Università degli Studi di Salerno
Corso di ingegneria del Software

HL HealthyLife

HealthyLife

Object Design Document

Versione 1.0

PARTECIPANTI AL PROGETTO

Nome	Matricola
Cataletti Raffaele	0512102372
D'Auria Antonio	0512102582
Nunziata Vincenzo	0512102594
Rotolo Westley	0512103310

Sommario

1.Introduzione

Qui sono riportati i compromessi implementativi fatti da parte degli sviluppatori, le guide linea che essi hanno seguito per le interfacce dei sottosistemi, la decomposizione dei sottosistemi in pacchetti, classi ed interfaccia. L'Object Design Document è usato come riferimento riguardo i dettagli delle interfacce, utile per i team durante le fasi di test.

NOTA: Questo documento si riferisce all'Object Design per lo stato corrente della fase di implementazione.

1.1 Object Design trade-offs

Per ridurre i tempi di rilascio del software è stato concordato che l'implementazione conterrà solo una parte dei sottoinsiemi del progetto. Infatti tutte le funzionalità riguardo alla messaggistica non sono state implementate.

1.1.1 Prestazioni e Costi

Il sistema realizzato utilizza a pieno le potenzialità delle risorse Open Source, abbattendo in maniera significativa i costi. Dato che possiamo dividere il nostro software in vari moduli, possiamo ricercare se è presente del materiale che rispetti i nostri scopi in maniera tale da poter incorporare tale codice nel nostro programma abbattendo anche i tempi di realizzazione e risorse umane. Nonostante quindi i vari risparmi a livello di costi e attingendo ad un budget relativamente basso, possiamo garantire un software che abbia buone prestazioni.

1.1.2 Interfaccia e Tempo di risposta

Dato lo scarso contenuto di materiale all'interno delle varie interfacce, composte per la maggior parte da testo, i tempi di risposta saranno molto rapidi. Ovviamente gli stessi rallenteranno con l'accrescere del database.

1.1.3 Costi di mantenimento

Grazie all'uso di materiale open source e l'utilizzo di GitHub per il codice il sistema può essere facilmente modificato, implementato con nuove funzioni o corretto in presenza di errori. Inoltre dato il posizionamento del sistema su pochi calcolatori avremo un minore costo di manutenzione eventuale.

1.1.4 Interfaccia User-Friendly

L'interfaccia, grazie all'utilizzo delle form e di una impostazione semplice e intuitiva, composta per la maggior parte da testo, permette un uso facile della gestione del sistema permettendo a chiunque di poterlo usare senza avere particolari competenze informatiche.

1.2 Linee guida per la documentazione delle interfacce

Vengono qui elencate le linee guida che gli sviluppatori dovranno seguire durante la stesura del codice.

1.2.1 Stile di programmazione

Lo spazio di identazione delle classi deve essere di un tab.

1.2.2 Naming Convention

Si seguirà lo stile “Gobba di cammello” per nominare classi, oggetti, metodi ecc. Le classi dovranno cominciare con lettera maiuscola necessariamente. Inoltre, i nomi delle classi e dei metodi dovranno fornire informazioni sul loro scopo.

Es. UtenteRegistrato.

I nomi dei metodi devono cominciare con una lettera minuscola.

Es. cercaNome().

I nomi dei metodi per l’accesso e la modifica delle variabili dovranno avere prefisso get (nel caso di accesso) e set(nel caso di modifica) e nel caso il di valori booleani is.

Le variabili dovranno iniziare con lettera minuscola.

Le costanti dovranno essere scritte a caratteri maiuscoli e separate dal carattere underscore nel caso vi si necessiti di spazio.

1.2.3 Documentazione

Ogni classe dovrà avere una breve spiegazione del suo scopo, dopo le istruzioni all’interno dell’header. Devono essere inoltre indicati l’autore della classe e altre informazioni utili utilizzando gli appropriati tag a seconda del linguaggio usato.

La descrizione dei metodi deve apparire prima di ogni dichiarazione di un metodo e deve descriverne lo scopo. Devono essere elencati i parametri del metodo, i valori di ritorno ed eventualmente le eccezioni che possono essere lanciate, utilizzando gli appropriati tag a seconda del linguaggio utilizzate.

Le variabili devono essere documentate solo se il loro scopo non è immediatamente intuibile.

