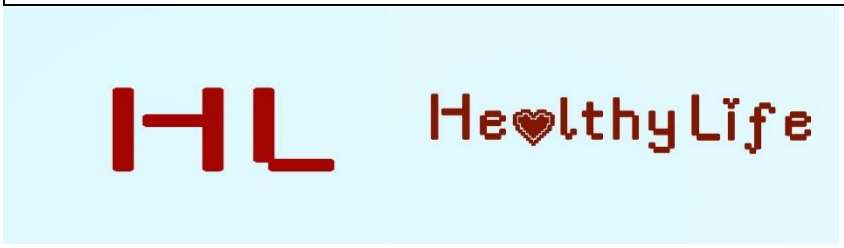
Università degli Studi di Salerno
Corso di ingegneria del Software



HealthyLife

Object Design Document

Versione 1.2

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 indentazione 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

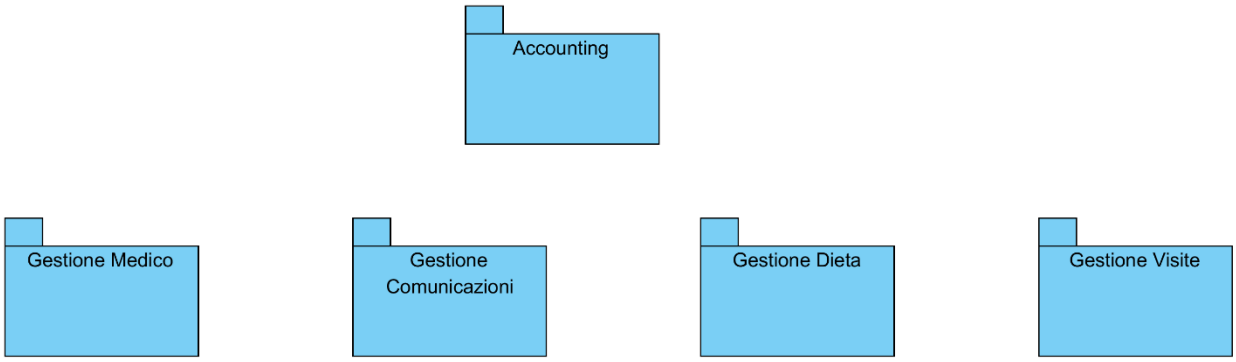
1.4 Riferimenti

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.

