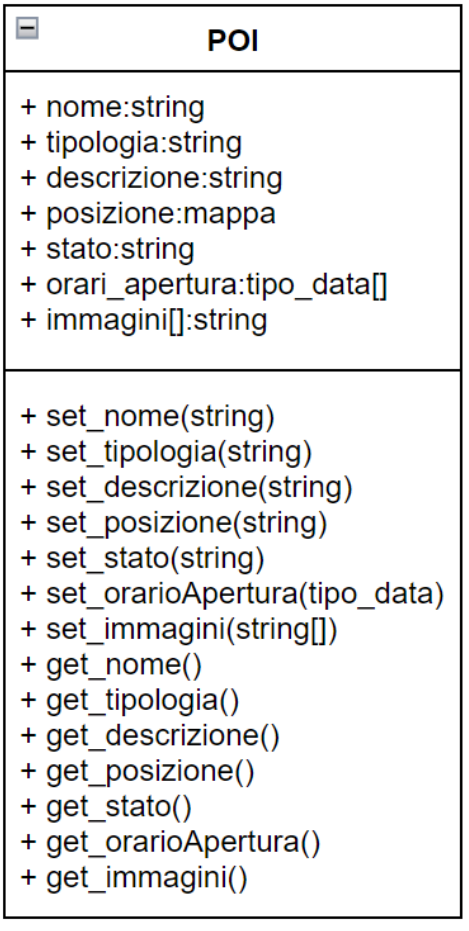

Figura 8. Classe per 'Punti d'Interesse (POI)'

1.9 GESTIONE INSERIMENTO POI

L'ultimo componente da analizzare all'interno del diagramma è "**Aggiornamento e Inserimento POI**".

Essendo che, tramite alcune funzionalità offerte dalla classe `POI` è possibile già aggiornare/modificare i POI, viene specificata la classe `formInserisciPOI`.

Questa classe definisce l'intera procedura per l'inserimento dei POI all'interno del sistema.

Oltre ad essere associata alla classe `POI` per via delle funzionalità offerte, questa classe è associata anche alla classe `Gestore`, in quanto è proprio il "gestore" che effettua questa procedura.

