

Documentazione progetto Ingegneria del Software

Battiato Matteo - VR500376
Beghini Lorenzo - VR501062

Settembre 2025

Contents

Requisiti ed interazioni utente-sistema

- Specifiche casi d'uso
 - Casi d'uso relativi ai Medici
 - Casi d'uso relativi ai Pazienti
 - Casi d'uso relativi Admin
- Diagrammi di attività

Sviluppo: progetto dell'architettura ed implementazione del sistema

- Note
- Organizzazione dei pacchetti
- Progettazione e pattern architetturali usati
- Implementazione e design pattern usati

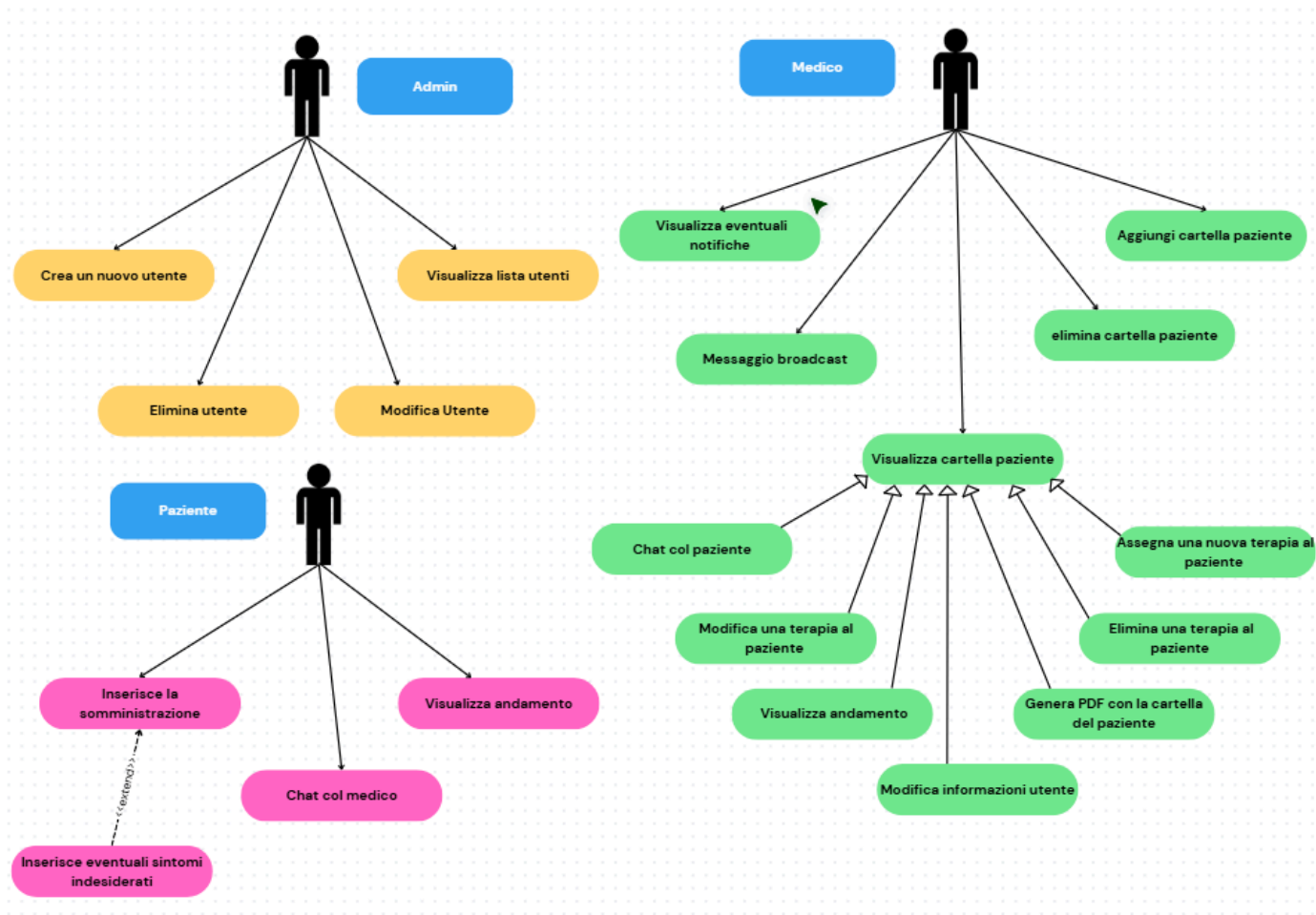
Attività di test e validazione

Requisiti ed interazioni utente-sistema

Specifiche casi d'uso

Note generali :

Il sistema creato può essere utilizzato sia dal medico che dai pazienti , inoltre da un addetto , amministratore , per la gestione utenti . Ogni tipo di utente ha le proprie credenziali , fornite dall'amministratore del sistema , con cui potranno effettuare l'autenticazione . Dopo aver effettuato l'autenticazione , nel caso quest'ultima vada a buon fine , l'utente verrà indirizzato alla rispettiva schermata iniziale , a seconda del tipo di utente che effettua il log in .