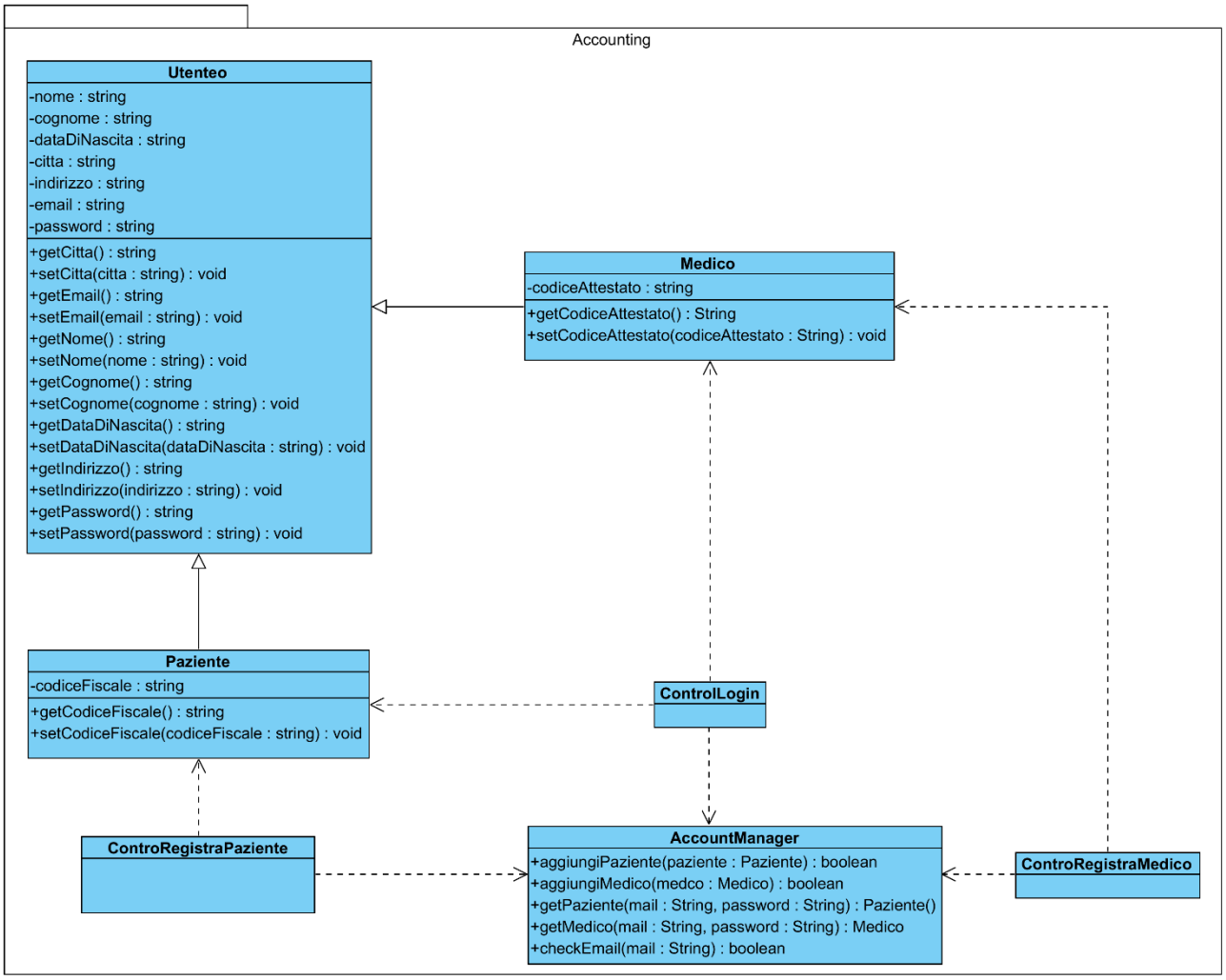


- Accounting: package che racchiude le classi che rappresentano gli utenti (Paziente e Medico) e le funzionalità di registrazione, login, modifica dei dati
- Gestione medico: package che racchiude le funzionalità offerte al paziente per la gestione del medico: aggiunta e cambio medico e le fuznioni di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

Nota bene: per semplificare I diagrammi sono stati omessi alcuni elementi, si veda la descrizione sotto ciascun diagramma.

3.1 Accounting



3.1.1 Utente

Descrizione: Un utente registrato alla piattaforma

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
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
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

Metodi				
Nome	Parametri	Return	Accesso	Descrizione
getCodiceAttestato	void	String	public	Restituisce attestato
setCodiceAttestato	String codAtt	void	public	Modifica il codice Attestato

3.1.4 Paziente

Descrizione: Un paziente registrato alla piattaforma.

Dipendenze:

Attributi			
Nome	Tipo	Accesso	Descrizione
codiceFiscale	String	private	Codice Fiscale di un paziente

Metodi				
Nome	Parametri	Return	Accesso	Descrizione
getCodiceFiscale	void	String	public	Restituisce il codice fiscale
setCodiceFiscale	String codFisc	void	public	Modifica il codice fiscale

3.1.5 ControlRegistraPaziente

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

Dipendenze: Paziente, AccountManager

Parametri		
Nome	Tipo	Descrizione
name	String	Nome del paziente
surname	String	Cognome del paziente
address	String	Indirizzo del paziente
girornoNascita	Int	Giorno della data di nascita
MeseNascita	Int	Mese della data di nascita
annoNascita	Int	Anno della data di nascita
Email	String	Email del paziente
Citta	String	Citta del paziene
codiceFiscale	String	Codice Fiscale del paziente
Password	String	Password
rPassword	String	Ripetizione della password

3.1.5 ControlRegistraMedico

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

Dipendenze: Medico, AccountManager

Parametri		
Nome	Tipo	Descrizione
name	String	Nome del medico
surname	String	Cognome del medico
address	String	Indirizzo del medico
girornoNascita	Int	Giorno della data di nascita
MeseNascita	Int	Mese della data di nascita
annoNascita	Int	Anno della data di nascita
Email	String	Email del medico
Citta	String	Citta del paziene
codiceAttestato	String	Codice Attestato del medico
Password	String	Password
rPassword	String	Ripetizione della password

