Dipartimento di Ingegneria e Scienza dell'Informazione

Progetto:

Progetto ingegneria del software: farmacia online e gestione dell'assunzione dei farmaci

Titolo del documento:

Diagramma delle Classi e OCL

Gruppo

Document Info

Doc. Name	*D3-FarmaciaCGZ*	Doc. Number	SR3
Description	Diagramma della classi del sistema comprensivo di OCL		

1. Scopo Del Documento

Nel presente documento viene definita l’architettura del sistema utilizzando il Class Diagram UML e codice in OCL (Object Constraint Language).

2. Diagramma Delle Classi

Nel presente capitolo vengono rappresentate le classi previste nell'ambito del progetto.

1. Utente

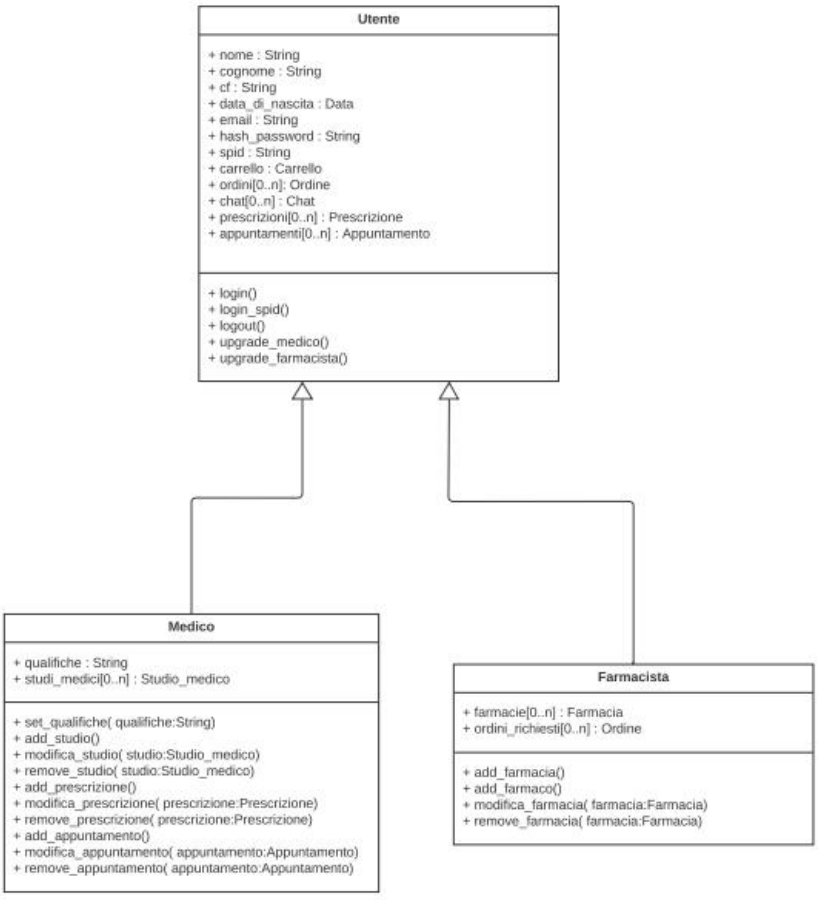
È l'utente autenticato che ha la possibilità di ricercare farmacie e farmaci per comprarli, studi medici per prenotare appuntamenti e avviare chat con I medici.

2. Medico

Utente autenticato con spid che ha la possibilità di aggiungere uno studio medico per renderlo visibile agli altri utenti, creare prescrizioni per i sui pazienti, programmarne l'assunzione dei farmaci e gestirne le visite.

3. Farmacista

Utente autenticato con spid che utilizza il sistema per rendere la propria farmacia visibile agli utenti e per inserire i farmaci da vendere



4. Registrazione

Avvia la registrazione di un utente

5. Conferma registrazione

Gestisce la conferma della registrazione tramite token inviato via mail

6. Autenticazione

Permette all'utente base di autenticarsi, in modo da avere accesso a quasi tutte le funzioni a lui disponibili

7. Autenticazione_spid

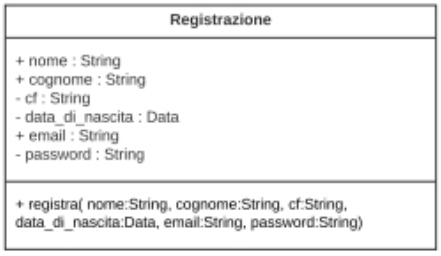
Interagisce con il sistema esterno di Spid per permette a medici e farmacisti di effettuare l'upgrade, in modo da poter accedere alle loro funzionalita riservate, e agli utenti base di acquistare i farmaci su prescrizione di cui hanno una prescrizione