1.3 Acronimi e abbreviazioni

RAD: Requirement Analisys Document

SDD: System Analisys Document

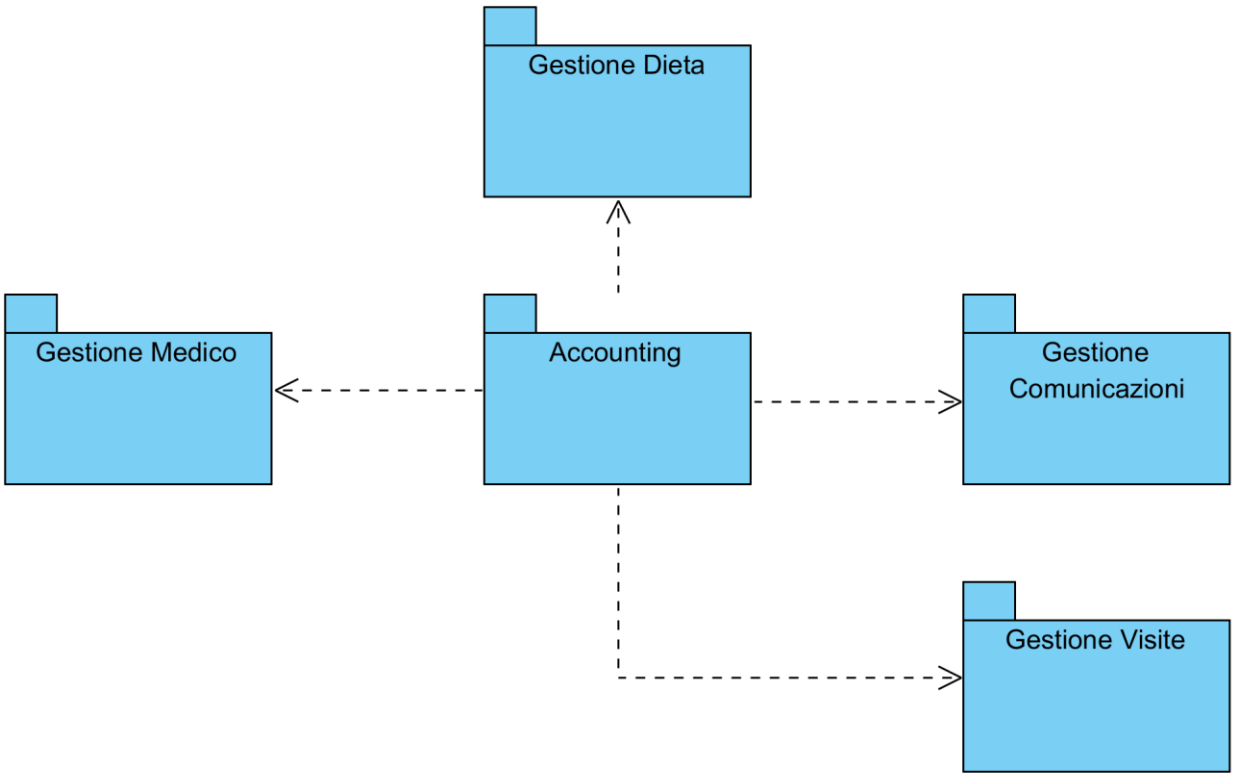
1.4 Riferimenti

RAD e SDD

2.Packages

La suddivisione delle varie classi è stata effettuata nell'SDD. In particolare la suddivisione è stata effettuata in sottoinsiemi.

In particolare le classi sono state racchiuse nei seguenti packages.

