Documentazione: Pazienti Ipertesi
Ingegneria del Software

Aldegheri Alessandro VR471346
Venturi Davide VR471414
Zerman Nicolò VR471404

Luglio 2023

Indice

Interazioni utenti-sistema

Specifiche casi d'uso

Note generali

Il sistema sviluppato nasce per soddisfare le necessità di due principali attori, **Medici** e **Pazienti**. Entrambe le categorie sono dotate di account, pre-inseriti dagli amministratori del sistema, le cui informazioni saranno fornite direttamente.
Potranno quindi poi svolgere l'azione di autenticazione e, in caso questa vada a buon fine, verranno reinderizzati alla rispettiva schermata iniziale, all'interno della quale potranno svolgere il loro compito.

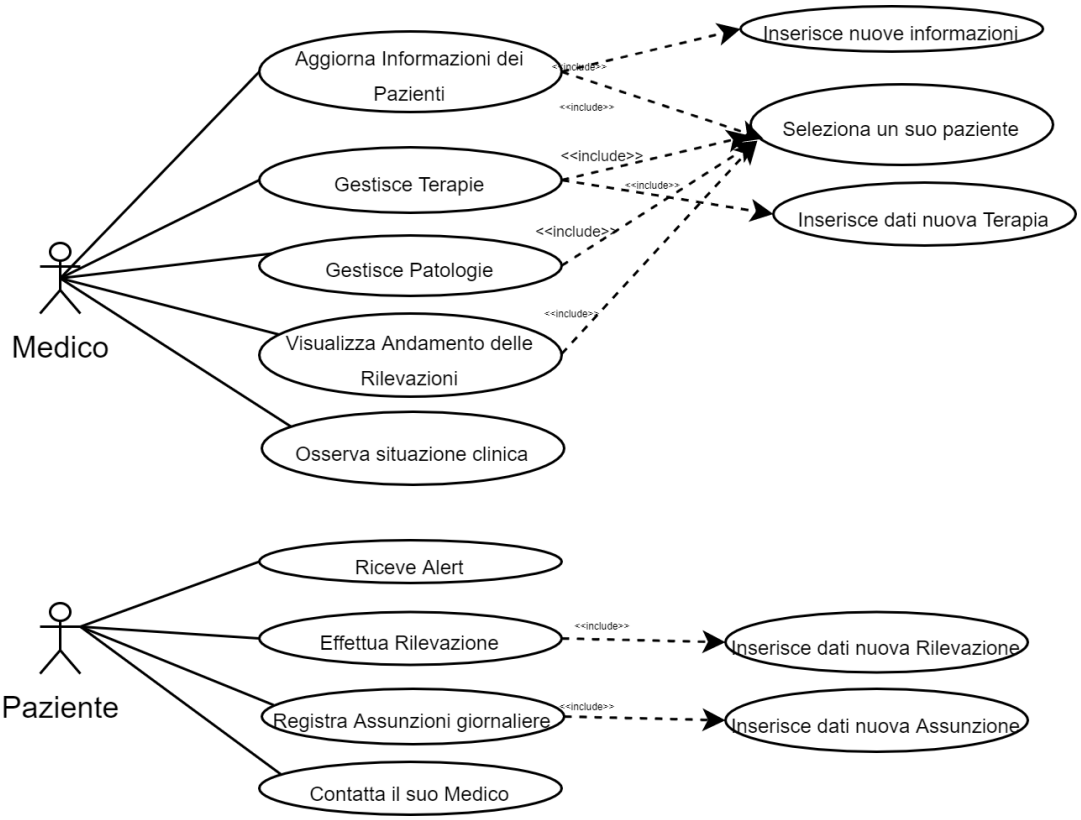


Immagine 1: Casi d'uso

Casi d'uso relativi ai Medici

Dopo aver completato il processo di autenticazione, il medico ha accesso alla sua interfaccia. Mediante quest'ultima potrà svolgere tutti i suoi doveri nel confronto dei propri pazienti.

Aggiornare le informazioni dei pazienti

Il medico può aggiornare una breve sezione di informazioni sul paziente, contenente fattori di rischio (fumatore, ex-fumatore, problemi di dipendenza, rischio di obesità, comorbidità presenti e così via)

Attori: *Medico*
Precondizioni: *Il medico deve essersi autenticato*
Passi:

1. *Il medico accede al sistema*
2. *Il medico è introdotto all'interfaccia di base*
3. *Il medico accede all'interfaccia di selezione di un suo paziente*
4. *Il medico seleziona il paziente interessato*
5. *Il medico viene rediretto ad una nuova schermata*
6. *Il medico inserisce le nuove informazioni nello spazio apposito*
7. *Il medico pigia il pulsante "UPDATE"*

Postcondizioni: *Le informazioni sono aggiornate*

Gestire le Terapie

In questa fase sono presenti tre alternative:

- **inserimento** nel sistema di una nuova terapia antipertensiva associata ad un proprio paziente, specificando: farmaco, numero di assunzioni giornaliere, la quantità di farmaco per ogni assunzione ed eventuali indicazioni
- **modifica** di una terapia già assegnata al paziente, in base all'evoluzione dello stato di quest'ultimo
- **terminazione** di una terapia già associata al paziente

Come data di inizio della terapia verrà assegnata la data del giorno in cui il medico sta facendo l'inserimento. Di conseguenza, anche la data di fine terapia sarà la data del giorno in cui medico sta effettuando la terminazione.

Attori: *Medico*
Precondizioni: *Il medico deve essersi autenticato*
Passi:

1. *Il medico accede al sistema*
2. *Il medico è introdotto all'interfaccia di base*
3. *Il medico accede all'interfaccia di selezione di un suo paziente*
4. *Il medico seleziona il paziente interessato*
5. *Il medico viene rediretto ad una nuova schermata*
6. *Il medico accede all'interfaccia di gestione delle terapie*
7. *Il medico inserisce i dati della nuova terapia*
8. *Il medico pigia il pulsante "INSERT"*

Postcondizioni: *La terapia è inserita*

Attori: *Medico*
Precondizioni: *Il medico deve essersi autenticato*
Passi:

1. *Il medico accede al sistema*
2. *Il medico è introdotto all'interfaccia di base*
3. *Il medico accede all'interfaccia di selezione di un suo paziente*
4. *Il medico seleziona il paziente interessato*
5. *Il medico viene rediretto ad una nuova schermata*
6. *Il medico accede all'interfaccia di gestione delle terapie*
7. *Il medico modifica le informazioni che vuole aggiornare*
8. *Il medico pigia il pulsante "UPDATE"*