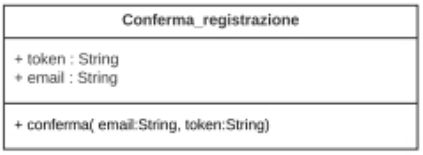
8. Avvia_reset_password

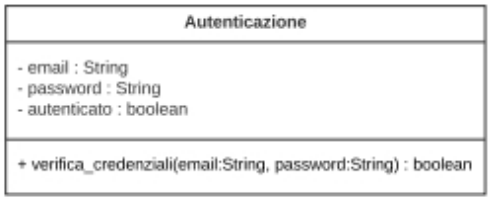
Si occupa di avviare il reset delle password inviando una mail ad un indirizzo registrato

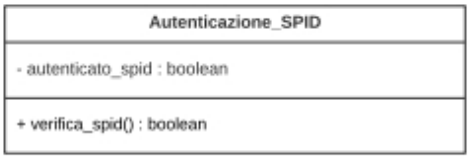
9. reset_password

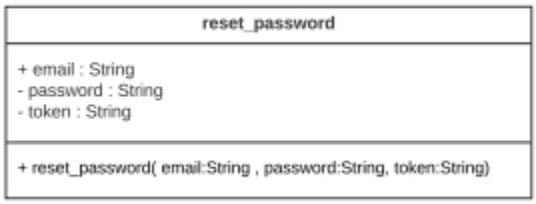
Permette all'utente di resettare la propria password accedendo al link ricevuto via mail

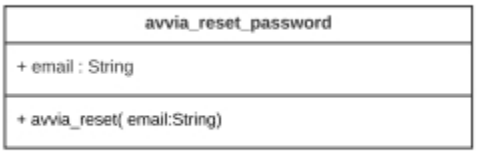












10.Data

Giorno, mese e anno. Utile per interagire con calendar.

11.Ora

Ore, minuti e secondi. Utile per interagire con calendar.

12.Slot_orario

Data e ora, alla base per creare Slot_farmaco e Slor_visita

13.Slot_farmaco

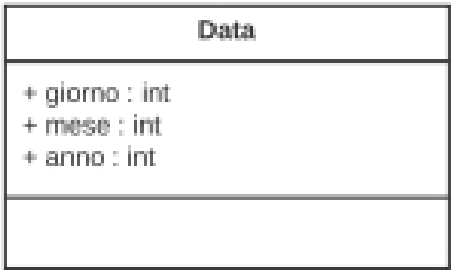
Utile all'interfaccia con google calendar per salvare gli orari di assunzione dei farmaci