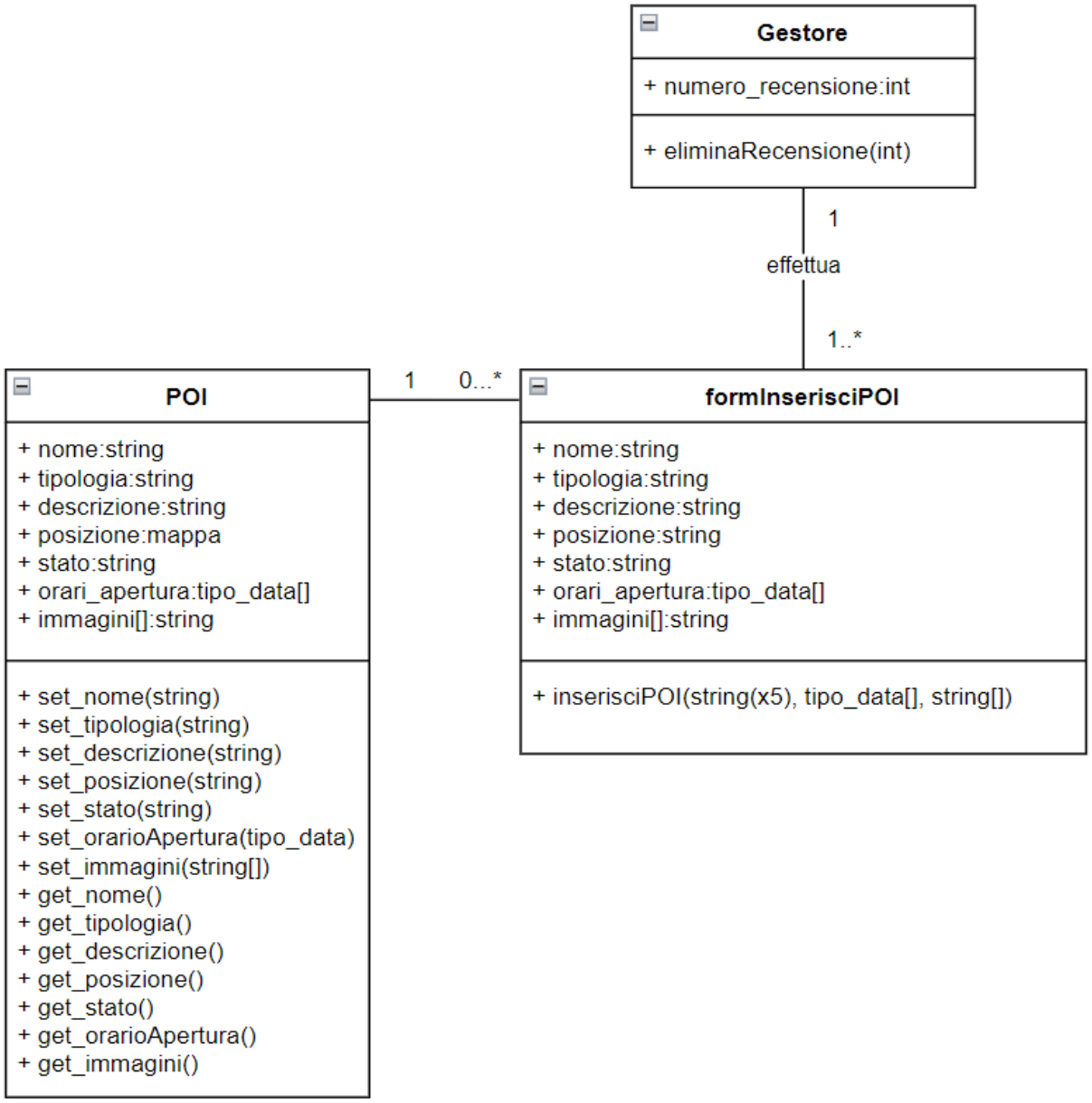

Figura 9. Classi per 'Gestione Inserimento POI'

1.10 DIAGRAMMA DELLE CLASSI COMPLESSIVO

Riportiamo di seguito il diagramma delle classi con tutte le classi fino ad ora presentate.

All'interno del diagramma è possibile identificare alcune classi ausiliarie, come ad esempio `tipo_data` o `mappa`.

Queste tipologie di classi ci permettono di descrivere eventuali tipi di dati strutturati usati presenti negli attributi delle classi.