Postcondizioni: *La terapia è aggiornata*

Attori: *Medico*
Precondizioni: *Il medico deve essersi autenticato*
Passi:

1. *Il medico accede al sistema*
2. *Il medico è introdotto all'interfaccia di base*
3. *Il medico accede all'interfaccia di selezione di un suo paziente*
4. *Il medico seleziona il paziente interessato*
5. *Il medico viene rediretto ad una nuova schermata*
6. *Il medico accede all'interfaccia di gestione delle terapie*
7. *Il medico seleziona la terapia da terminare*
8. *Il medico pigia il pulsante "END"*

Postcondizioni: *La terapia è conclusa*

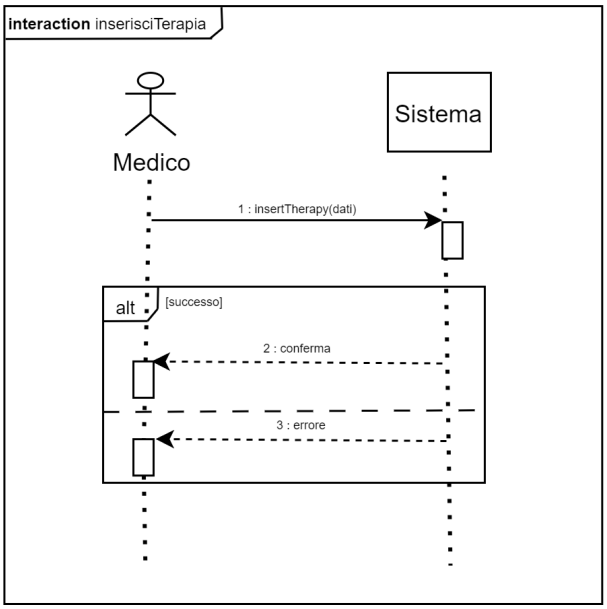


Immagine 2: Inserimento della terapia

Gestire le patologie

In questa fase sono presenti due alternative:

- **inserimento/aggiunta** di una nuova patologia, con conseguente data di inizio
- **terminazione** di una patologia che il paziente ha superato, con conseguente data di fine

La gestione delle date di inizio e fine è identica a quella spiegata nel paragrafo precedente.

Attori: *Medico*
Precondizioni: *Il medico deve essersi autenticato*
Passi:

1. *Il medico accede al sistema*
2. *Il medico è introdotto all'interfaccia di base*
3. *Il medico accede all'interfaccia di selezione di un suo paziente*
4. *Il medico seleziona il paziente interessato*
5. *Il medico viene rediretto ad una nuova schermata*
6. *Il medico accede all'interfaccia di gestione delle patologie*
7. *Il medico seleziona la patologia, dalla memoria del sistema, da aggiungere*
8. *Il medico pigia il pulsante "ADD"*

Postcondizioni: *La patologia è inserita*

Attori: *Medico*
Precondizioni: *Il medico deve essersi autenticato*
Passi:

1. *Il medico accede al sistema*
2. *Il medico è introdotto all'interfaccia di base*
3. *Il medico accede all'interfaccia di selezione di un suo paziente*
4. *Il medico seleziona il paziente interessato*
5. *Il medico viene rediretto ad una nuova schermata*
6. *Il medico accede all'interfaccia di gestione delle patologie*
7. *Il medico seleziona la patologia da terminare*
8. *Il medico pigia il pulsante "END"*

Postcondizioni: *La patologia è terminata*

Visualizzare l'andamento delle rilevazioni

Il medico potrà visualizzare un grafico che mostra l'andamento delle pressioni dei vari pazienti in un arco di tempo a scelta.
Per permettere al medico di fare una valutazione corretta, insieme ai valori di SBP e DBP, sono presenti anche terapie, sintomi e patologie presenti durante le misurazioni effettuate.

Attori: *Medico*
Precondizioni: *Il medico deve essersi autenticato*
Passi:

1. *Il medico accede al sistema*
2. *Il medico è introdotto all'interfaccia di base*
3. *Il medico accede all'interfaccia di selezione di un suo paziente*
4. *Il medico seleziona il paziente interessato*
5. *Il medico viene rediretto ad una nuova schermata*
6. *Il medico accede all'interfaccia di visualizzazione delle rilevazioni*
   (a) *Il medico seleziona un intervallo temporale*
   (b) *Il medico lascia l'intervallo temporale di default*
7. *Il medico visualizza l'andamento delle rilevazioni*

Postcondizioni: *Nessuna*

Osservare la situazione clinica dei pazienti

Il medico deve avere la possibilità di vedere chi non rispetta le terapie assegnate e chi registra pressioni oltre le soglie definite.
Difatti, dopo il login, il medico verrà subito introdotto all'interfaccia che avrà come primissime informazioni quelle appena descritte, dato che sono di fondamentale importanza.

Attori: *Medico*
Precondizioni: *Il medico deve essersi autenticato*
Passi:

1. *Il medico accede al sistema*
2. *Il medico è introdotto all'interfaccia di base*
3. *Il medico osserva i livelli di ipertensione e le terapie non rispettate dei suoi pazienti*

Postcondizioni: *Nessuna*

Casi d'uso relativi ai Pazienti

Ricevere Alert

Il sistema, all'accesso del paziente, deve inviare un alert se non sono ancora state presi tutti i farmaci giornalieri. Quindi, mostrerà quali sono i farmaci da prendere e anche la quantità prescritta dal medico.
Questo sarà mostrato ad ogni accesso fino a che il paziente non ha finito di prendere i farmaci per quel giorno.

Attori: *Paziente*
Precondizioni: *Il paziente deve essersi autenticato, devono esserci degli alert*
Passi:

1. *Il paziente accede al sistema*
2. *Il paziente visualizza gli alert*
3. *Il paziente chiude gli alert*