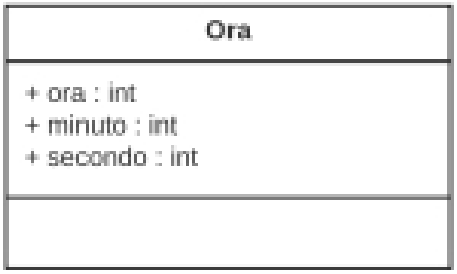
14.Slot_visita

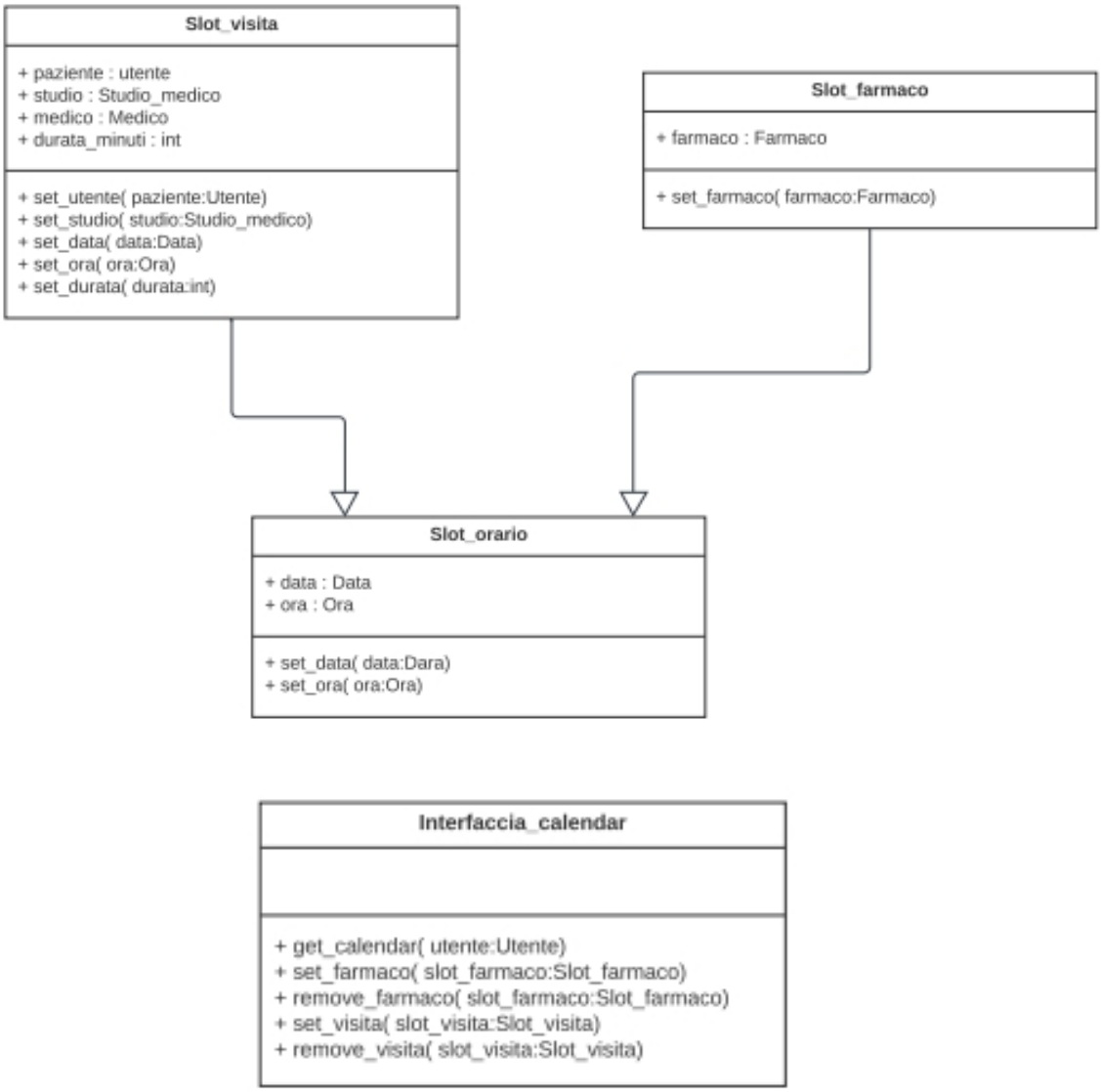
Utile all'interfaccia con google calendar per salvare gli orari delle visite disponibili e prenotati.

15.Interfaccia_calendar

Interagisce con google calendar per ricavare il calendario di un utente, al fine di programmare l'assunzione dei farmaci i programmare le visite dei medici.





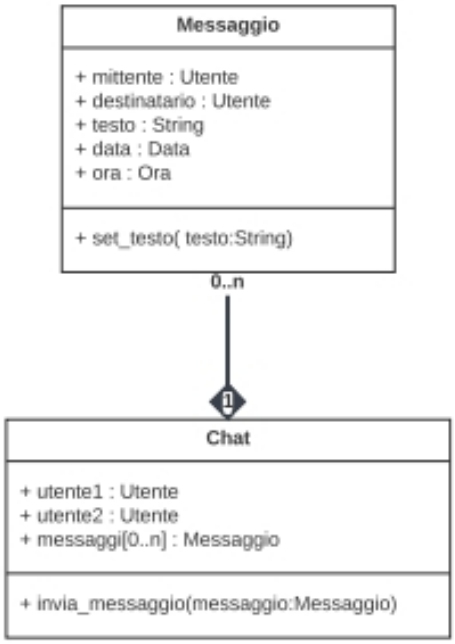


16.Messaggio

Rappresenta il singolo messaggio presente in una chat

17.Chat

Necessaria per la comunicazione tra utente e medico, permette di inviare messaggi. Può essere avviata sia dal paziente che dal medico



18.Farmacia

Rappresenta una farmacia creata da un farmacista. Permette all'utente di trovarne la posizione, e al farmacista di aggiungere farmaci in vendita