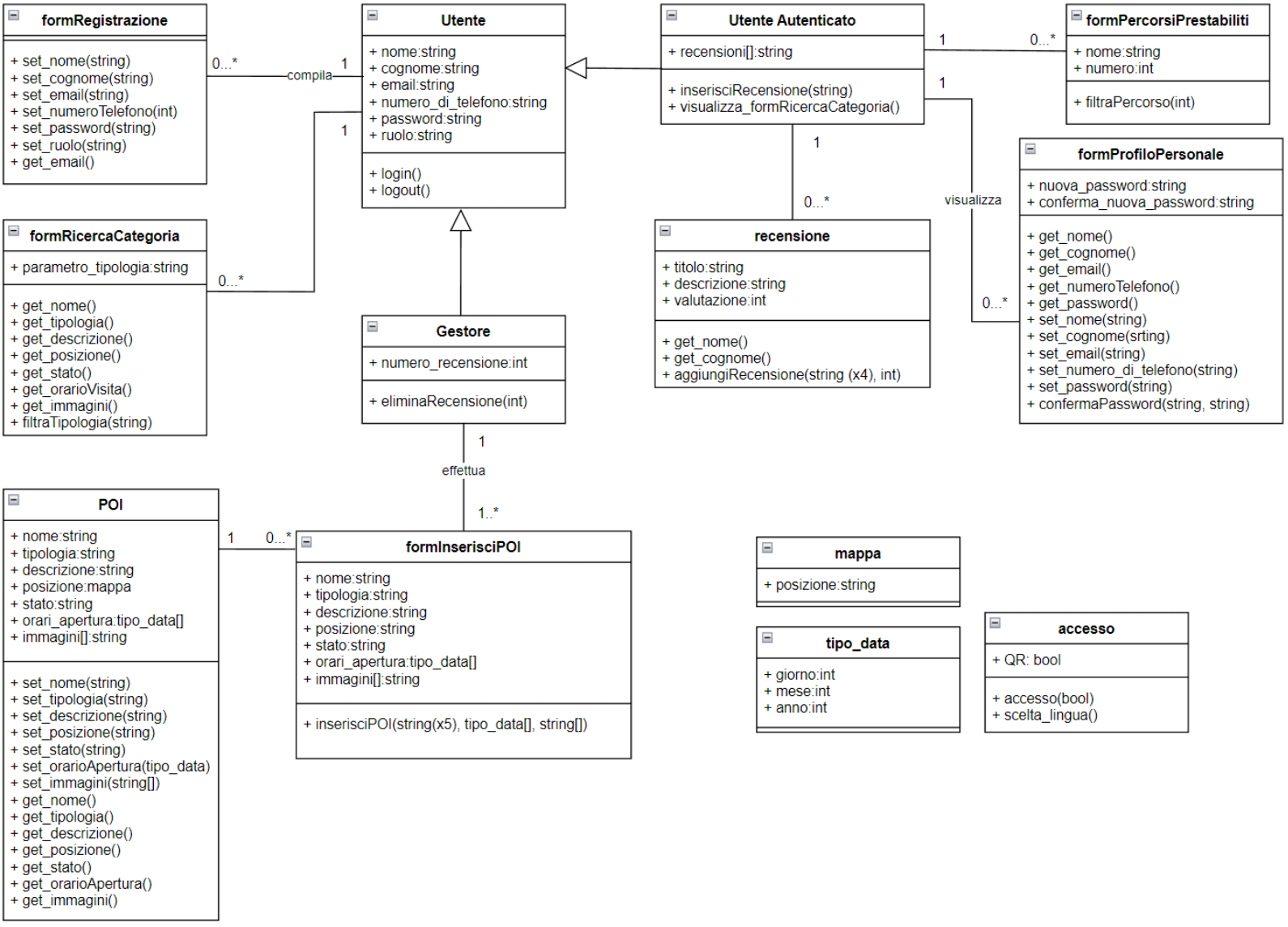

Figura 10. Diagramma delle classi complessivo

2. CODICE IN OBJECT CONSTRAINT LANGUAGE (OCL)

In questo capitolo è descritta in modo formale la logica prevista nell'ambito di alcune operazioni di alcune classi.
Tale logica viene descritta in Object Constraint Language (OCL) perché tali concetti non sono esprimibili in nessun altro modo formale nel contesto di UML.

2.1 ACCESSO AL SITO

La procedura per l'accesso al sito web è gestita dal metodo `accesso(bool)` che si trova all'interno della classe `accesso`:

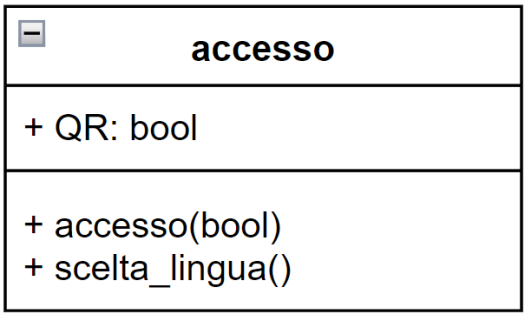

Figura 11. Classe per 'Accesso al sito'

L'esecuzione di questo metodo comporta una variazione del valore dell'attributo `QR`, perciò la funzione viene espressa in OCL attraverso una postcondizione con questo codice:

```
context accesso::accesso(bool)
post: (accesso.QR=true) OR (accesso.QR=false)
```

2.2 GESTIONE RICERCA PER CATEGORIA

All'interno della classe `Utente Autenticato` viene definito il metodo `visualizza_formRicercaCategoria`, che permette la visualizzazione della funzionalità di ricerca in base alla categoria messa a disposizione anche per l'"utente non autenticato":