Postcondizioni: *Gli alert sono stati segnalati*

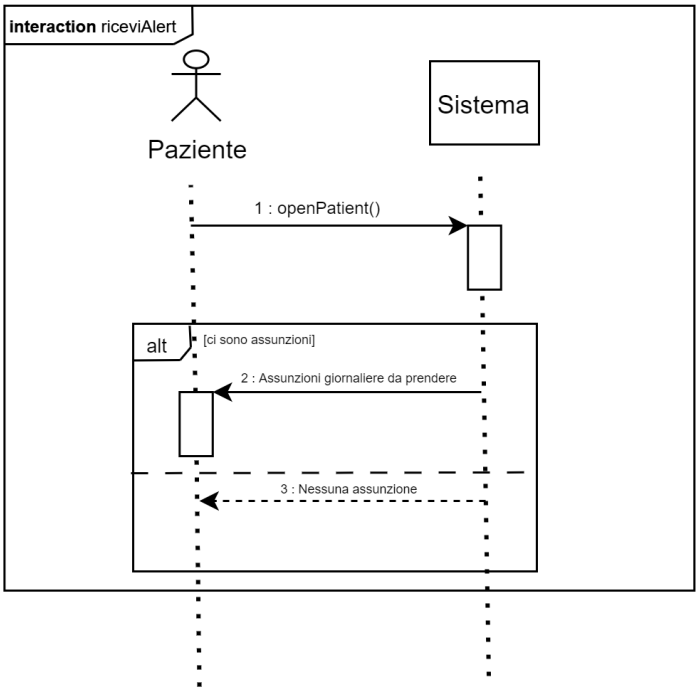


Immagine 3: Ricezione Alert

Effettuare Rilevazioni

Il paziente deve poter effettuare l'inserimento delle rilevazioni giornaliere di pressione arteriosa (la pressione sistolica (SBP) e la pressione diastolica (DBP)) ed eventuali sintomi.
Le relazioni con le eventuali terapie e patologie temporalmente concomitanti alle misurazioni, verranno segnate all'interno del database.
Il paziente non ha un limite di inserimento di misurazioni giornaliere, difatti potrà effettuarne all'infinito.

Attori: *Paziente*
Precondizioni: *Il paziente deve essersi autenticato*
Passi:

1. *Il paziente accede al sistema*
2. *Il paziente è introdotto all'interfaccia di base*
3. *Il paziente accede all'interfaccia di inserimento delle rilevazioni*
4. *Il paziente inserisce i dati della rilevazione ed eventuali sintomi*
5. *Il paziente pigia sul pulsante "INSERT MEASUREMENT"*

Postcondizioni: *La misurazione è inserita*

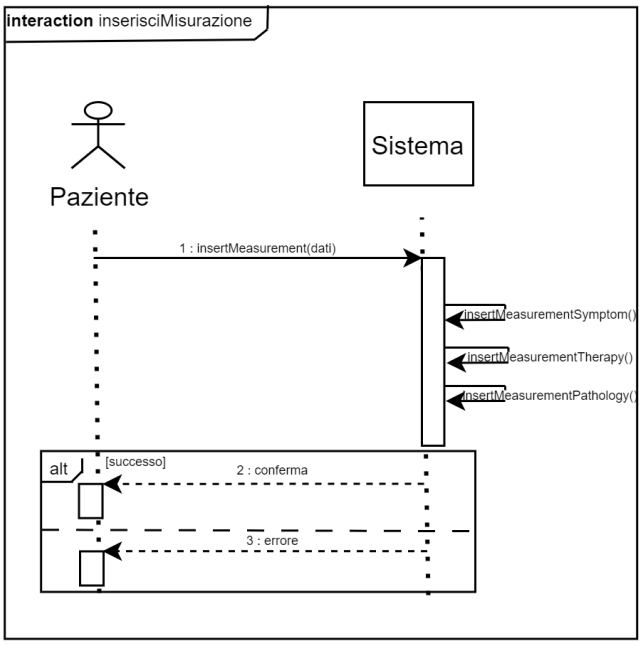


Immagine 4: Inserimento della misurazione

Registrare Assunzioni Giornaliere

Il paziente dovrà registrare le assunzioni giornaliere dei vari farmaci delle varie terapie a lui assegnate. Il sistema in automatico registrerà anche giorno e ora dell'assunzione, in modo da poter effettuare i giusti controlli sul paziente.

Attori: *Paziente*
Precondizioni: *Il paziente deve essersi autenticato*
Passi:
1. *Il paziente accede al sistema*
2. *Il paziente è introdotto all'interfaccia di base*
3. *Il paziente seleziona la terapia*
4. *Il paziente inserisce la quantità di farmaco assunta*
5. *Il paziente pigia sul pulsante "INSERT INTAKE"*

Postcondizioni: *L'assunzione è inserita*

Contattare il proprio medico

Infine, deve essere data la possibilità al paziente di contattare il proprio medico in caso di qualsiasi tipo di problema o anche solo per chiedere delle informazioni.
Sarà presente un bottone "CONTACT" che, una volta cliccato, lo reindirizzerà alla casella di posta, inserendo nel destinatario la mail del medico che lo segue.

Attori: *Paziente*
Precondizioni: *Il paziente deve essersi autenticato*
Passi:
1. *Il paziente accede al sistema*
2. *Il paziente è introdotto all'interfaccia di base*
3. *Il paziente accede all'interfaccia di visualizzazione delle sue informazioni*
4. *Il paziente pigia sul pulsante "CONTACT"*

Postcondizioni: *L'applicativo "Mail" viene aperto*

Diagrammi di attività

Di seguito sono riportati i diagrammi di attività per l'autenticazione, del medico e del paziente. I due attori, per eseguire qualsiasi azione, devono essere autenticati. Una volta effettuato l'accesso hanno la possibilità di eseguire determinate attività, in base al ruolo che ricoprono.

Nota: *Si è deciso di non mostrare la possibilità di ripetere più volte la stessa operazione per semplici ragioni di chiarezza della lettura. Quindi, a meno che non venga eseguito un logout, è possibile eseguire le altre azioni ciclicamente.*