19.Farmaco

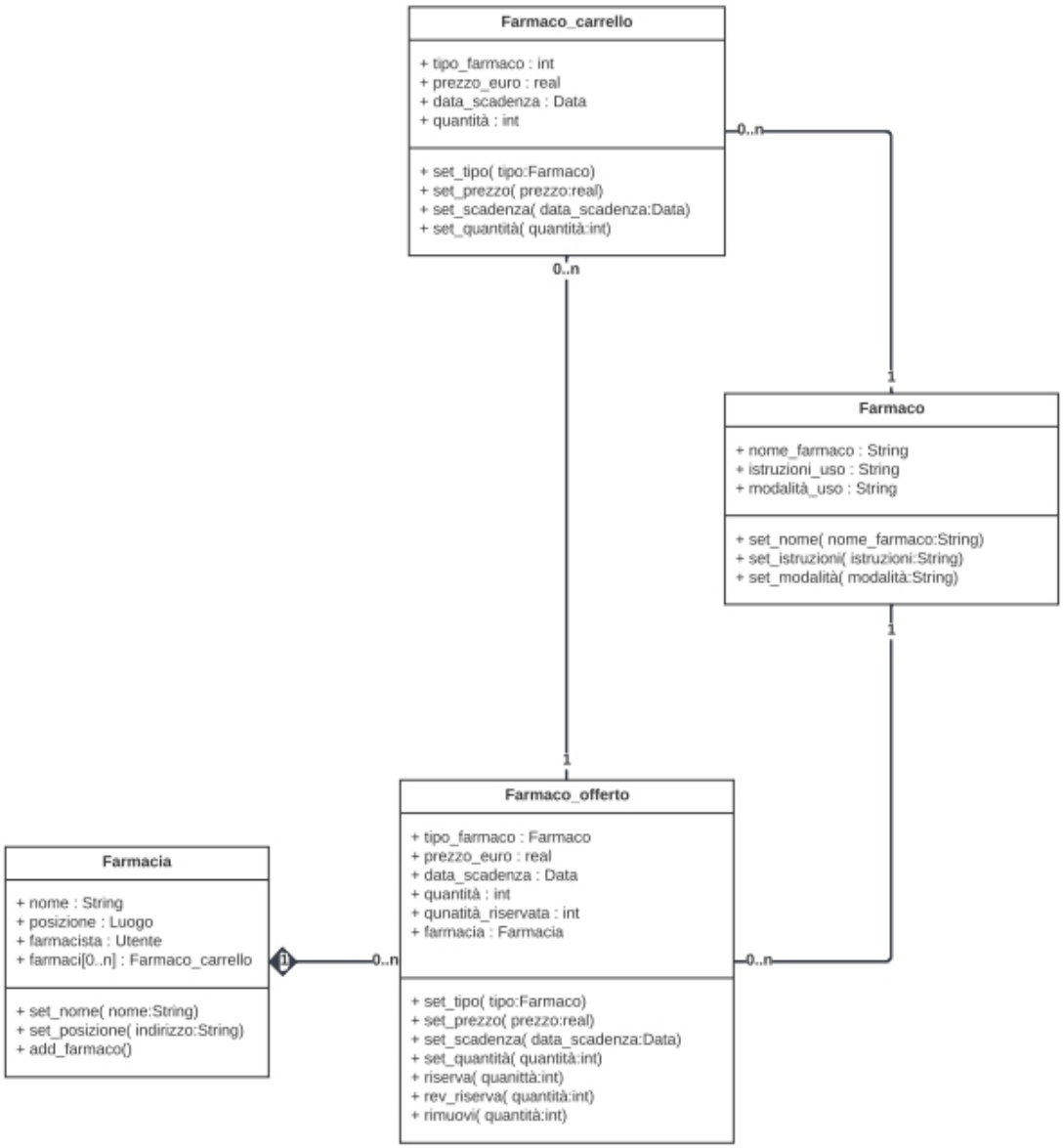
Rappresenta una tipologia di farmaco, utile per rendere disponibili tutte le informazioni senza che vengano inserite ogni volta da tutti i farmacisti

20.Farmaco_offerto

Rappresenta un lotto di uno specifico farmaco, tutti con la stessa scadenza, offerti da una farmacia. Vengono salvati a database e visualizzati dall'utente durante la ricerca

21.Farmaco_carrelo

Rappresenta il singolo farmaco salvato nel carrello di un utente.