Casi d'uso relativi ai Medici

Dopo aver effettuato l'autenticazione, il medico viene indirizzato alla pagina dedicata alla gestione e supervisione dei suoi pazienti .

Visualizza eventuali notifiche

I medici potranno visionare , con l'utilizzo di una campanella con un eventuale numero di notifiche, se hanno ricevuto delle notifiche o devono ancora visualizzare dei messaggi .

Il medico premendo sulla campanella potrà visionare in un box dedicato quali e da chi ha ricevuto gli eventuali messaggi .

Attori : Medico

Precondizioni : il medico dev’essersi autenticato .

Passi :

1. Accede al sistema tramite credenziali .
2. Il medico preme sul bottone a forma di campanella .

Postcondizioni : I messaggi devono essere contrassegnati come letti.

In seguito ulteriori dettagli riguardano visualizza cartella paziente .

Aggiungi/elimina cartella paziente

In questa fase il medico può effettuare due azioni :

- Aggiungere una nuova cartella paziente
- Eliminare una cartella paziente

Con cartella paziente si intende una sezione che contiene i dati del paziente , come una tabella con le informazioni delle terapie che sono state assegnate a quel determinato paziente , un grafico col suo andamento ed una sezione con eventuali informazioni del paziente .

Attori : Medico

Precondizioni : il medico dev’essersi autenticato .

Passi :

1. Accede al sistema tramite credenziali .
2. Il medico preme sul bottone verde per creare una nuova cartella .
3. Il medico preme sul bottone rosso per eliminare una cartella .

Postcondizioni : nessuna .

In seguito ulteriori dettagli riguardano visualizza cartella paziente .

Visualizza cartella paziente

In questa fase il medico potrà visionare la cartella del paziente e svolgere diverse azioni :

- Assegna una nuova terapia al paziente
- Chat col paziente
- Elimina una terapia al paziente
- Modifica una terapia al paziente
- Visualizza andamento
- Genera PDF con la cartella del paziente
- Modifica informazioni utente

Il paziente che il medico desidera osservare verrà visualizzato dopo che il medico avrà inserito l'username del paziente in un apposito input , mentre la chat si potrà osservare una volta che la cartella del paziente verrà caricata correttamente con l'utilizzo di un'apposito bottone che aprirà la chat .

Attori : Medico

Precondizioni : il medico dev’essersi autenticato .

Passi :

1. Accede al sistema tramite credenziali .
2. Il medico inserisce username del paziente .
3. Avvia la ricerca ed il caricamento della cartella .
4. Il medico potrà svolgere diverse azioni all’interno della cartella come :
 - 4.1 Assegna una nuova terapia al paziente .
 - 4.2 Chat col paziente .
 - 4.3 Elimina una terapia al paziente .
 - 4.4 Modifica una terapia al paziente .
 - 4.5 Visualizza andamento .
 - 4.6 Genera PDF con la cartella del paziente .
 - 4.7 Modifica informazioni utente .

Postcondizioni : nessuna .

Messaggio broadcast

Il medico può inoltre inviare un messaggio nella chat dei suoi pazienti grazie ad un box dedicato , utile per messaggi informativi , come ferie , orari ecc...

Attori : Medico
Precondizioni : il medico dev’essersi autenticato .
Passi :
1. Accede al sistema tramite credenziali .
2. Il medico scrive il messaggio nell’aposto box .
3. Il medico preme sul bottone invio per inoltrare a tutti i suoi pazienti il messaggio .
Postcondizioni : inserimento messaggio database .

Il paziente potrà vedere il messaggio una volta aperta la chat col medico .

Casi d’uso relativi ai Pazienti

Dopo aver effettuato l'autenticazione, il paziente viene indirizzato alla pagina dedicata all'inserimento delle sue somministrazioni quotidiane, dove potrà visionare anche lui il suo andamento .