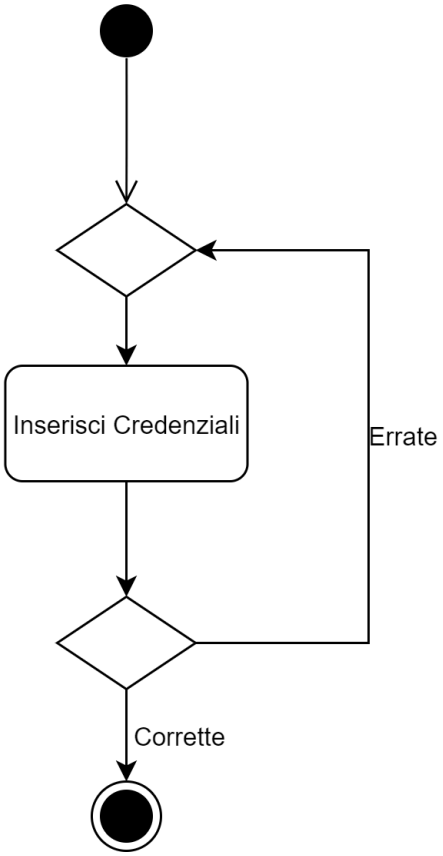


Immagine 5: Autenticazione

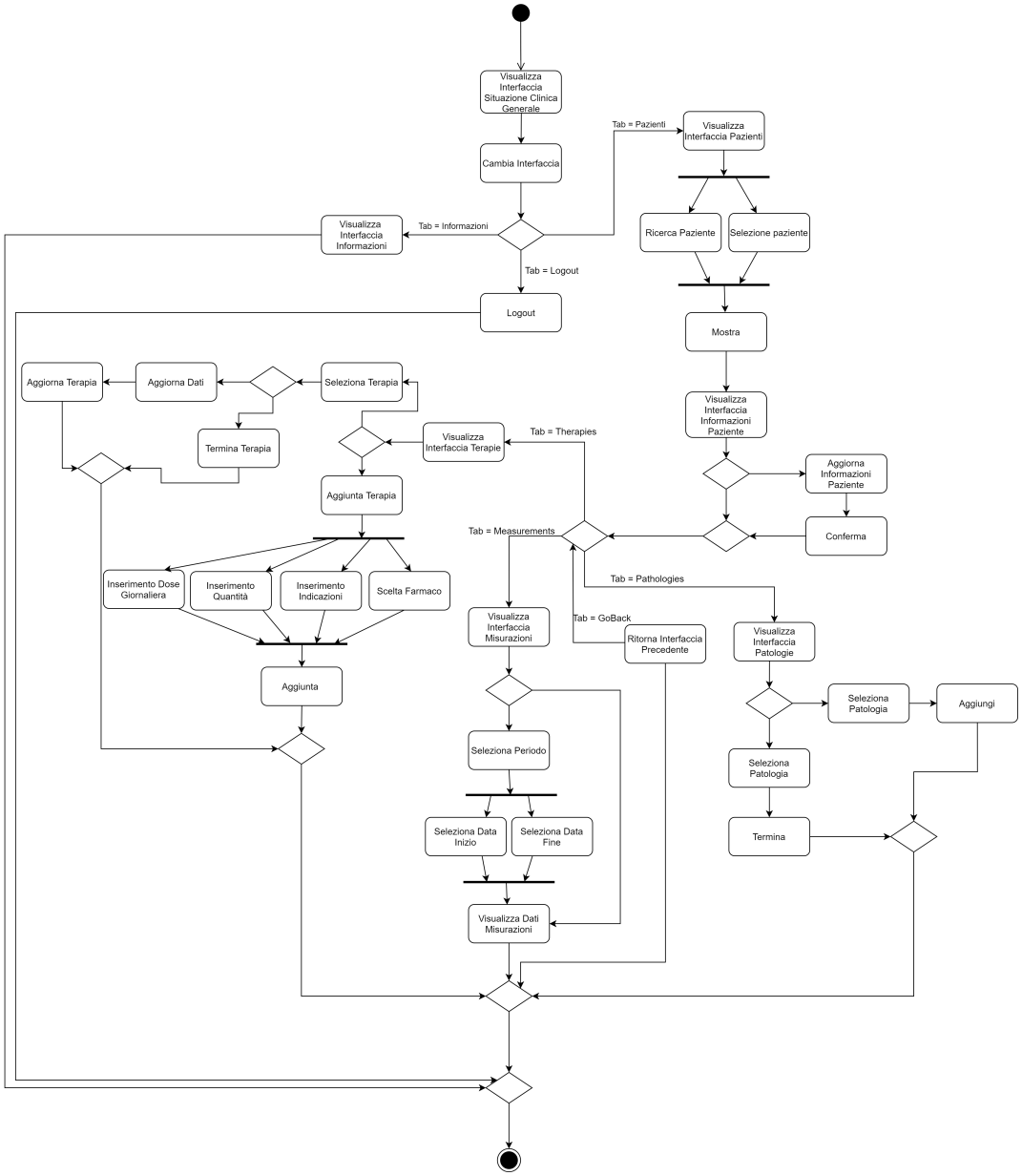


Immagine 6: Attività Medico

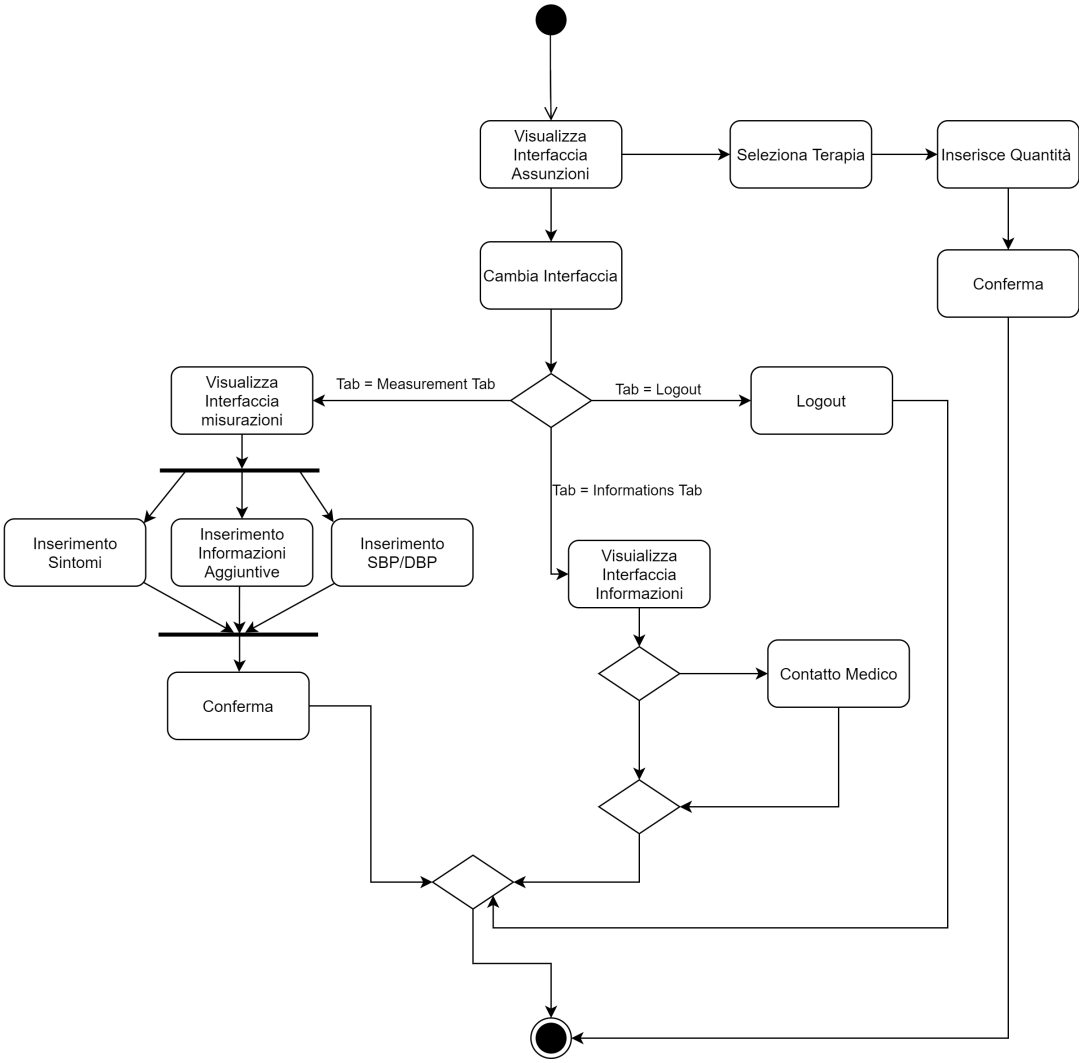


Immagine 7: Attività Paziente

Progettazione e sviluppo del sistema

Note sullo sviluppo

Il processo di sviluppo del software è stato affiancato dall'utilizzo della piattaforma web GitHub. Grazie a quest'ultimo è stato possibile utilizzare il sistema di controllo delle versioni Git e sviluppare, in parallelo, diverse parti del codice. Inoltre, per la collaborazione remota, ci siamo avvalsi del plugin *CodeTogether*.
Il piano di sviluppo adottato è stato principalmente di tipo ***Incrementale***, basato su una organizzazione ***Plan Driven***.
Inizialmente è stata condotta la fase di analisi delle specifiche richieste, in modo da riuscire a dividerle in sottoproblemi per cercare di lavorare al meglio e, quindi, di iniziare l'implementazione nella miglior maniera possibile. Successivamente, sono stati generati i vari use-cases e i diagrammi di attività.
Poi si è passati alla stesura del codice e ad ogni modifica significativa sono stati effettuati svariati test. Ovviamente, questo ha portato anche ad includere quella che è l'attività di refactoring sul codice già scritto, per risolvere i problemi che erano sorti.
Al termine di ogni versione è stata prodotta la documentazione riguardante ciò che era stato implementato. Quindi è stato raccolto il materiale UML, composto dai diagrammi di classe, anche aggiungendo alcuni UML descrittivi (presenti in questa documentazione).