3.1.6 ControlLogin

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

Dipendenze: AccountManager, Medico, Paziente

Parametri		
Nome	Tipo	Descrizione
mail	String	Indirizzo email
password	String	Password

3.1.7 Login

Descrizione: Pagina di login

Dipendenze: ControlLogin

Parametri		
Nome	Tipo	Descrizione
-	-	-

3.1.8 RegistraPaziente

Descrizione: Pagina di registrazione di un paziente

Dipendenze: ControlRegistraPaziente

Parametri		
Nome	Tipo	Descrizione
-	-	-

3.1.9 RegistraMedico

Descrizione: Pagina di registrazione di un medico

Dipendenze: ControlRegistraMedico

Parametri		
Nome	Tipo	Descrizione
-	-	-

3.1.10 AccountManager

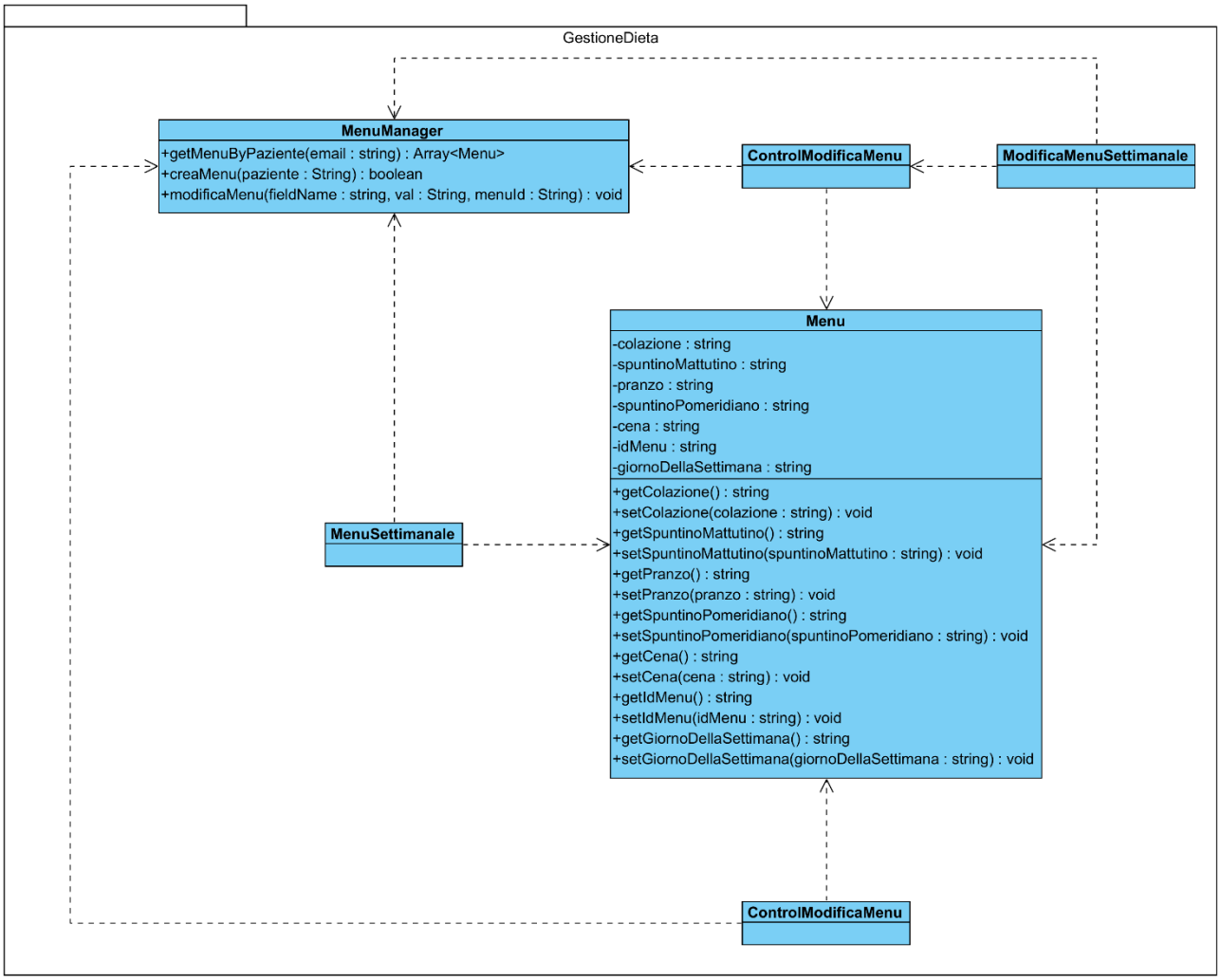
Descrizione: Classe che si occupa dell'interazione col database per la gestione degli utenti.