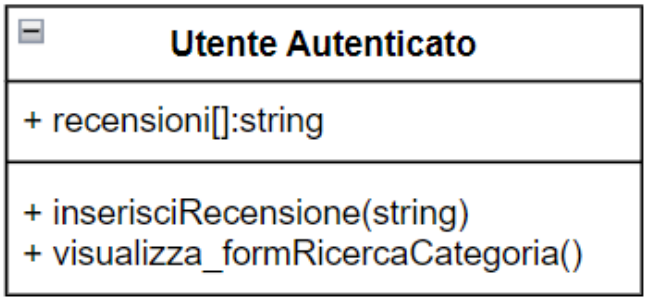

Figura 12. Classe per 'Gestione Ricerca per Categoria'

la funzione viene espressa in OCL tramite una precondizione nel seguente modo:

```
context UtenteAutenticato::visualizza_formRicercaCategoria()
pre: ((Utente.email != NULL)=true) AND ((Utente.password != NULL)=true)
```

2.3 GESTIONE INSERIMENTO RECENSIONE

Sempre all'interno della classe `Utente Autenticato` viene definito un'altro metodo chiamato `inserisciRecensione(string)`, che permette solamente all'"utente autenticato" di inserire recensioni riguardante il sito o i vari POI (vedi ***Figura 12***), e quindi la funzione viene espressa in OCL tramite una precondizione nel seguente modo:

```
context Utente Autenticato::inserisciRecensione(string[])
pre: ((Utente.email != NULL)=true)
AND ((Utente.password != NULL)=true) AND ((Utente Autenticato.recensioni != NULL)=true)
```

2.4 CONTROLLO RICERCA PER CATEGORIA

Il metodo `visualizza_formRicercaCategoria`, che si trova all'interno della classe `Utente Autenticato` (vedi ***Figura 12***), necessita di alcuni controlli in precondizione, perciò realizziamo il seguente codice in OCL:

```
context Utente Autenticato::visualizza_formRicercaCategoria()
pre: (Utente.ruolo = "UtenteAutenticato")
```

2.5 CONTROLLO INSERIMENTO RECENSIONE

Anche il metodo `inserisciRecensione(string)`, che si trova sempre all'interno della classe `Utente Autenticato` (vedi ***Figura 12***), necessita di alcuni controlli in precondizione come nel punto **2.4**, perciò scriviamo il seguente codice OCL.

```
context Utente Autenticato::inserisciRecensione(string)
pre: (Utente.ruolo = "UtenteAutenticato")
```

2.6 PROCEDURA PASSWORD PROFILO PERSONALE

Dalla classe `formProfiloPersonale` troviamo il metodo `confermaPassword(string, string)`:

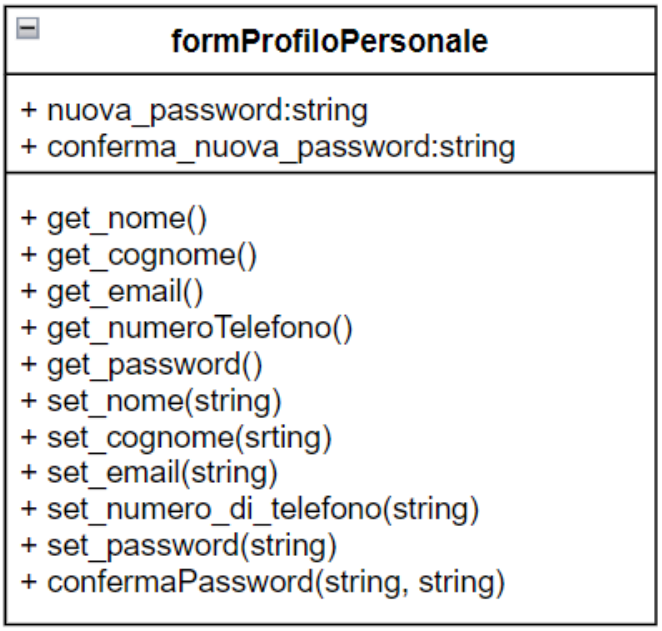

Figura 13. Classe per 'Procedura Password Profilo Personale'

la funzionalità permette di controllare l'inserimento corretto di una nuova password per un "utente autenticato".
Viene definito il codice in OCL in pre e postcondizione:

```
context formProfiloPersonale::confermaPassword(string, string)
pre: (formProfiloPersonale.nuova_password =formProfiloPersonale.conferma_nuova_password)
post: (Utente.password = formProfiloPersonale.nuova_password)
```

2.7 CONTROLLO ELIMINA RECENSIONI

La classe `Gestore` possiede il metodo `eliminaRecensione(int)`.
Tale metodo richiede un controllo, effettuato in OCL tramite una precondizione:

```
context Gestore::eliminaRecensione(int)
pre: (Utente.ruolo = "Gestore")
```

3. DIAGRAMMA DELLE CLASSI CON CODICE OCL

Riportiamo, infine, il diagramma delle classi con tutte le classi fino ad ora presentate ed il codice OCL individuato.