Pattern MVC

Per la creazione del software si è utilizzato il pattern architetturale **MVC** (Model View Controller), supportato dalle librerie di JavaFX. E' stata effettuata questa scelta perchè il pattern si basa sulla separazione, netta, di tre componenti all'interno del software. Ad ogni componente è stato assegnato un package differente dagli altri. Le componenti sono:

- **Model**: fornisce i metodi per accedere ai dati e alle varie informazioni memorizzate.
- **View**: si occupa di rappresentare visivamente il modello all'utente. E' il componente con cui l'utente può interagire.
- **Controller**: riceve i comandi dell'utente mediante la View e li attua, talvolta modificando anche lo stato degli altri due componenti.

In generale, la separazione delle componenti MVC aiuta a garantire che il software sia facilmente manutenibile e modificabile.

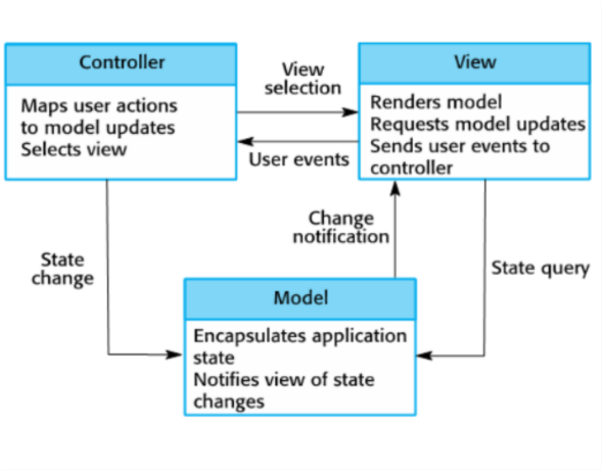


Immagine 8: Pattern MVC

Model

Il Model contiene e gestisce tutte le classi che ci permettono di operare e gestire i dati a noi interessati.
Per il salvataggio vero e proprio di tutte le varie entità ci siamo appoggiati ad un DBMS relazionale molto semplice da utilizzare, SQLite3. Difatti, sono state create entità identificate da chiavi primarie e legate fra loro da chiavi esterne.
Questo ci ha permesso di inserire all'interno del Model, e più precisamente nella classe DB_Model, tutta la logica di interazione basata su query SQL e gestione di oggetti OOP (Objected-Oriented Programming).