Inserisce la somministrazione

Il paziente potrà inserire la somministrazione che ha appena fatto , ha sei somministrazioni da fare durante il giorno , la prima al mattino prima e dopo la colazione , la seconda prima e dopo pranzo e la terza prima e dopo cena . Se durante il giorno dovesse dimenticarsi di inserirle il sistema invierà un messaggio , sotto forma di alert , dove gli ricorda le somministrazioni dei pasti che ha saltato , il sistema invierà al medico un messaggio qualora il paziente dovesse inserire dei valori al di fuori del range stabilito o se quest'ultimo non inserisce somministrazioni da almeno 3 giorni .

Attori : Paziente
Precondizioni : il paziente dev’essersi autenticato .
Passi :
1. Accede al sistema tramite credenziali .
2. Il paziente inserisce le somministrazioni .
3. Il paziente salva le somministrazioni .
Postcondizioni : salvataggio nel database .

In seguito ulteriori dettagli riguardano inserisce eventuali sintomi indesiderati o in chat col medico.

Visualizza andamento

Il paziente può osservare l'andamento delle sue somministrazioni con l'utilizzo di un grafico che può essere modellato nel tempo (a seconda del periodo che vuole visionare il paziente) .

Attori : Paziente
Precondizioni : il paziente dev’essersi autenticato .
Passi :
1. Accede al sistema tramite credenziali .
2. Il paziente osserva il grafico andamenti .
3. Il paziente preme i pulsanti per spostarsi nel tempo .
Postcondizioni : nessuna .

Inserisce eventuali sintomi indesiderati

Il paziente può inserire eventuali sintomi causati dalle somministrazioni .

Attori : Paziente
Precondizioni : il paziente dev’essersi autenticato .
Passi :
1. Accede al sistema tramite credenziali .
2. Il paziente inserisce i sintomi .
3. Il paziente salva i sintomi inseriti .
Postcondizioni : salvataggio nel database .

Chat col medico

Il paziente , con l'utilizzo di una chat , può inviare messaggi col medico , il sistema può inoltrare messaggi al medico in questa chat come spiegato in inserisce la somministrazione . Il paziente saprà di avere dei nuovi messaggi da leggere grazie ad una notifica sul bottone per l'apertura della chat .

Attori : Paziente
Precondizioni : il paziente dev’essersi autenticato .
Passi :
1. Accede al sistema tramite credenziali .
2. Il paziente preme il pulsante per aprire la chat .
3. Il paziente legge o scrive eventuali messaggi e li invia .
Postcondizioni : recapito/salvataggio messaggi .

Casi d’uso relativi Admin

Dopo aver effettuato l'autenticazione, l'amministratore viene indirizzato alla pagina dedicata alla gestione degli utenti .

Crea un nuovo utente

L'amministratore può creare un nuovo utente , dandogli username , tipo utente , password e nel caso fosse un paziente gli viene assegnato il suo medico .

Attori : amministratore
Precondizioni : l’amministratore dev’essersi autenticato .
Passi :
1. Accede al sistema tramite credenziali .
2. L’amministratore inserisce i dati del nuovo utente .
3. L’amministratore salva ill nuovo utente se non ci sono evenntuali problemi .
Postcondizioni : salvataggio nel database .

Visualizza lista utenti

L'amministratore può visionare tutta la lista degli utenti , coi loro dati .

Attori : amministratore
Precondizioni : l’amministratore dev’essersi autenticato .
Passi :
1. Accede al sistema tramite credenziali .
2. L’amministratore visualizza la lista utenti .
Postcondizioni : nessuna .

Modifica utente

L'amministratore può modificare un utente già esistente , inserisce il dato da modificare , quello nuovo , seleziona l'utente dalla tabella e salva .

Attori : amministratore
Precondizioni : l’amministratore dev’essersi autenticato .
Passi :
1. Accede al sistema tramite credenziali .
2. L’amministratore inserisce il nuovo dato .
3. L’amministratore seleziona l’utente dalla tabella .
4. L’amministratore salva le modifiche .
Postcondizioni : salvataggio nel database .