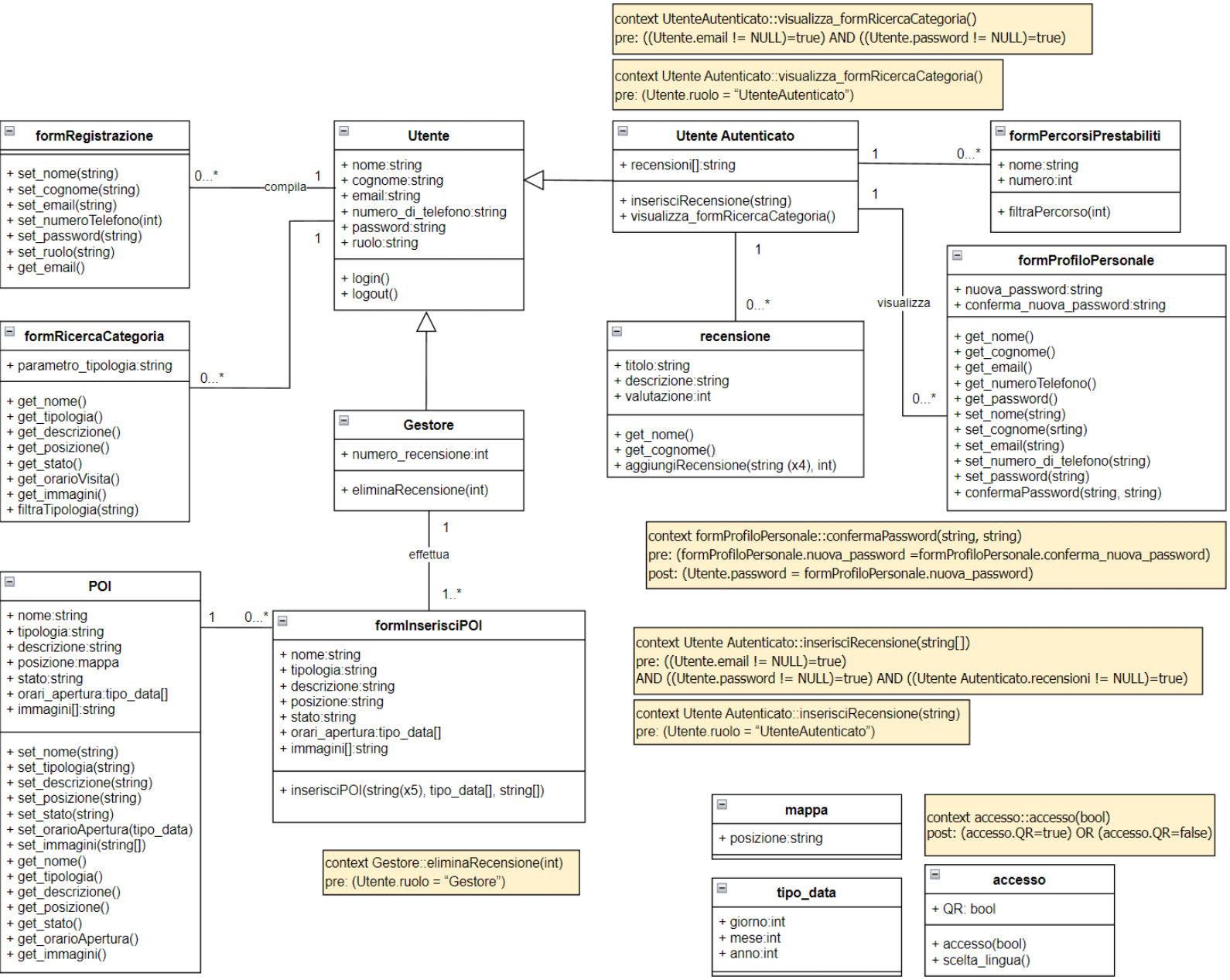

Figura 14. Diagramma delle classi con codice OCL