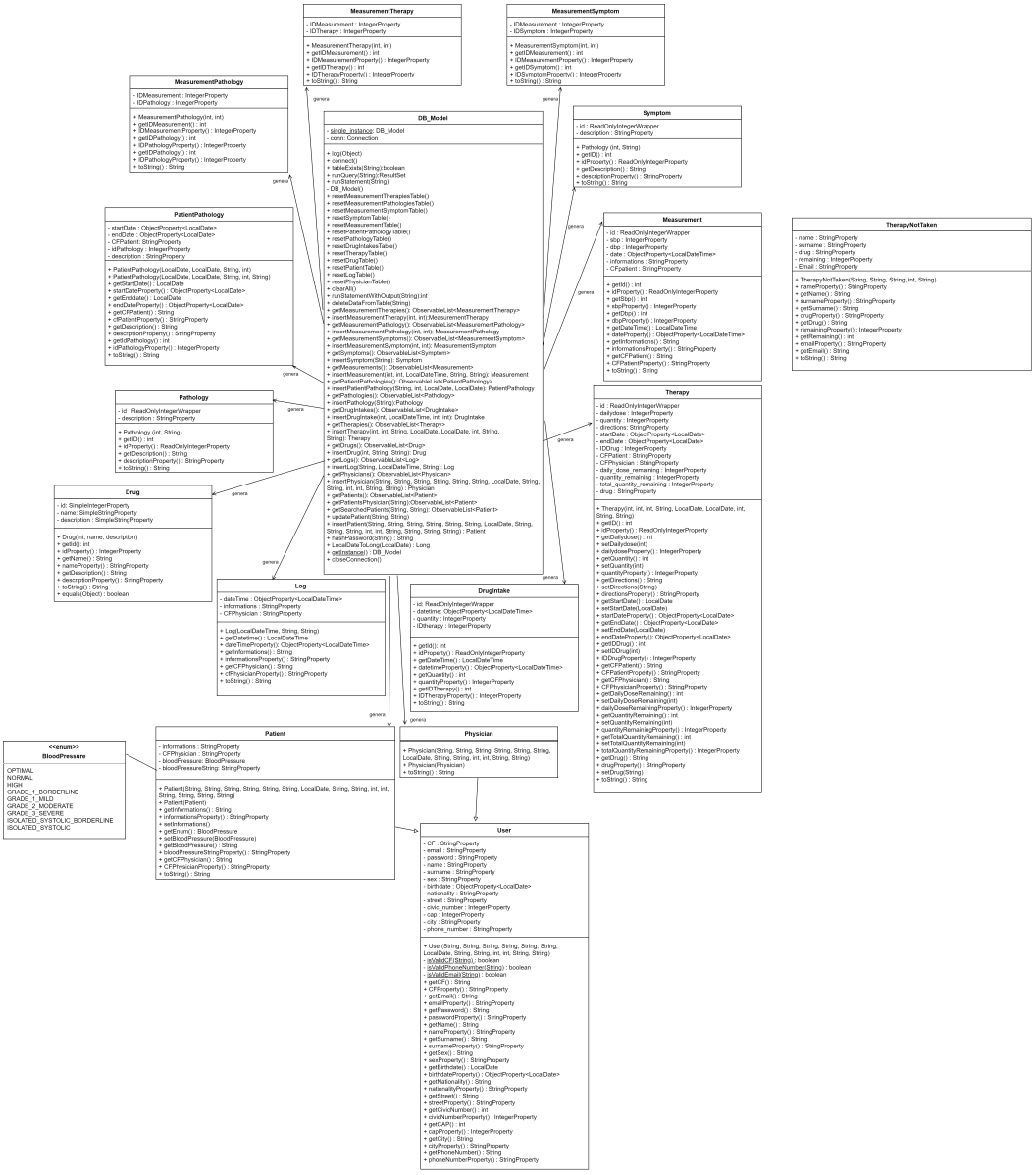


Immagine 9: Diagramma delle classi del modello

View

La View rappresenta la vista, ovvero tutto ciò che interagisce con il nostro utente. Avendo utilizzato il framework JavaFX, la nostra View è composta da file .fxml.

Controller

Il Controller è quella componente che permette di legare le azioni dell'utente sulla View alle rispettive modifiche da attuare sul database tramite il Model.
Oltre a questo, il controller può anche modificare le viste in base alle richieste dell'utente. Difatti, ogni vista è legata ad un controller.

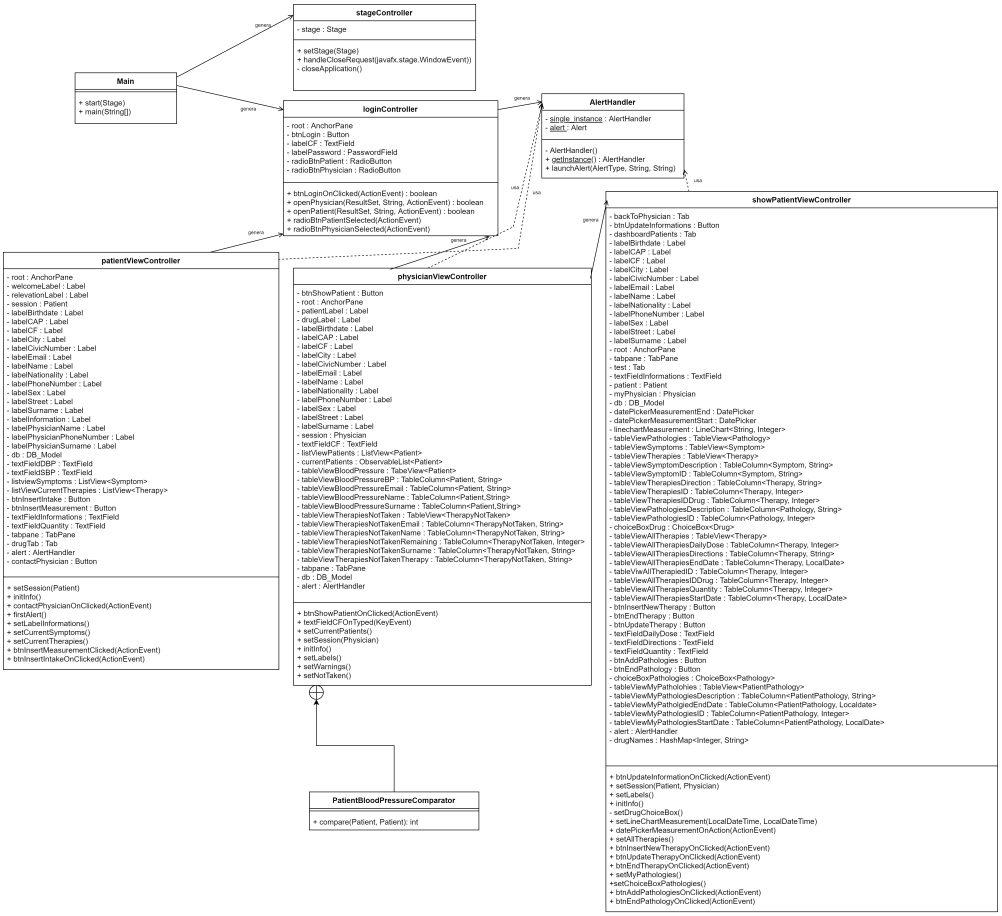


Immagine 10: Diagramma delle classi della vista-controllore

Design pattern utilizzati

In questa sezione vengono discussi i vari design pattern utilizzati:

- **Iterator Pattern**: utilizzato in modo implicito dato che è alla base della programmazione Java.
- **Fadade Pattern**: la classe DB_Model propone dei metodi che nascondono ai controller la complessità del sistema.
- **Observer Pattern**: meccanismo implementato nativamente nei componenti nel framework JavaFX, è alla base del binding tra azioni sulla View e funzioni del Controller.
- **Singleton Pattern**: adottato per le classi DB_Model e Alert, ovvero oggetti utilizzabili con una singola istanza globale.

Diagrammi di sequenza del software

Successivamente sono rappresentati alcuni diagrammi di sequenza che mostrano alcune interazioni, fra classi, fondamentali per il corretto funzionamento del software.

Nel login l'utente inserirà le credenziali, oltre a selezionare se si tratta di un paziente o un medico, e pigerà il pulsante. Le credenziali saranno confrontate dal sistema con i dati presenti nel database, e dopodichè ci sarà un riscontro.
Da notare che, per ragioni di sicurezza, la password salvata nel database è rielaborata con delle funzioni di hash.
La userView.fxml non esiste nel software ma è stata inserita per semplificare lo schema. Nella pratica il controller riconosce il vero ruolo dell'attore (medico o paziente) e caricherà la view corrispettiva.