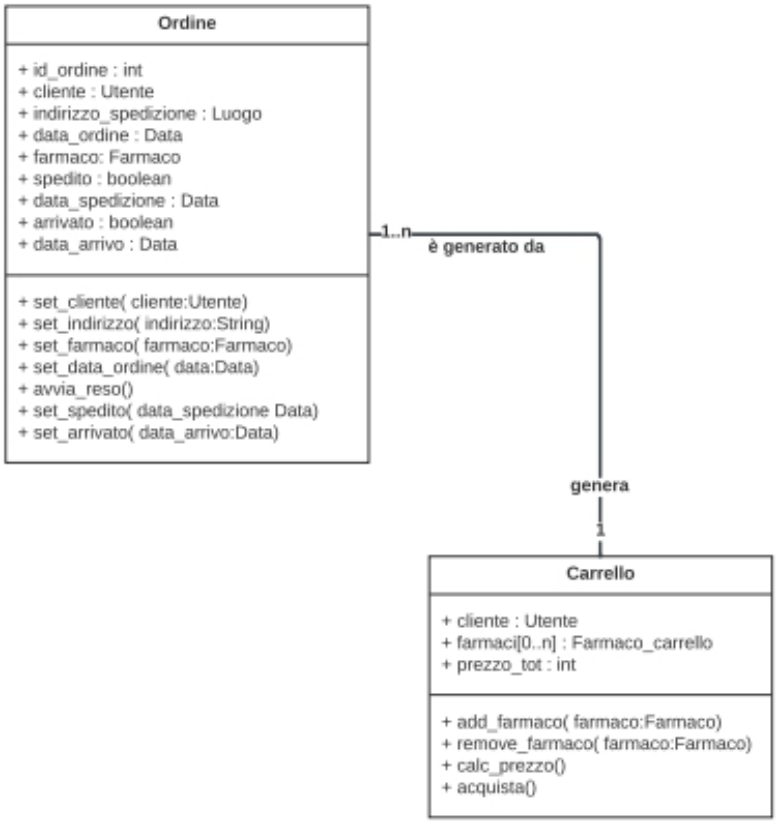


22.Carrello

Associa ad un utente una lista di farmaci che può acquistare.

23.Ordine

Generato dal carrello una volta avvenuto l’acquisto, rappresenta gli oggetti nel database.