Dipendenze:

Attributi			
Nome	Tipo	Accesso	Descrizione
-	-	-	-

Metodi				
Nome	Parametri	Return	Accesso	Descrizione
aggiungiPaziente	Paziente paziente	Boolean	public	Aggiunge un paziente al db
aggiungiMedico	Medico medico	Boolean	public	Aggiunge un medico al db
getPaziente	String mail, String passwor	Paziente	public	Cerca nel database un paziente che rispetti i parametri
getMedico	String mail, String passwor	Medico	public	Cerca nel database un medico che rispetti i parametri
checkEmail	String mail	Boolean	Public	Controlla se un indirizzo email è già presente nel database

3.2 GestioneDieta



3.2.1 Menu

Descrizione: Contiene informazioni sui cibi nel menù

Dipendenze:

Attributi			
Nome	Tipo	Accesso	Descrizione
idMenu	String	Public	Id del menu nel database
giornoDellaSettimana	String	Public	Giorno della settimana a cui è riferito il menu
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
getId	void	String	public	Restituisce l'id del menu
SetId	String id	void	public	Setta l'id del menu
getGiornoDellaSettimana	void	String	public	Restituisce il giorno della settimana a cui è riferito il menu
setGiornoDellaSettimana	String getGiornoDellaSettimana	void	public	Setta il giorno della settimana a cui è riferito il menu
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 ControlModificaMenu

Descrizione: Control che si occupa di creare e modificare i menù dei pazienti

Dipendenze: Menu, MenuManager

Parametri		
Nome	Tipo	Descrizione
pazienteId	String	Id (email) del paziente a cui sno riferiti i menu
menuId	Strind	Id del menu da mudificare
Fieldname	String	Campo del menu da settare (pranzo, spuntino,ecc)
Val	String	Contenuro del campo da settare

3.2.3 MenuSettimanale

Descrizione: Pagina che visualizza i menu di un paziente

Dipendenze: Menu, MenuManager

Parametri		
Nome	Tipo	Descrizione
Paziente	String	Id (email) del paziente a cui sno riferiti i menu

3.2.4 ModificaMenuSettimanale

Descrizione: Pagina che visualizza e permette di modificare il menu di un paziente

Dipendenze: Menu, MenuManager, ControlModificaMenu

Parametri		
Nome	Tipo	Descrizione
Paziente	String	Id (email) del paziente a cui sno riferiti i menu