Elimina utente

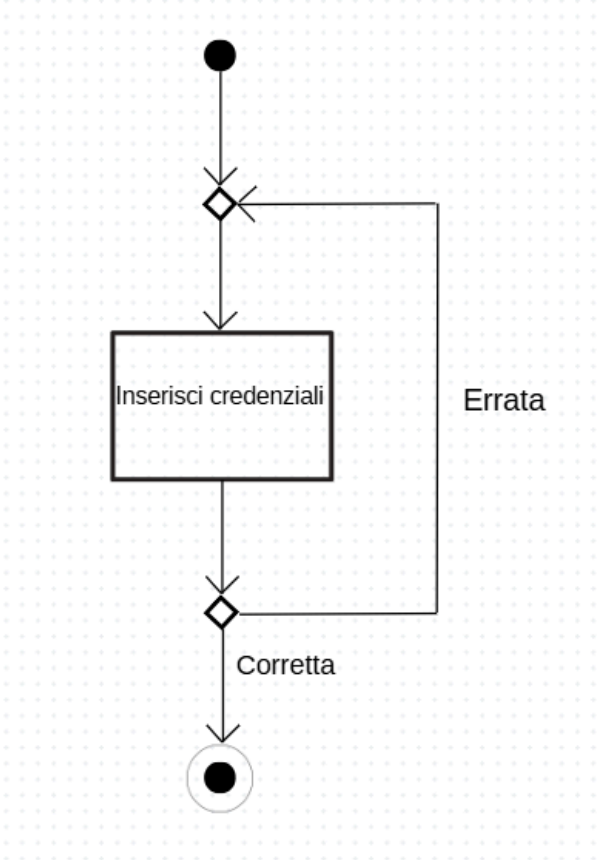
L'amministratore può eliminare utente presente nella tabella .

Attori : amministratore
Precondizioni : l’amministratore dev’essersi autenticato .
Passi :
1. Accede al sistema tramite credenziali .
2. L’amministratore seleziona l’utente dalla tabella da eliminare .
3. L’amministratore salva le modifiche .
Postcondizioni : salvataggio nel database .

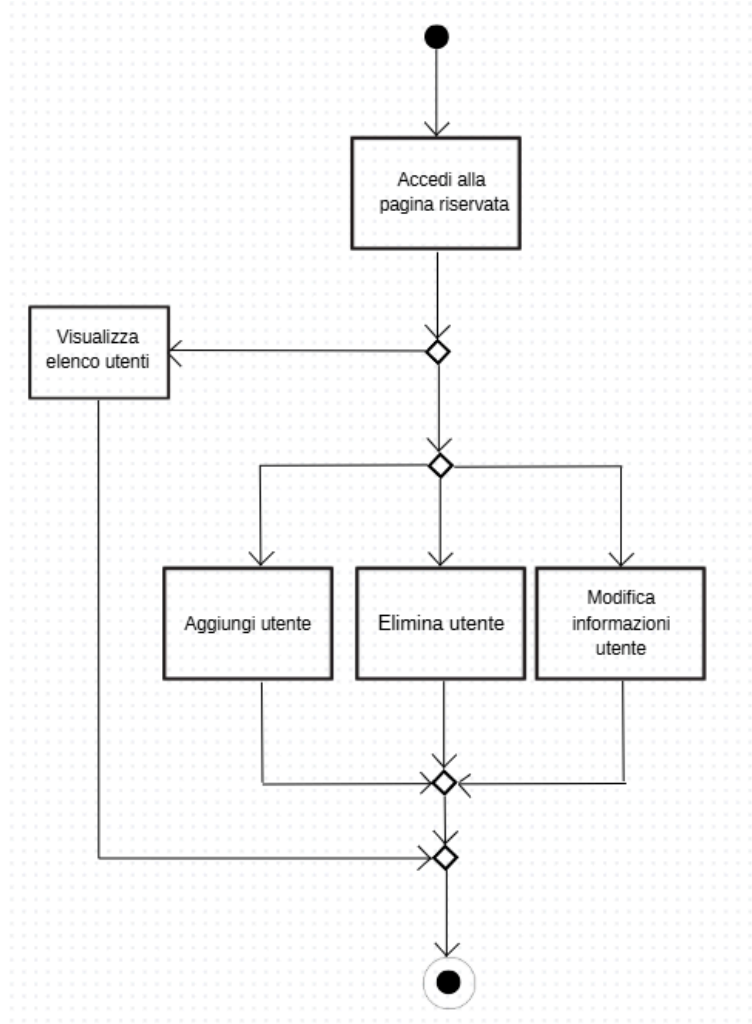
Diagrammi di attività

I seguenti diagrammi rappresentano il flusso di una singola serie di operazioni .