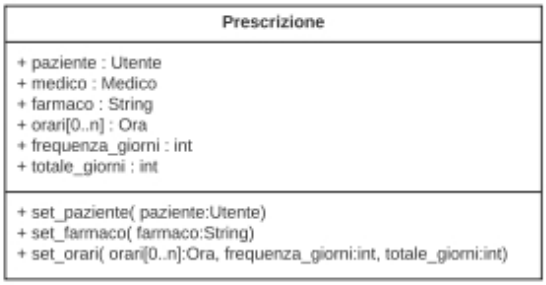


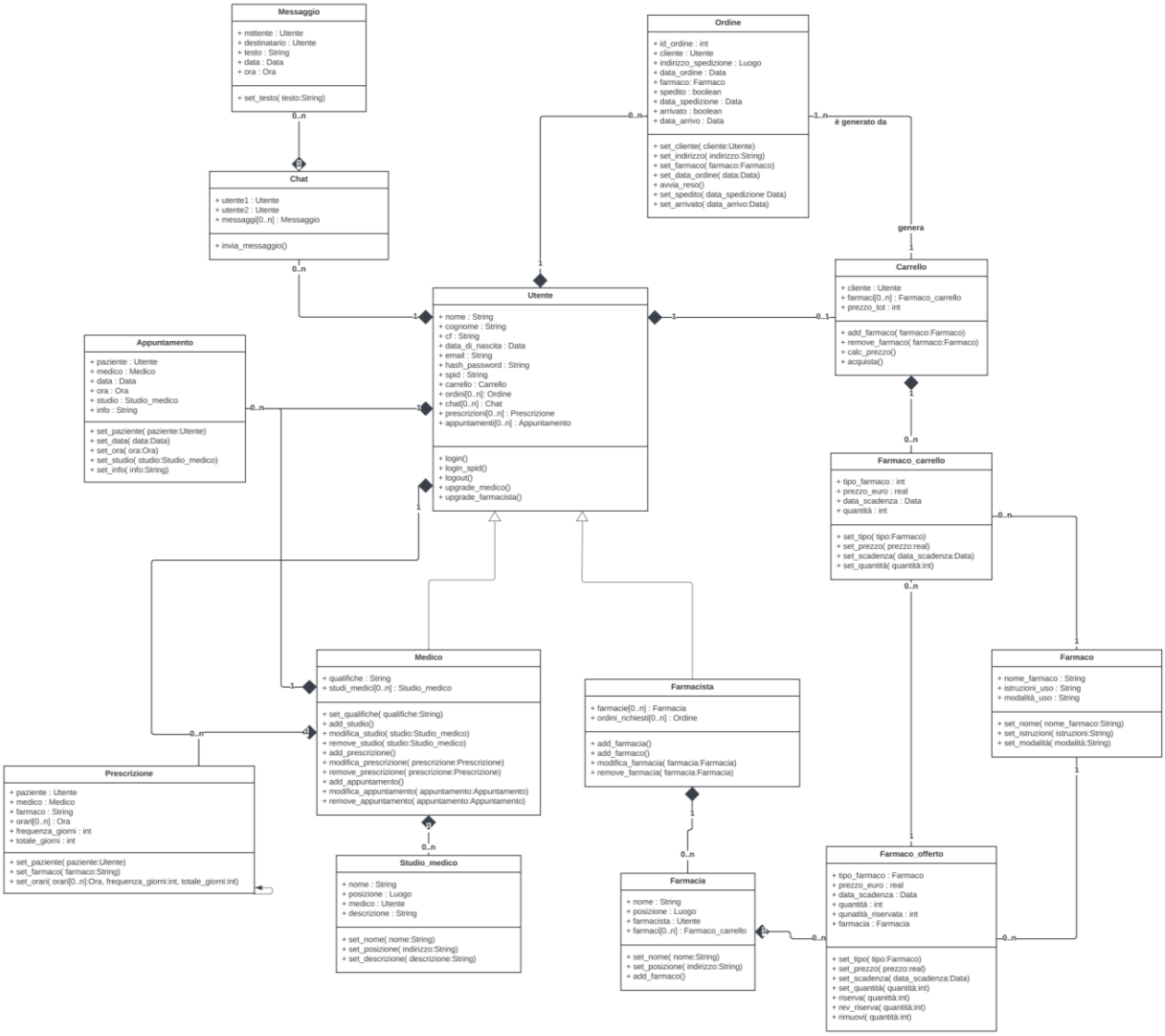
24.Studio medico

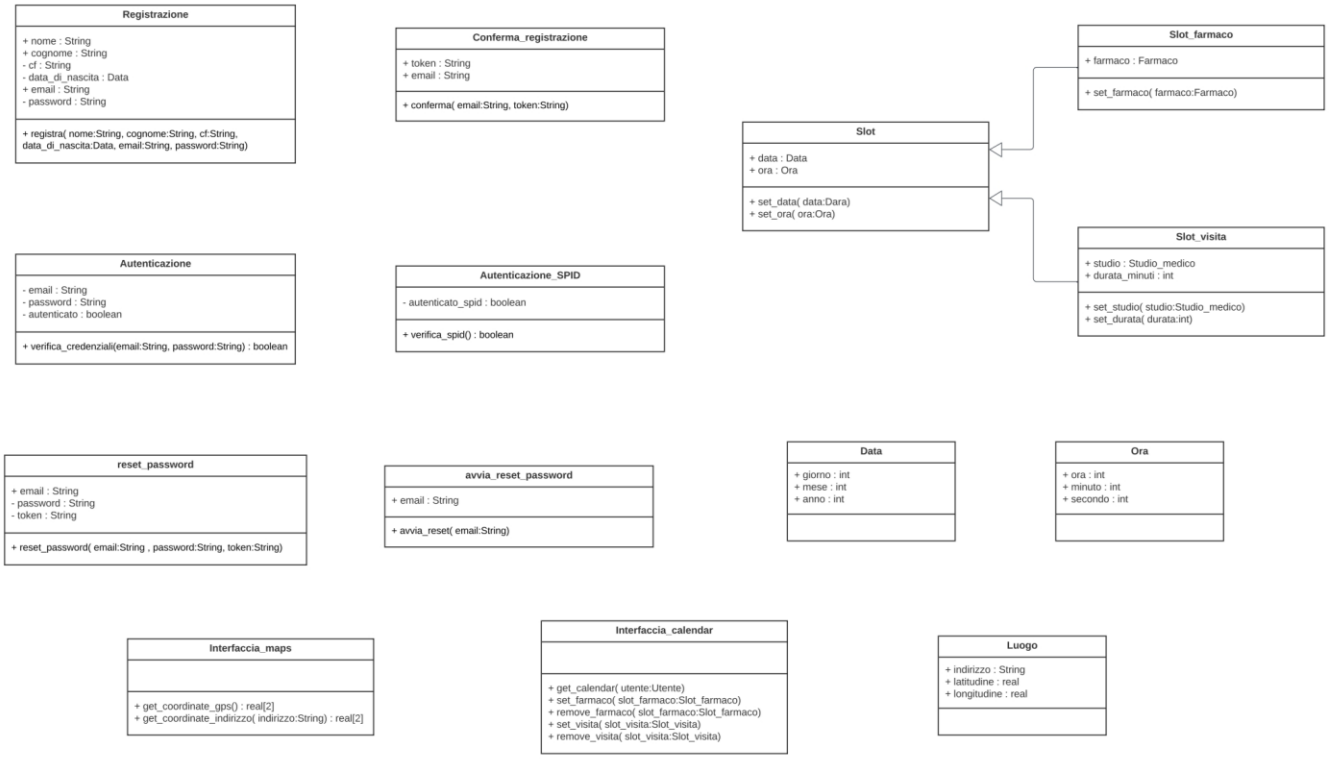
Rappresenta lo studio medico salvato a database. Può essere creato da un medico per permettere agli utenti di essere trovato e iniziare una chat

25.Prescrizione

Rappresenta sia la prescrizione fatta da un medico ad un paziente che la schedule di assunzione dei farmaci. Può essere creata solo da un medico