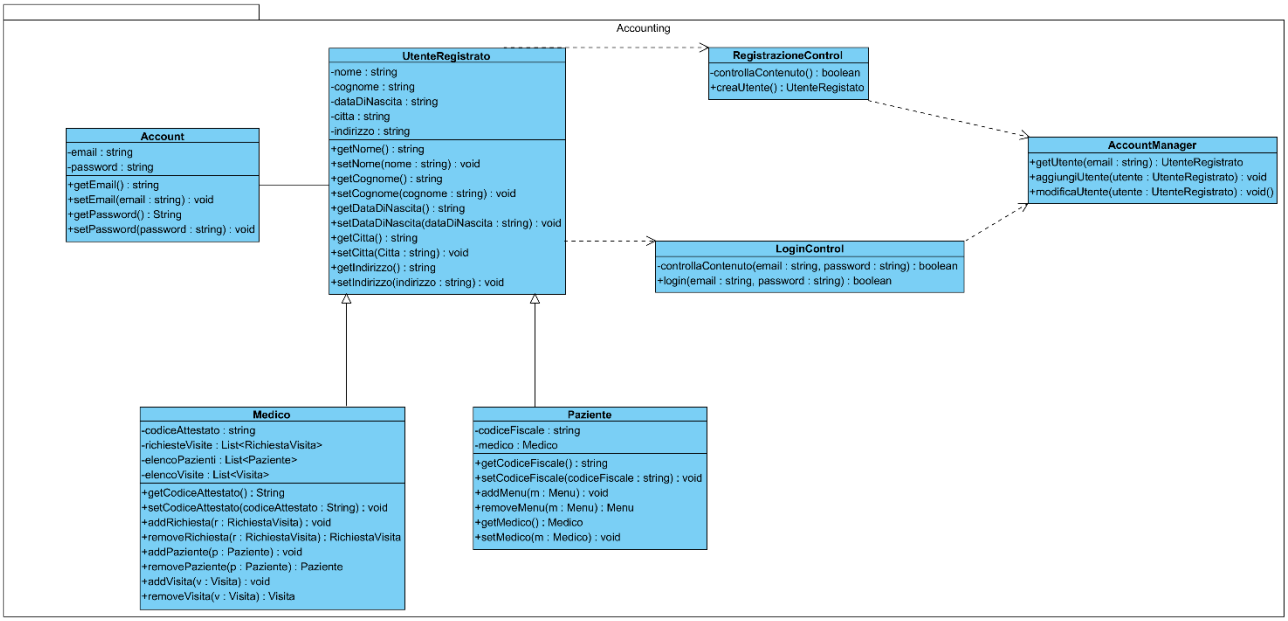


- Accounting: package che racchiude le classi che rappresentano gli utenti (Paziente e Medico) e le funzionalità di registrazione, login, modifica dei dati
- Gestione medico: package che racchiude le funzionalità offerte al paziente per la gestione del medico: aggiunta e cambio medico
- Gestione dieta: package che racchiude le classi che rappresentano i menù della dieta e le funzionalità di aggiunta, modifica e visualizzazione

- Gestione comunicazioni: package che racchiude le classi che rappresentano le comunicazioni tra medico e paziente e le funzionalità di invio e visualizzazione
- Gestione visite: package che racchiude le classi che rappresentano le richieste di visita e la loro gestione.

3 Class Interfaces

3.1 Accounting



3.1.1 Account

Descrizione: Descrive un account del sistema.

Dipendenze:

Attributi			
Nome	**Tipo**	**Accesso**	**Descrizione**
Email	String	private	Email che identifica l'Account
Password	String	private	Password richiesta come credenziale di accesso

Metodi				
Nome	**Parametri**	**Return**	**Accesso**	**Descrizione**
getEmail	void	String	public	Restituisce l'Email dell'account
setEmail	String Email	void	public	Setta l'Email dell'account

getPassword	void	String	public	Restituisce la password dell'account
SetPassword	String Password	void	public	Setta la password dell'account

3.1.2 UtenteRegistrato

Descrizione: Un utente registrato alla piattaforma

Dipendenze: Account

Attributi			
Nome	**Tipo**	**Accesso**	**Descrizione**
Nome	String	private	Nome dell'Utente registrato
Cognome	String	private	Cognome dell'Utente registrato
DataDiNascita	String	private	Data di nascita dell'Utente registrato
Città	String	private	Città dell'Utente registrato
Indirizzo	String	private	Indirizzo dell'Utente registrato

Metodi				
Nome	**Parametri**	**Return**	**Accesso**	**Descrizione**
getAccount	void	Account	public	Restituisce l'Account
setAccount	Account a	void	public	Setta l'Account
getNome	void	String	public	Restituisce il nome
setNome	String Nome	void	public	Setta il nome
getCognome	void	String	public	Restituisce il cognome
setCognome	String Cognome	void	public	Setta il cognome
getDataDiNascita	void	String	public	Restituisce la data di nascita
setDataDiNascita	String Data	void	public	Setta la data di nascita
getCittà	void	String	public	Restituisce la città
setCittà	String Città	void	public	Setta la città
getIndirizzo	void	String	public	Restituisce l'indirizzo
setIndirzzo	String Indirizzo	void	public	Setta l'indirizzo

3.1.3 Medico

Descrizione: Un medico registrato alla piattaforma.