3.2.5 MenuManager

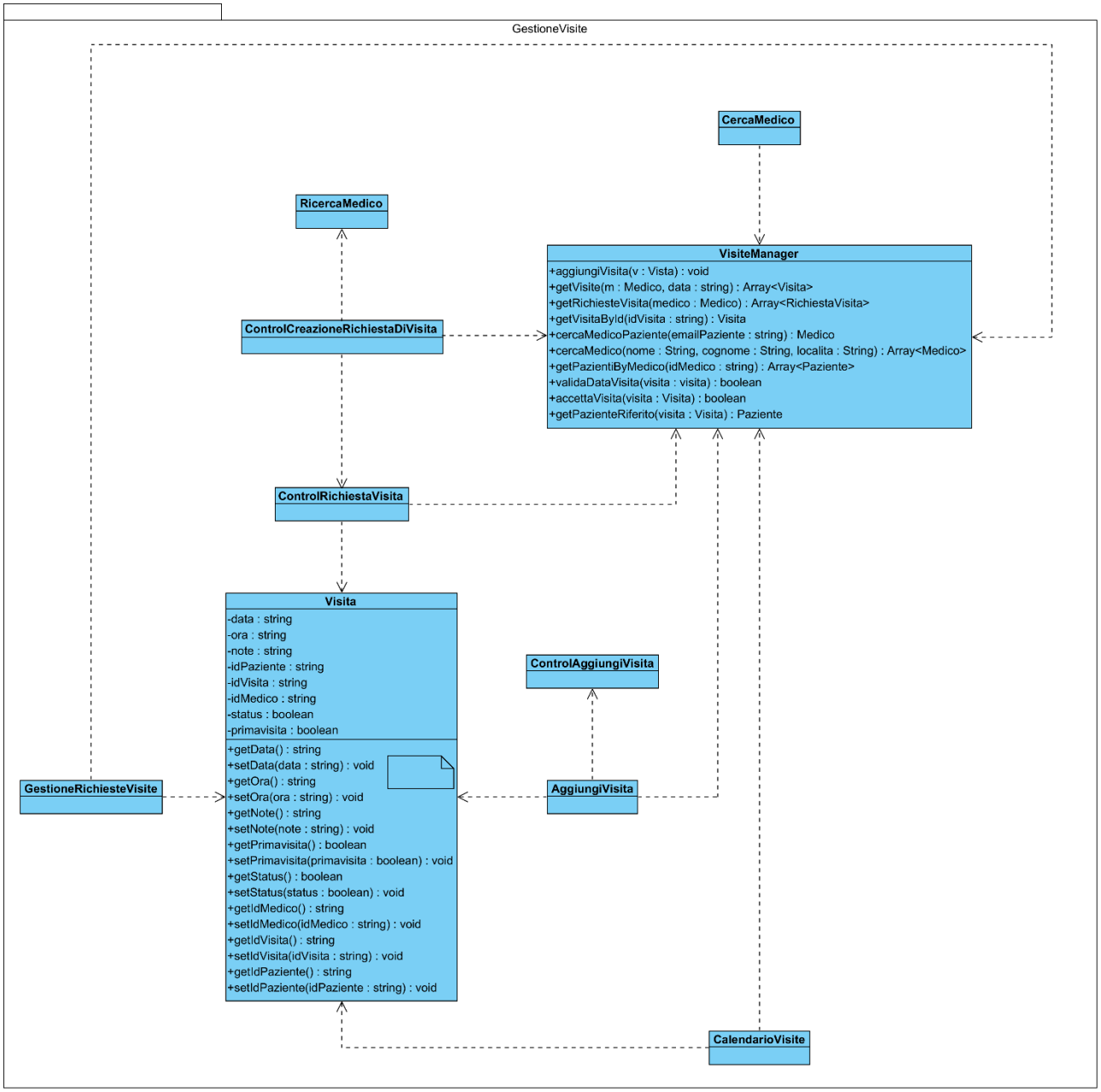
Descrizione: Classe che si occupa dell'interazione col database per la gestione della dieta

Dipendenze:

Attributi			
Nome	Tipo	Accesso	Descrizione
-	-	-	-

Metodi				
Nome	Parametri	Return	Accesso	Descrizione
getMenuByPaziente	String email	Array<Menu>	public	Cerca nel db tutti i menu di un paziente e li restituisce in un array
creaMenu	String paziente	Boolean	public	Crea sette menu per il paziente, uno per giorno della settimana
modificaMenu	String fieldName, String val, String menuId	Boolean	public	Modifica un campo (pranzo, cena, ecc) di uno specifico menu

3.3 Gestione Visite



3.3.1 Visita

Descrizione: Classe che rappresenta una visita o prima visita. Se non ha settattato il campo status allora è una richiesta di visita

Dipendenze:

Attributi			
Nome	Tipo	Accesso	Descrizione
idVisita	String	private	Id della visita
data	String	private	Data della visita
ora	String	private	Ora della visita
note	String	private	Note della visita
idPaziente	String	private	Paziente di questa visita
IdMedico	String	Public	Medico a cui è riferita la visita
Status	Boolean	Public	Status della visita: true accettata, false rifiutata, null per una richiesta di visita
Primavisita	Boolean	Public	Indica se si tratta di una richiesta di prima 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	String	public	Restituisce l'id paziente
setPaziente	String id	void	public	Modifica il paziente
getMedico	void	String	public	Restituisce l'id del medico
setMedico	String id	void	public	Modifica il medico
getStatus	void	Boolean	public	Restituisce lo status
setStatus	Boolean value	void	public	Modifica lo status
setPrimaVisita	Boolean value	void	public	Modifica il tipo di visita
getPrimaVisita	void	Boolean	public	Restituisce il tipo di visita

3.3.2 ControlCreazioneRichiestaDiVisita

Descrizione: Control che controlla se il paziente ha gia un medco. In caso affermativo invoca ControlRichiestaVisita che invia la visita, altrimenti carica la pagina di ricerca di un medico