Messaggio
+ mittente : Utente
+ destinatario : Utente
+ testo : String
+ data : Data
+ ora : Ora
+ set_testo(testo:String)
0..n
1
Chat
+ utente1 : Utente
+ utente2 : Utente
+ messaggi[0..n] : Messaggio
+ invia_messaggio()
0..n
Ordine
+ id_ordine : int
+ cliente : Utente
+ indirizzo_spedizione : Luogo
+ data_ordine : Data
+ farmaco: Farmaco
+ spedito : boolean
+ data_spedizione : Data
+ arrivato : boolean
+ data_arrivo : Data
+ set_cliente(cliente:Utente)
+ set_indirizzo(indirizzo:String)
+ set_farmaco(farmaco:Farmaco)
+ set_data_ordine(data:Data)
+ avvia_reso()
+ set_spedito(data_spedizione Data)
+ set_arrivato(data_arrivo:Data)
0..n
1..n
è generato da
genera
1
Carrello
+ cliente : Utente
+ farmaci[0..n] : Farmaco_carrello
+ prezzo_tot : int
+ add_farmaco(farmaco:Farmaco)
+ remove_farmaco(farmaco:Farmaco)
+ calc_prezzo()
+ acquista()
1
0..n
Utente
+ nome : String
+ cognome : String
+ cf : String
+ data_di_nascita : Data
+ email : String
+ hash_password : String
+ spid : String
+ carrello : Carrello
+ ordini[0..n]: Ordine
+ chat[0..n] : Chat
+ prescrizioni[0..n] : Prescrizione
+ appuntamenti[0..n] : Appuntamento
+ login()
+ login_spid()
+ logout()
+ upgrade_medico()
+ upgrade_farmacista()
1
0..1
Appuntamento
+ paziente : Utente
+ medico : Medico
+ data : Data
+ ora : Ora
+ studio : Studio_medico
+ info : String
+ set_paziente(paziente:Utente)
+ set_data(data:Data)
+ set_ora(ora:Ora)
+ set_studio(studio:Studio_medico)
+ set_info(info:String)
0..n
1
Farmaco_carrello
+ tipo_farmaco : int
+ prezzo_euro : real
+ data_scadenza : Data
+ quantità : int
+ set_tipo(tipo:Farmaco)
+ set_prezzo(prezzo:real)
+ set_scadenza(data_scadenza:Data)
+ set_quantità(quantità:int)
0..n
0..n
Medico
+ qualifiche : String
+ studi_medici[0..n] : Studio_medico
+ set_qualifiche(qualifiche:String)
+ add_studio()
+ modifica_studio(studio:Studio_medico)
+ remove_studio(studio:Studio_medico)
+ add_prescrizione()
+ modifica_prescrizione(prescrizione:Prescrizione)
+ remove_prescrizione(prescrizione:Prescrizione)
+ add_appuntamento()
+ modifica_appuntamento(appuntamento:Appuntamento)
+ remove_appuntamento(appuntamento:Appuntamento)
1
0..n
Farmacista
+ farmacie[0..n] : Farmacia
+ ordini_richiesti[0..n] : Ordine
+ add_farmacia()
+ add_farmaco()
+ modifica_farmacia(farmacia:Farmacia)
+ remove_farmacia(farmacia:Farmacia)
Farmaco
+ nome_farmaco : String
+ istruzioni_uso : String
+ modalità_uso : String
+ set_nome(nome_farmaco:String)
+ set_istruzioni(istruzioni:String)
+ set_modalità(modalità:String)
1
Prescrizione
+ paziente : Utente
+ medico : Medico
+ farmaco : String
+ orari[0..n] : Ora
+ frequenza_giorni : int
+ totale_giorni : int
+ set_paziente(paziente:Utente)
+ set_farmaco(farmaco:String)
+ set_orari(orari[0..n]:Ora, frequenza_giorni:int, totale_giorni:int)
1
0..n
Studio_medico
+ nome : String
+ posizione : Luogo
+ medico : Utente
+ descrizione : String
+ set_nome(nome:String)
+ set_posizione(indirizzo:String)
+ set_descrizione(descrizione:String)
1
0..n
Farmacia
+ nome : String
+ posizione : Luogo
+ farmacista : Utente
+ farmaci[0..n] : Farmaco_carrello
+ set_nome(nome:String)
+ set_posizione(indirizzo:String)
+ add_farmaco()
1
0..n
Farmaco_offerto
+ tipo_farmaco : Farmaco
+ prezzo_euro : real
+ data_scadenza : Data
+ quantità : int
+ qunatità_riservata : int
+ farmacia : Farmacia
+ set_tipo(tipo:Farmaco)
+ set_prezzo(prezzo:real)
+ set_scadenza(data_scadenza:Data)
+ set_quantità(quantità:int)
+ riserva(quanittà:int)
+ rev_riserva(quantità:int)
+ rimuovi(quantità:int)
1
0..n