Dipendenze:

Attributi			
Nome	**Tipo**	**Accesso**	**Descrizione**

codiceAttestato	String	private	Codice dell'attestato del medico
richiesteVisite	List <RichiestaVisita>	private	Elenco delle richieste di visita
elencoPazienti	List<Pazienti>	private	Elenco dei pazienti
elencoVisite	List<Visita>	private	Elenco delle visite

Metodi				
Nome	**Parametri**	**Return**	**Accesso**	**Descrizione**
getCodiceAttestato	void	String	public	Restituisce attestato
setCodiceAttestato	String codAtt	void	public	Modifica il codice Attestato
addRichiesta	RichiestaVisita r	void	public	Aggiunge una richiesta
removeRichiesta	RichiestaVisita r	RichiestaVisita	public	Rimuove e restituisce la richiesta
addPaziente	Paziente p	void	public	Aggiunge paziente
removePaziente	Paziente p	Paziente	public	Rimuove e restituisce il paziente
addVisita	Visita v	void	public	Aggiunge visita
removeVisita	Visita v	Visita	public	Rimuove e restituisce visita

3.1.4 Paziente

Descrizione: Un paziente registrato alla piattaforma.

Dipendenze: Menu', Medico

Attributi			
Nome	**Tipo**	**Accesso**	**Descrizione**
codiceFiscale	String	private	Codice Fiscale di un paziente
listMenu	List<Menu>	private	Lista dei menù
medico	Medico	private	Medico del paziente

Metodi				
Nome	**Parametri**	**Return**	**Accesso**	**Descrizione**
getCodiceFiscale	void	String	public	Restituisce il codice fiscale

setCodiceFiscale	String codFisc	void	public	Modifica il codice fiscale
addMenu	Menu m	void	public	Aggiunge un menu
removeMenu	Menu m	Menu	public	Rimuove e restituisce l'elemento
getMedico	void	Medico m	public	Restituisce il medico
setMedico	Medico m	value	public	Modifica il medico

3.1.5 RegistrazioneControl

Descrizione: Effettua i controlli sui parametri di registrazione, e nel caso in cui siano corretti completa la registrazione.

Dipendenze: utenteRegistrato, Account Manager

Attributi			
Nome	**Tipo**	**Accesso**	**Descrizione**
-	-	-	-

Metodi				
Nome	**Parametri**	**Return**	**Accesso**	**Descrizione**
controllaContenuto	String Nome, Cognome, Codice Fiscale, DataDiNascita, Città Indirizzo, Email, Password, RipetiPassword	boolean	private	Metodo che controlla se i parametri sono stati inseriti in maniera corretta e non siano lasciati vuori.
creaUtente	String Nome, Cognome, Codice Fiscale, DataDiNascita, Città Indirizzo, Email, Password, RipetiPassword	UtenteRegistrato	public	Metodo che crea un oggetto di tipo utenteRegistrato nel caso in cui la condizione precedente sia andata a buon fine.

3.1.6 LoginControl

Descrizione: Classe che si occupa della verifica delle credenziali per l'accesso

Dipendenze: AccountManager

Attributi			
Nome	**Tipo**	**Accesso**	**Descrizione**
-	-	-	-

Metodi				
Nome	**Parametri**	**Return**	**Accesso**	**Descrizione**
controllaContenuto	String Email, String Password	boolean	private	Controlla se i parametri inseriti siano tutti presenti in maniera corretta.
login	String Email, String Password	boolean	public	Richiede all'AccountManager di trovare l'utente nel db e controlla che i dati corrispondano