Dipendenze: VisiteManager, ControlRichiestaVisita, RicercaMedico

Parametri		
Nome	Tipo	Descrizione
-	-	-

3.3.3 CercaMedico

Descrizione: pagina per la ricerca di un medico

Dipendenze: VisiteManager, Medico

Parametri		
Nome	Tipo	Descrizione
Nome	String	Nome del medico
Cognome	String	Cognome del medico
Localitò	String	Localita dove si trova il medico

3.3.4 ControlRichiestaVisita

Descrizione: Control che crea una visita o una prima visita e la invia.

Dipendenze: VisiteManager, Visita

Parametri		
Nome	Tipo	Descrizione
Medico	String	Medico della visita
Paziente	String	Paziente della visita
Primavisita	Boolean	Tipo di visita

3.3.5 GestioneRichiesteVisite

Descrizione: Pagina che mostra le richieste di visita inviate al medico: permette di accettarle

Dipendenze: VisiteManager, Visita, Paziente

Parametri		
Nome	Tipo	Descrizione
Medico	String	Medico della visita

3.3.6 AggiungiVisita

Descrizione: Pagina che permette di aggiungere una visita. Se ha settato nei parametri una richiesta di visita (idVisita e pazienteVisita) allora seleziona in automatico il paziente della visita.

Dipendenze: VisiteManager, Visita, Paziente, ControlAggiungiVisita

Parametri		
Nome	Tipo	Descrizione
Medico	String	Medico della visita
pazienteSelezionato	String	Paziente della visita
idVisita	String	Id della richiesta di visita

3.3.7 ControlAggiungiVisita

Descrizione: Control che aggiunge una visita al medico

Dipendenze: VisiteManager, Visita

Parametri		
Nome	Tipo	Descrizione
Medico	String	Medico della visita
Paziente	String	Paziente della visita
data	String	Data della visita
Orario	String	Orario della visita
Messaggio	Strind	Note della visita

3.3.8 CalendarioVisite

DescrizionePagina che mostra tutte le visite del medico per una certa data

Dipendenze: VisiteManager, Visita

Parametri		
Nome	Tipo	Descrizione
Medico	String	Medico della visita
data	String	Data della visita