Registrazione
+ nome : String
+ cognome : String
- cf : String
- data_di_nascita : Data
+ email : String
- password : String
+ registra(nome:String, cognome:String, cf:String, data_di_nascita:Data, email:String, password:String)
Conferma_registrazione
+ token : String
+ email : String
+ conferma(email:String, token:String)
Slot_farmaco
+ farmaco : Farmaco
+ set_farmaco(farmaco:Farmaco)
Slot
+ data : Data
+ ora : Ora
+ set_data(data:Dara)
+ set_ora(ora:Ora)
Slot_visita
+ studio : Studio_medico
+ durata_minuti : int
+ set_studio(studio:Studio_medico)
+ set_durata(durata:int)
Autenticazione
- email : String
- password : String
- autenticato : boolean
+ verifica_credenziali(email:String, password:String) : boolean
Autenticazione_SPID
- autenticato_spid : boolean
+ verifica_spid() : boolean
reset_password
+ email : String
- password : String
- token : String
+ reset_password(email:String , password:String, token:String)
avvia_reset_password
+ email : String
+ avvia_reset(email:String)
Data
+ giorno : int
+ mese : int
+ anno : int
Ora
+ ora : int
+ minuto : int
+ secondo : int
Interfaccia_maps
+ get_coordinate_gps() : real[2]
+ get_coordinate_indirizzo(indirizzo:String) : real[2]
Interfaccia_calendar
+ get_calendar(utente:Utente)
+ set_farmaco(slot_farmaco:Slot_farmaco)
+ remove_farmaco(slot_farmaco:Slot_farmaco)
+ set_visita(slot_visita:Slot_visita)
+ remove_visita(slot_visita:Slot_visita)
Luogo
+ indirizzo : String
+ latitudine : real
+ longitudine : real

3. OCL

Di seguito alcune OCL utili a definire il comportamento del sistema.

1. Upgrade medico/farmacista

i medici e i farmacisti, per poterlo diventare, devono prima autenticarsi tramite spid. Di conseguenza possiamo scrivere :

context Medico **inv**: self.spid!=Null

context Farmacista **inv**: self.spid!=Null

Oppure possiamo scrivere :

context Utente::upgrade_medico() **pre**: self.spid!=Null

context Utente::upgrade_farmacista() **pre**: self.spid!=Null

2. Farmaci in vendita

Per quanto riguarda I farmaci offerti possiamo dire che:

context Farmaco_offerto **inv**: self.prezzo_euro>0 and self.data_scadenza>time() and self.quantità>=self.quantità_riservata and self.quantità>0 and self.quantità_riservata>=0

context Farmaco_offerto::riserva(quantità_scelta) **inv**: self.quantità_riservata = self.quantita_riservata@pre+self.quantità_scelta

Dove quantità indica la quantita messa a disposizione dal farmacista che non è ancora stata venduta, quantità_riservata in dica la quantità di farmaco che è bloccata perche qualcuno sta procedendo con un acquisto, ma che non è ancora stata definitivamente acquistata, mentre quantità_scelta è la quantità di farmaco che si sta provando a mettere nel carrello

3. Farmaci e carrello

Per I farmaci nel carrello si può dire che:

context Farmaco_carrello **inv**: self.prezzo_euro>0 and
data_scadenza>time() and quantità>0

context carrello **inv**: self.farmaci->size()>=0 and self.prezzo_tot>=0 and
(self.farmaci->size()>0 implies self.prezzo_tot>0) and
(self.prezzo_tot>0 implies self.farmaci->size()>0)

context carrello::acquista()
pre: self.farmaci->size()>0 and self.prezzo_tot>0
post: self.farmaci->size()=0 and self.prezzo_tot=0

Infatti ogni farmaco nel carrello deve avere un prezzo strettamente positivo, non può essere scaduto, e dev'essere in quantità strettamente positiva.

Inoltre sia il numero di farmaci che il prezzo totale del carrello sono >=0, ma se uno non è 0 allora neanche l'altro lo è.

Quando procedo con l'acquisto devo avere almeno un farmaco nel carrello, e quindi un prezzo positivo, e ad acquisto terminato il carrello si dev'essere svuotato

4. Ordini

Sugli ordini possiamo dare delle condizioni di consequenzialità, infatti:

context Ordine **inv**:
(self.data_spedizione!=Null implies self.data_ordine< self.data_spedizione)
and (self.arrivato implies self.spedito)

Cioè che se l'ordine è stato spedito allora questo è stato fatto dopo che è stato creato, e che se l'ordine è arrivato a destinazione allora lo ha fatto dopo essere stato spedito

5. Chat

Sulle chat possiamo dire che:

context chat **inv**: self.messaggi->size()>0

context chat::invia_messaggio(messaggio)
post: self.messaggi->size()=self.messaggi->size()@pre+1

Cioè che se una chat esiste allora ha almeno un messaggio, altrimenti non sarebbe ancora stata creata, e che chiamando invia_messaggio() il numero di messaggi nella chat aumenta di uno