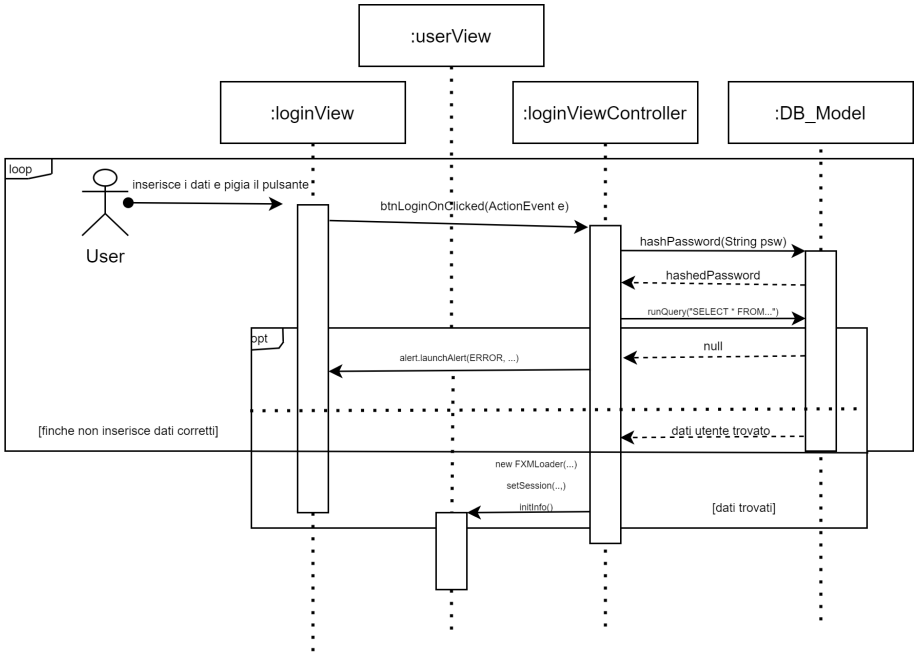


Immagine 11: Diagramma di sequenza del login

Il paziente, all'accesso, riceverà un alert che mostrerà tutti i farmaci che deve ancora assumere giornalmente. Ovviamente i conteggi sono fatti basandosi sulle terapie assegnate, all'attore, da parte del suo medico.

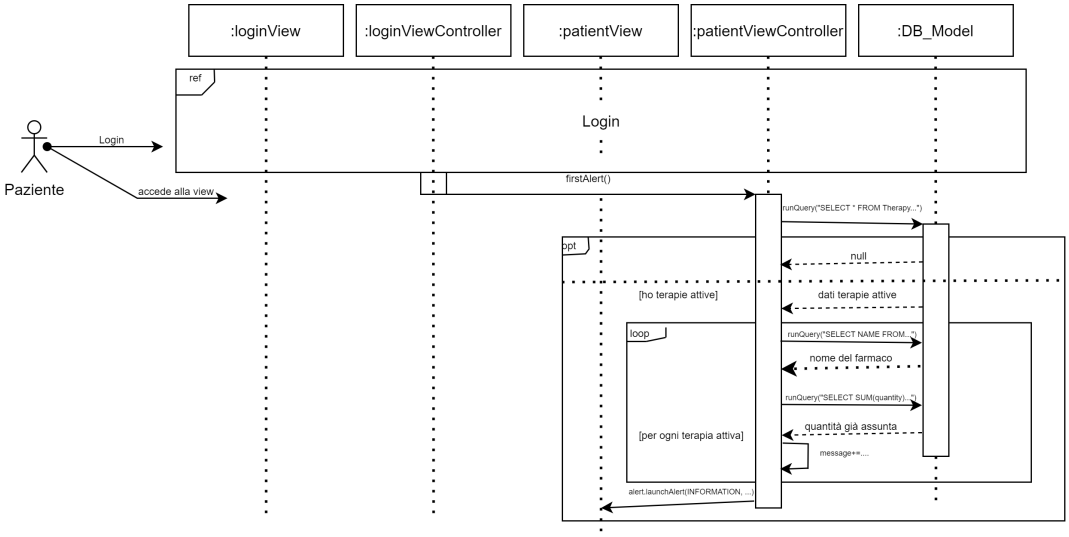


Immagine 12: Diagramma di sequenza degli avvisi ricevuti dal paziente al login

La funzionalità dell'aggiunta di una misurazione è specifica dei pazienti. E' possibile inserire misurazioni infinite.

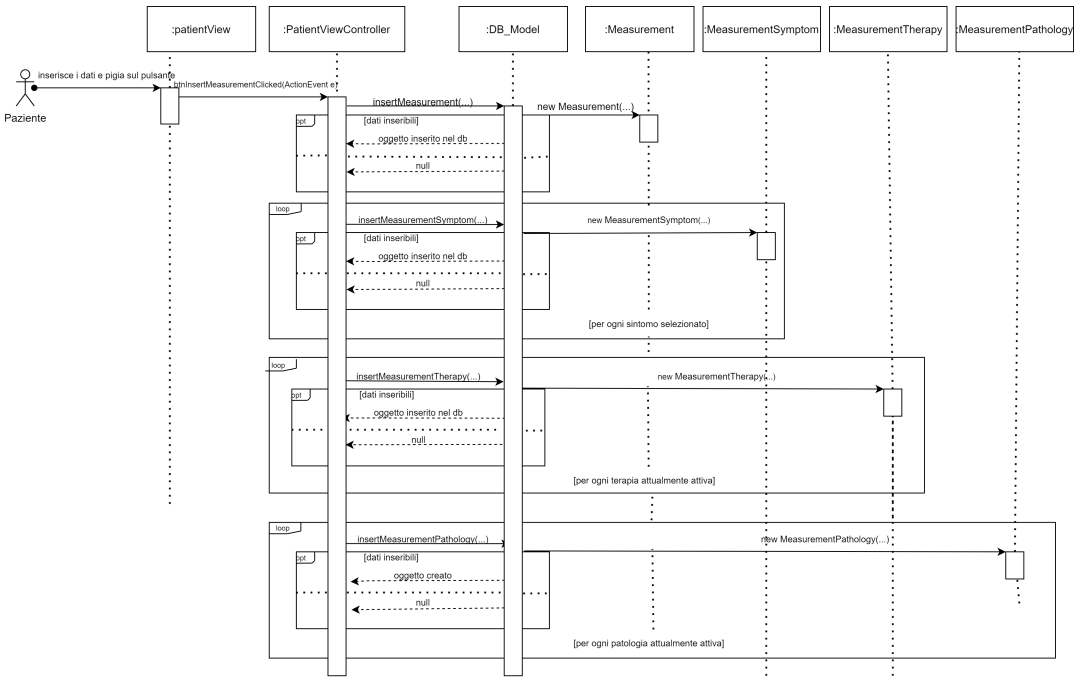


Immagine 13: Diagramma di sequenza dell'inserimento di una misurazione

Attività di Testing

I test sono stati fondamentali per capire cosa era incorretto all'interno del software. Sono stati svolti dei test sia a livello software che a livello pratico.
Si sono svolte le seguenti attività:

- Test con JUnit
- Test da parte degli sviluppatori
- Test da parte di utenti generici
- Test da parte di utente del settore

Test con JUnit

Abbiamo utilizzato **JUnit** per testare alcune delle funzionalità del software, selezionandole in base alla loro importanza per il corretto funzionamento del sistema.
Le funzioni principalmente testate riguardano il **DBModel** e sono:

- **testConnect()**
  Verifica che la connessione al database abbia avuto successo.
- **testTableExists()**
  Verifica che la tabella sia realmente presente all'interno del database.
- **testRunQuery()**
  Verifica che la query eseguita ritorna degli elementi e non *null*.
- **testGetInstance()**
  Verifica che il DBModel esista realmente.

Test degli sviluppatori

Anche noi sviluppatori abbiamo compiuto svariate azioni per testare al meglio il funzionamento dell'applicazione, e per vedere se il comportamento fosse quello previsto. Alcuni dei test svolti sono:

- **Verifica del corretto funzionamento dell'autenticazione**
  Abbiamo provato ad effettuare il login sia con dati presenti nel database sia con dati non presenti. Si è tentato anche di accedere ad un medico con delle credenziali di un paziente e viceversa.
- **Test delle TextField**
  Verificata la correttezza nell'inserimento dati. Per le TextField che si aspettano valori numerici, sono stati aggiunti degli ActionListeners che vanno ad eliminare il contenuto se non è considerato numerico. Invece per le TextField che si aspettano stringhe, abbiamo verificato l'annullamento delle operazioni con l'aggiunta di Alert in caso di inserimenti nulli.
- **Test delle ListView e ChoiceBox**
  Verificata la correttezza nelle azioni di selezione. Una qualsiasi operazione che richiede una selezione viene annullata in caso di selezione nulla.
- **Prova dei vari componenti a database vuoto**
  Si riesce a gestire la non presenza di dati nei principali componenti (lineChart, tableView, listView, choiceBox).
- **Verificato** che il **medico** possa modificare e vedere informazioni solamente dei suoi pazienti, oltre ad aggiungere terapie e patologie. Però, potrà vedere tutti i gradi ipertensione di tutti i pazienti.
- **Verificato il sistema degli avvisi** mostrati a video all'accesso da parte dei pazienti.

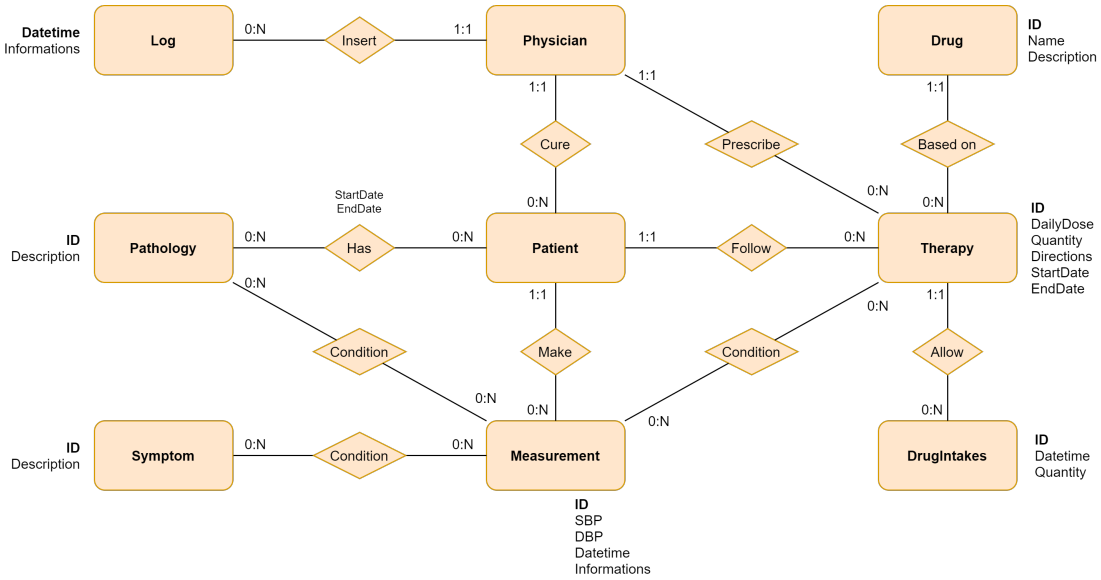
Test di utenti generici

Il software è stato sottoposto ad un test da parte di diverse persone completamente distaccate dallo sviluppo, anche con conoscenze informatiche limitate. Non si è voluto in nessun modo spiegare l'utilizzo del software, in modo che il test, non guidato, ci permettesse di ottenere risultati utili e non influenzati. Ci siamo limitati solo a spiegare, a grandi linee, i fini del sistema, senza dare spiegazioni, se non richiesto con delle domande.
Tutto questo ci ha permesso di rilevare degli errori che noi sviluppatori non eravamo riusciti a trovare. Inoltre, ci sono state proposte alcune nuove funzionalità, che miravano a migliorare la *user experience*.

Test di utente del settore

Infine il software è stato sottoposto anche ad un parente (medico) che lavora nel settore trattato all'interno del progetto. Anche questo test è stato effettuato con le modalità descritte nel paragrafo precedente.
Da questo abbiamo ricevuto delle critiche costruttive che, però, ci hanno permesso di migliorare l'interfaccia del medico nel miglior modo possibile.

Schema ER della base di dati



DB Data for table 'Physician':
- CF
- Email
- Password
- Name
- Surname
- Sex
- Birthdate
- Nationality
- Street
- CivicNumber
- CAP
- City
- PhoneNumber

DB Data for table 'Patient':
- CF
- Email
- Password
- Name
- Surname
- Sex
- Birthdate
- Nationality
- Street
- CivicNumber
- CAP
- City
- PhoneNumber
- Informations

Immagine 14: Schema ER