3.3.9 VisiteManager

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

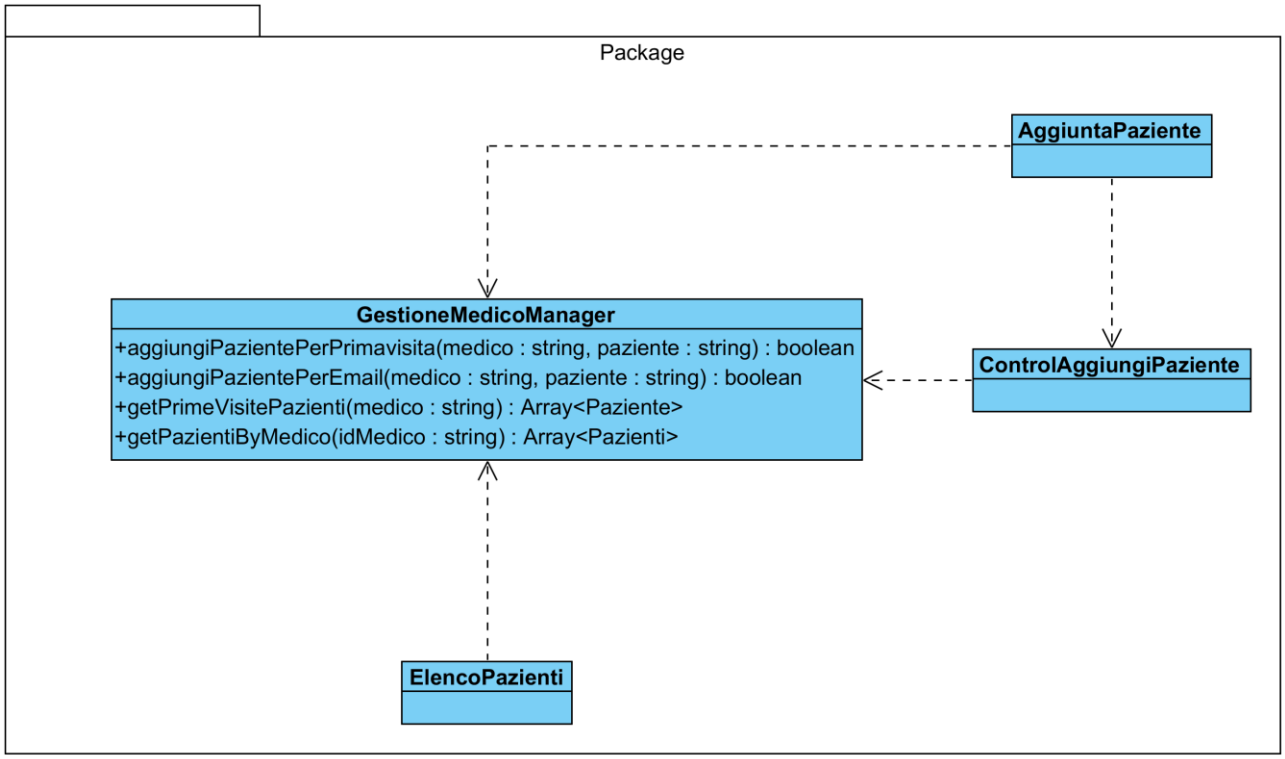
Dipendenze: Medico, Paziente

Attributi			
Nome	Tipo	Accesso	Descrizione
-	-	-	-

Metodi				
Nome	Parametri	Return	Accesso	Descrizione
AggiungiVisita	Visita v	Boolean	public	Aggiungi una visita
getVisite	Medico m, String data (opz.)	Array<Visita>	public	Restituisce la lista delle visite di un medico. Se settato, restituisce solo le visite di una certa data
getVisitaById	String visitaId	Visita	Public	Restituisce la visita che ha un id specifico

cercaMedicoPaziente	String emailPaziente	Medico	Public	Trova il medico di un certo paziente
cercaMedico	String nome, String cognome, String localita	Array<Medico>	Public	Resitituisce tutti i medici che rispettano i parametri passati
getPazientiByMedico	String idMedico	Array<Paziente>	Public	Trova tutti i pazienti di un medico
getPazienteRiferito	Visita visita	Paziente	Public	Restituisce il paziente di una visita
validaDataVisita	Visita visita	Boolean	Public	Controlla che la data di una visita non coincida con altre visite nel db
accettaVisita	Visita visita	Boolean	Public	Setta a true lo status di una visita
getRicheisteDiVisita	String medico	Array<Visita>	Public	Cerca tutte le richieste di visita di un medico

3.4 GestioneMedico



3.4.1 AggiuntaPaziente

Descrizione: Pagina che permette di aggiungere un paziente tra quelli che hanno richiesto una prima visita, oppure tramite email

Dipendenze: GestioneMedicoManager, ControlAggiungiPaziente

Parametri		
Nome	Tipo	Descrizione
--	-	-

3.4.2 ControlAggiungiPaziente

Descrizione: Control che aggiunge il paziente

Dipendenze: GestioneMedicoManager

Parametri		
Nome	Tipo	Descrizione
Tipo	String	Tipo di aggiunta: per prima visita o per email
IdPaziente	String	Id del paziente se è di tipo prima visita
emailPaziente	String	Email del paziente se è di tipo email

3.4.2 ElencoPazienti

Descrizione: Pagina che elenca tutti i pazienti di un medico

Dipendenze: GestioneMedicoManager, Paziente

Parametri		
Nome	Tipo	Descrizione
EmailMedico	String	Email del medico

3.4.3 GestioneMedicoManager

Descrizione: Classe che si occupa dell'interazione col database per il sottositema Gestione medico

Dipendenze:Paziente

Attributi			
Nome	Tipo	Accesso	Descrizione
-	-	-	-

Metodi				
Nome	Parametri	Return	Accesso	Descrizione
aggiungiPazientePerPrimavisita	String medico, String paziente	Boolean	public	Lega al medico un paziente che ha effettuato una prima visita
aggiungiPazientePerEmail	String medico, String paziente	Boolean	Public	Lega al medici un paziente cercato per email.
getPrimeVisitePazienti	String medico	Array<Paziente>	Public	Cerca tutti i pazienti di un medico che hanno una prima visita con esso
getPazientiByMedico	String idMedico	Array<Pazienti>	public	Cerca tutti i pazienti di un medico