3.1.7 AccountManager

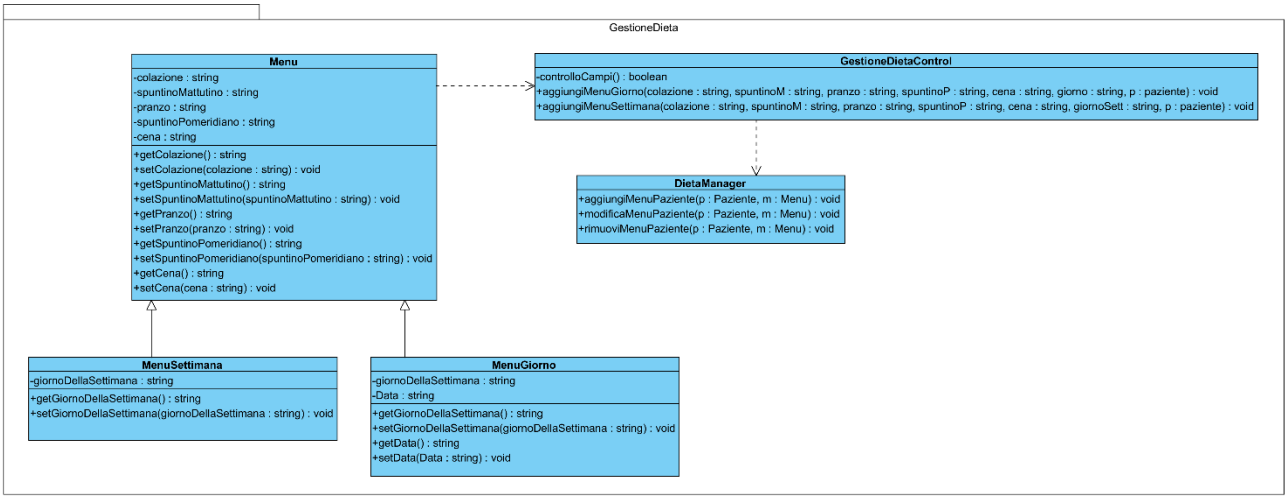
Descrizione: Classe che si occupa dell'interazione col database per la gestione degli utenti.

Dipendenze:

Attributi			
Nome	**Tipo**	**Accesso**	**Descrizione**
-	-	-	-

Metodi				
Nome	**Parametri**	**Return**	**Accesso**	**Descrizione**
getUtente	String email	UtenteRegistrato	public	Cerca un utente nel db tramite l'email e lo restituisce
aggiungiUtente	UtenteRegistrato utente	void	public	Aggiunge un utente registrato al db
modificaUtente	UtenteRegistrato utente	void	public	Modifica i dati di un utente nel db

3.2 GestioneDieta



3.2.1 Menu

Descrizione: Contiene informazioni sui cibi nel menù

Dipendenze:

Attributi			
Nome	**Tipo**	**Accesso**	**Descrizione**
colazione	String	private	Colazione del menù
spuntinoMattutino	String	private	Spuntino mattutino del menù
pranzo	String	private	Pranzo del menù
spuntinoPomeridiano	String	private	Spuntino pomeridiano del menù
cena	String	private	Cena del menù

Metodi				
Nome	**Parametri**	**Return**	**Accesso**	**Descrizione**
getColazione	void	String	public	Restituisce la colazione
SetColazione	String colazione	void	public	Setta la colazione
getSpuntinoMattutino	void	String	public	Restituisce lo spuntino mattutino
setSpuntinoMattutino	String spuntinoM	void	public	Setta lo spuntino mattutino
getPranzo	void	String	public	Restituisce il pranzo
setPranzo	String pranzo	void	public	Setta il pranzo
getSpuntinoPomeridiano	void	String	public	Restituisce lo spuntino pomeridiano

setSpuntinoPomeridiano	String spuntinoP	void	public	Setta lo spuntino pomeridiano
getCena	void	String	public	Restituisce la cena
setCena	String cena	void	public	Setta la cena

3.2.2 MenuSettimana

Descrizione: Il menù di una settimana

Dipendenze: Menu

Attributi			
Nome	**Tipo**	**Accesso**	**Descrizione**
giornoDellaSettimana	String	private	Giorno della settimana

Metodi				
Nome	**Parametri**	**Return**	**Accesso**	**Descrizione**
getGiornoDellaSettimana	void	String	public	Restituisce il giorno
setGiornoDellaSettimana	String giorno	void	public	Setta il giorno

3.2.3 MenuGiorno

Descrizione: Il menù di un giorno

Dipendenze: Menu

Attributi			
Nome	**Tipo**	**Accesso**	**Descrizione**
giornoDellaSettimana	String	private	Giorno della settimana
Data	String	private	Data