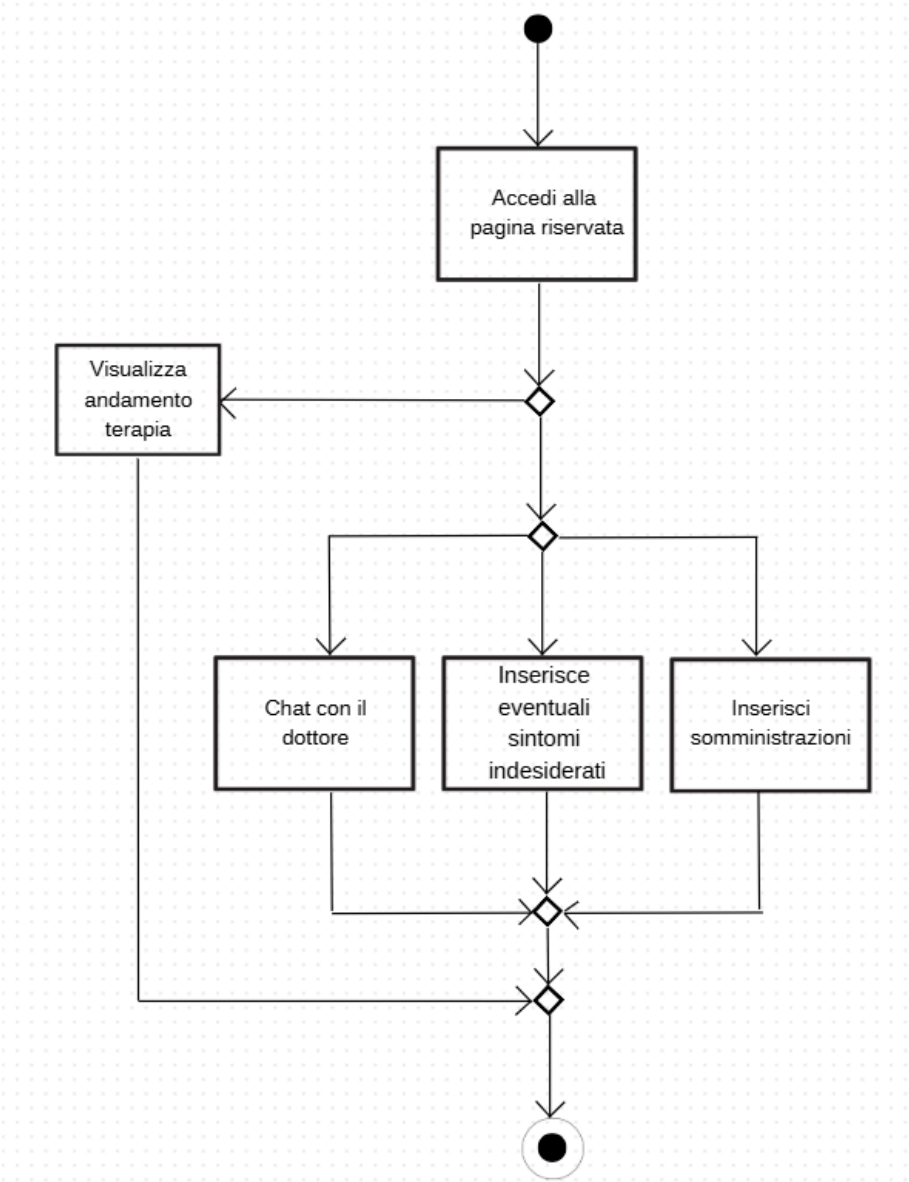
Autenticazione



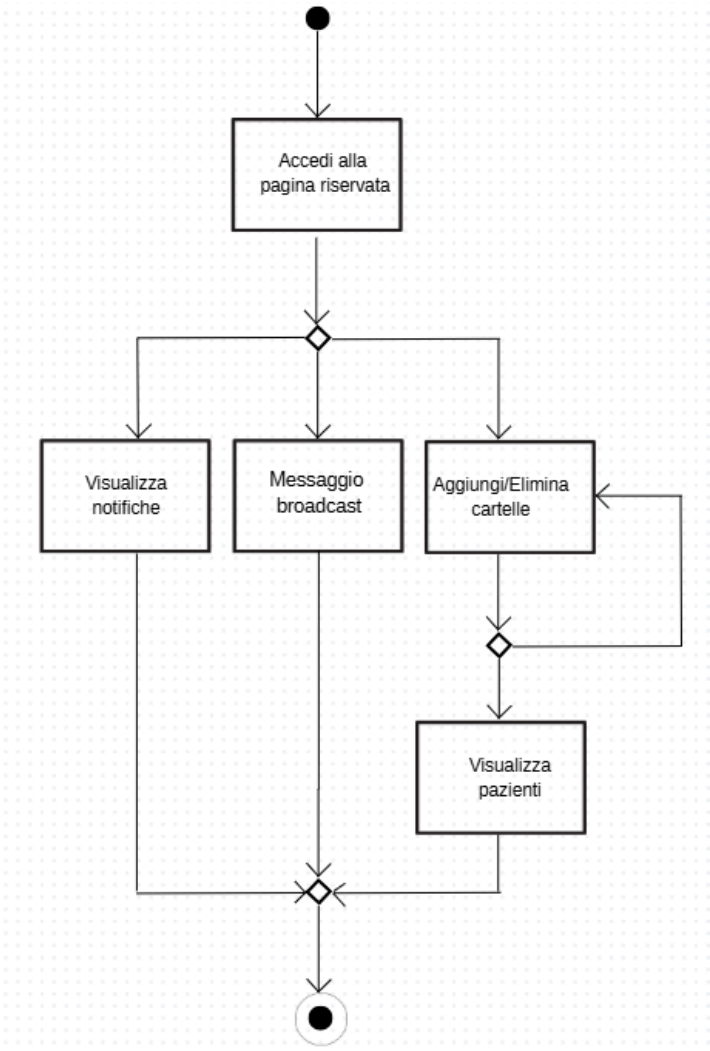
Admin



Paziente



Medico



Sviluppo: progetto dell’architettura ed implementazione del sistema

Note sul processo di sviluppo

Nel mio progetto è stata realizzata un’applicazione per la gestione di pazienti e medici per il diabete.

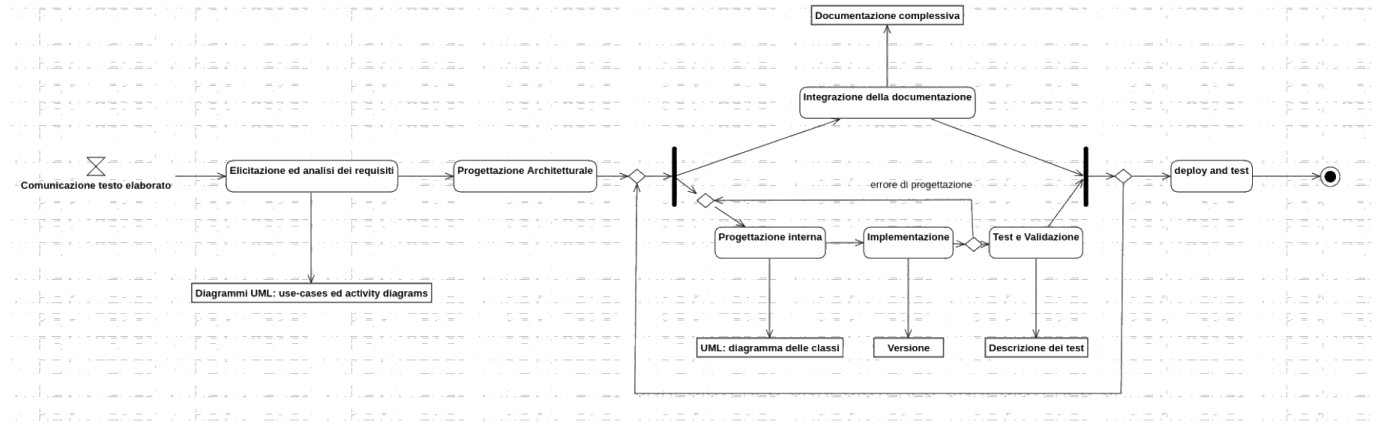
Nel processo di sviluppo si è cercato di mantenere sequenziali le fasi di progettazione , implementazione e validazione . Il motivo di questa scelta è stato quello di cercare di progettare e poi implementare .

Dopo ogni modifica sono stati svolti dei test .

Lo sviluppo del progetto non è sempre stato diretto e progressivo, ma ha avuto momenti di revisione e miglioramento del codice già scritto .

Durante lo sviluppo dell’architettura e dell’implementazione del sistema, è stata redatta una documentazione del codice volta a facilitare la comprensione, l’uso e la futura manutenzione del software .

La persistenza dei dati è gestita tramite il pattern DAO, che consente di astrarre l’accesso al database SQLite e mantenere separata la logica applicativa dalla logica di memorizzazione.