Metodi				
Nome	**Parametri**	**Return**	**Accesso**	**Descrizione**
getGiornoDellaSettimana	void	String	public	Restituisce il giorno
setGiornoDellaSettimana	String giorno	void	public	Setta il giorno
getData	void	String	public	Restituisce la data
setData	String data	void	public	Setta la data

3.2.4 GestioneDietaControl

Descrizione: Classe che si occupa di creare, modificare e visualizzare i menù dei pazienti

Dipendenze: Menu, Paziente, DietaManager

Attributi			
Nome	**Tipo**	**Accesso**	**Descrizione**
-	-	-	-

Metodi				
Nome	**Parametri**	**Return**	**Accesso**	**Descrizione**
controlloCampi	String colazione, spuntinoM, pranzo, spuntinoP, cena	boolean	private	Controlla che i campi siano stati riempiti correttamente dall'utente
aggiungiMenuGiorno	String colazione, spuntinoM, pranzo, spuntinoP, cena, giorno. Paziente p	void	public	Aggiunge a un paziente un nuovo menù
aggiungiMenuSettimana	String colazione, spuntinoM, pranzo, spuntinoP, cena, giornoSett. Paziente p	void	public	Aggiunge a un paziente un nuovo menù della settimana
visualizzaMenu	Paziente p	void	public	Visualizza i menu di un paziente

3.2.5 DietaManager

Descrizione: Classe che si occupa dell'interazione col database per la gestione della dieta

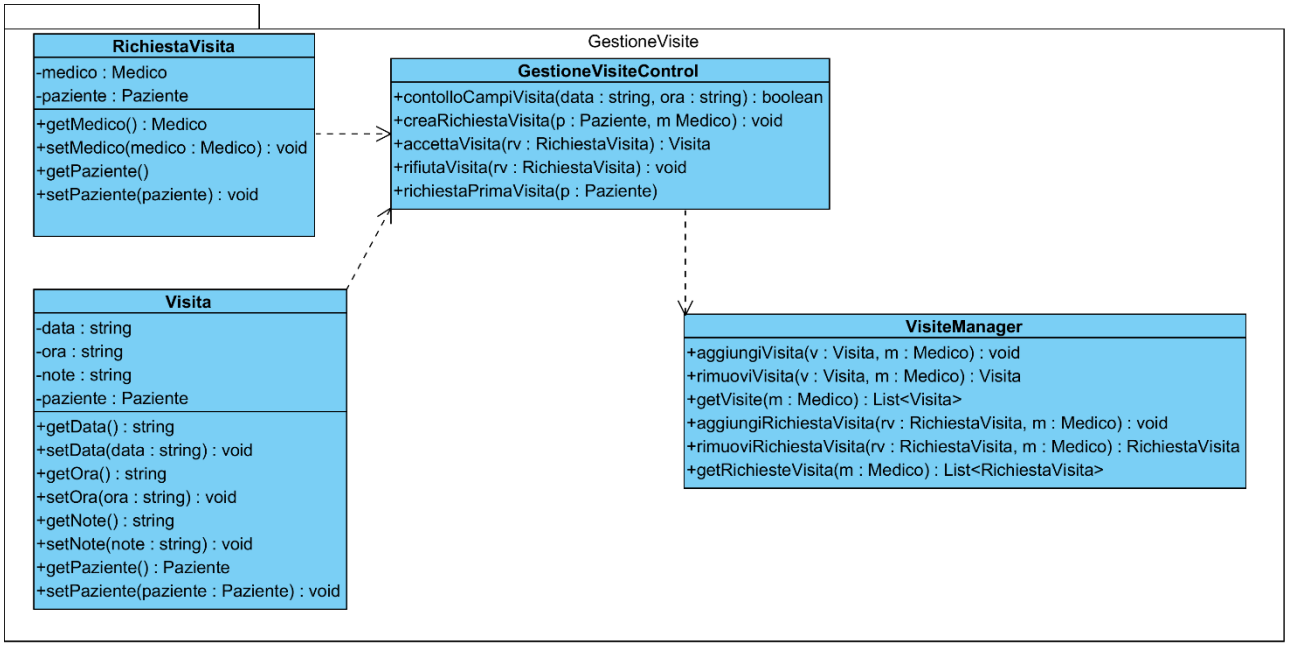
Dipendenze:

Attributi			
Nome	**Tipo**	**Accesso**	**Descrizione**
-	-	-	-

Metodi				
Nome	**Parametri**	**Return**	**Accesso**	**Descrizione**

getMenuPaziente	Paziente p	List<Menu>	public	Cerca nel db tutti i menu di un paziente e li restituisce in una Lista
aggiungiMenuPaziente	Paziente p, Menu m	void	public	Aggiunge al db un menù per il paziente specificato
modificaMenuPaziente	Paziente p, Menu m	void	public	Modifica un menu di un paziente
rimuoviMenuPaziente	Paziente p, Menu m	void	public	Rimuove un menu del paziente

3.3 GestioneVisite



3.3.1 RichiestaVisita

Descrizione: Classe che rappresenta la richiesta di visita fatta da un paziente al medico

Dipendenze: Medico, Paziente

Attributi			
Nome	**Tipo**	**Accesso**	**Descrizione**
medico	Medico	private	Medico a cui è riferita la richiesta
paziente	Paziente	private	Paziente che richiede la visita

Metodi

Nome	Parametri	Return	Accesso	Descrizione
getMedico	void	Medico	public	Restituisce il medico
setMedico	Medico m	void	public	Setta un medico
getPaziente	void	Paziente	public	Restituisce il paziente
setPaziente	Paziente p	Void	public	Setta il paziente

3.3.2 Visita

Descrizione: Classe che rappresenta una visita creata da un medico

Dipendenze: Paziente

Attributi			
Nome	**Tipo**	**Accesso**	**Descrizione**
data	String	private	Data della visita
ora	String	private	Ora della visita
note	String	private	Note della visita
paziente	Paziente	private	Paziente di questa visita

Metodi				
Nome	**Parametri**	**Return**	**Accesso**	**Descrizione**
getData	void	string	public	Restituisce la data
setData	string value	void	public	Modifica la data
getOra	void	string	public	Restituisce l'ora
setOra	String value	void	public	Modifica l'ora
getNote	void	string	public	Restituisce la nota
setNote	String value	void	public	Modifica la nota
getPaziente	void	Paziente	public	Restituisce il paziente
setPaziente	Paziente value	void	public	Modifica il paziente

3.3.3 GestioneVisiteControl

Descrizione: Classe che si occupa di gestire creazione e invio di richieste di visita, e creazione, modifica ed eliminazione di visite

Dipendenze: VisiteManager, Paziente, Medico, RichiestaVisita, Visita

Attributi			
Nome	**Tipo**	**Accesso**	**Descrizione**
-	-	-	-

Metodi

Nome	Parametri	Return	Accesso	Descrizione
controlloCampiVisita	String data, String ora	boolean	private	Controlla che i campi siano stati compilati correttamente
creaRichiestaVisita	Paziente p, Medico m	void	public	Crea una nuova visita
accettaVisita	RichiestaVisita rv	Visita	public	Accetta la RichiestaVisita, trasformandola in visita
rifiutaVisita	RichiestaVisita rv	void	public	Elimina la RichiestaVisita
richiestaPrimaVista	Paziente P	boolean	public	Permette al paziente di scegliere un medico e di inviargli una richiesta di visita

3.3.4 VisiteManager

Descrizione: Classe che si occupa dell'interazione col db per la gestione delle visite

Dipendenze:

Attributi			
Nome	**Tipo**	**Accesso**	**Descrizione**
-	-	-	-

Metodi				
Nome	**Parametri**	**Return**	**Accesso**	**Descrizione**
AggiungiVisita	Visita v, Medico m	void	public	Aggiungi una visita
RimuoviVisita	Visita v, Medico m	Visita	public	Rimuovi visita e restituisci la visita eliminata
getVisite	Medico m	List<Visita>	public	Restituisce la lista delle visite

AggiungiRichiestaVisita	RichiestaVisita rv, Medico m	void	public	Aggiunge una richiesta di visita
RimuoviRichiestaVisita	RichiestaVisita rv, Medico m	RichiestaVisita	public	Rimuove una richiesta di visita
getRichiesteVisite	Medico m	List<RichiestaVisita>	public	Restituisce la lista delle richieste di visita