Organizzazione dei pacchetti

La suddivisione dei pacchetti nel progetto è la seguente :

- `controllers` : gestisce la logica di interazione con l'utente.
- `models` : definisce le entità e strutture dati.
- `DAO` : incapsula la logica di accesso al database.
- `utility` : contiene classi di supporto e metodi riutilizzabili.
- `view` : potrebbe includere logiche di visualizzazione o wrapper per le scene.
- `resources` : suddiviso in `FXML` , `CSS` , `img` per una gestione ordinata degli asset.

Progettazione e pattern architetturali usati

Il progetto è stato sviluppato seguendo un’architettura modulare e scalabile, con una chiara separazione tra i diversi livelli dell’applicazione. La struttura riflette l’adozione del pattern **MVC (Model-View-Controller)**, integrato con altri pattern di progettazione per migliorare la manutenibilità e la leggibilità del codice .

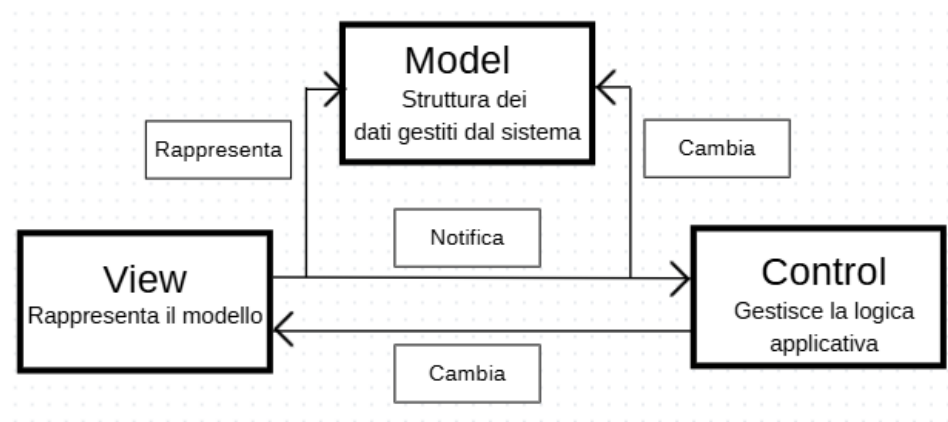
Ogni membro è stato inserito in un package separato .

- **Model** (`models`) : contiene le classi che rappresentano i dati e la logica di business .
- **View** (`resources/FXML`) : definisce l’interfaccia utente tramite file FXML .
- **Controller** (`controllers`) : gestisce gli eventi dell’interfaccia e collega la view al model .

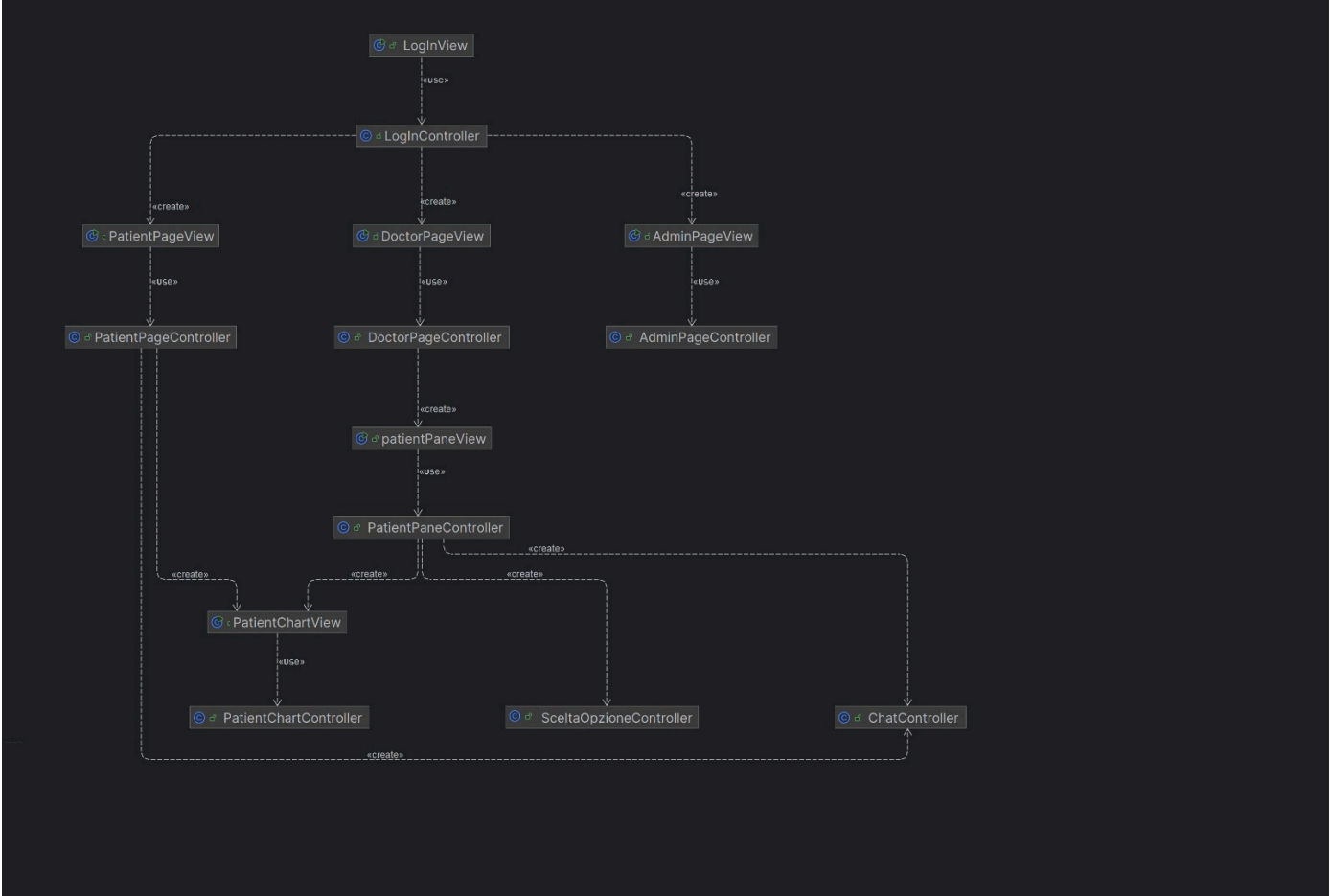
Questa suddivisione consente una gestione chiara delle responsabilità e facilita eventuali modifiche o estensioni future.

Il progetto utilizza un database SQLite (`identifier.sqlite` , `miodatabase.db`) per la memorizzazione dei dati. L’accesso è gestito tramite il pattern DAO, che garantisce l’astrazione e la separazione tra logica applicativa e persistenza.

Architettura MVC



Di seguito viene mostrato il Diagramma delle classi :



Se si desidera avere un quadro completo di tutto il progetto, consigliamo la visione della seguente immagine :

<https://github.com/battiatomatteo/Progetto-Java-Fx/blob/main/relazione/ProgettoJavaFXImg.png>

Design pattern usati

Pattern	Problema risolto	Dove / come lo ho applicato	Vantaggi
MVC (Model-View-Controller)	Separare la logica dell'interfaccia (View), la gestione degli eventi e del flusso (Controller), e i dati / logica di business (Model)	In JavaFX, normalmente ogni finestra / scena ha un controller; il Model sono le classi che rappresentano pazienti, medici, etc., e il codice che interagisce col DB è separato. Potresti descrivere quali classi sono controllers, quali sono model.	Mantiene il codice più organizzato, facile da modificare la GUI senza toccare la logica dei dati, facilita il testing.
DAO (Data Access Object)	Incapsulare l'accesso al database, separando la logica SQL/DB dal resto dell'applicazione	Se hai una classe o un pacchetto che si occupa solo di interazioni col database (operazioni CRUD: create, read, update, delete) per entità Paziente , Medico .	Isola i dettagli del DB, cambiando DB (o query) non serve cambiare il resto del codice, migliora la manutenzione.
Observer / Listener	Notificare componenti interessate quando uno stato cambia (utile nelle GUI per aggiornare viste quando cambia il modello)	JavaFX già usa listener / binding per proprietà osservabili; se usi proprietà osservabili in model o controller per aggiornare automaticamente la GUI, è un esempio.	GUI più reattiva, meno codice boilerplate per l'aggiornamento manuale, migliore separazione.

Note :

Nel progetto, la logica di accesso al database è incapsulata in classi DAO dedicate. Ad esempio, ho una classe `PazienteDAO` che implementa i metodi `insert` , `update` , `delete` , `findById` , `findAll` . Questo permette al resto dell'applicazione (GUI) di non preoccuparsi delle query SQL .

Per l'interfaccia grafica è stata adottata una struttura che rispecchia il pattern MVC: ogni schermata JavaFX ha un controller che risponde agli eventi (clic, input utente), un modello (classi dati e logica) separato, e le viste definite nei file FXML o nel codice JavaFX .

Test

Gestione errori / validazioni: verifiche di input da parte dell'utente, controllo che il DB risponda, gestione di eccezioni SQL.