



UNIVERSITA' DEGLI STUDI DI
NAPOLI FEDERICO II

Scuola Politecnica e delle Scienze di Base

Corso di Laurea in Ingegneria Informatica

Tesina finale in Progettazione e Sviluppo di Sistemi Software

OspedApp: Sistema di gestione di cartelle ospedaliere

Anno Accademico 2019/2020



Prof.ssa

Fasolino Anna Rita

Candidati:

Benfenati Domenico

Coppola Vincenzo

De Vivo Simona

Della Torca Salvatore

Esposito Salvatore

Indice

Presentazione del Progetto.....	5
Capitolo 1: Descrizione del progetto.....	7
Processo di Sviluppo	7
Gestione del team e suddivisione del lavoro	7
Riunioni di progetto.....	7
Stima dei costi di sviluppo	7
Primo workshop dei requisiti	9
Secondo workshop dei requisiti	9
Terzo workshop dei requisiti	9
Quarto workshop dei requisiti.....	10
Strumenti utilizzati	10
Capitolo 2: Specifica dei requisiti	12
Introduzione	12
Obiettivi del sistema	12
Descrizione testuale dei requisiti.....	12
Descrizione generale	12
Prospettive del prodotto.....	12
Opportunità di business.....	13
Demografia di mercato	13
Dichiarazione dei problemi	13
Descrizione delle parti interessate.....	13
Obiettivi di alto livello e problemi delle parti interessate	14
Requisiti funzionali.....	14
Requisiti non funzionali.....	15
Regole di business.....	16
Capitolo 3 : Glossario dei termini.....	17
Capitolo 4 : Analisi dei requisiti.....	19
Identificazione degli attori.....	19
Diagramma dei casi d'uso.....	19
Descrizione breve dei casi d'uso.....	19
Caso d'uso 1 : Effettua login	19
Caso d'uso 2 : Creazione di una cartella clinica	20
Caso d'uso 3 : Visualizza cartella clinica attiva.....	20
Caso d'uso 4 : Aggiorna cartella clinica.....	20

Caso d'uso 5 : Visualizza cartelle paziente	20
Caso d'uso 6 : Richiesta esame	20
Tabella Attori – Obiettivi	20
Descrizione di dettaglio dei casi d'uso	21
Caso d'uso 1 : Effettua login	21
Caso d'uso 2 : Creazione di una cartella clinica	21
Caso d'uso 3 : Visualizza cartella clinica attiva.....	22
Caso d'uso 4 : Aggiorna cartella clinica.....	23
Caso d'uso 5 : Visualizza cartelle paziente.....	24
Caso d'uso 6 : Richiesta esame	25
Modello di dominio del sistema	26
Sequence diagram di analisi	27
Effettua login.....	27
Creazione di una cartella clinica.....	28
Visualizza cartella clinica attiva.....	28
Aggiorna cartella clinica	28
Visualizza cartelle paziente	29
Richiesta esame	29
Activity diagram di analisi.....	30
Effettua login.....	30
Creazione di una cartella clinica.....	30
Visualizza cartella clinica attiva.....	31
Aggiorna cartella clinica	32
Visualizza cartelle paziente	33
Richiesta esame	33
Capitolo 5 : Architettura Software.....	34
Vista architetturale di alto livello	34
Vista architetturale di alto livello lato client	34
Vista architetturale di alto livello lato server	35
Diagramma dei componenti	35
Descrizione dell'architettura nel dettaglio	36
Diagramma architetturale di dettaglio lato client	36
Diagramma architetturale di dettaglio lato server	37
Diagrammi di sequenza di dettaglio	38
Sequence diagram di dettaglio – Effettua Login.....	39

Sequence diagram di dettaglio – Creazione cartella clinica.....	40
Sequence diagram di dettaglio – Visualizza cartella clinica	41
Sequence diagram di dettaglio – Aggiorna cartella clinica	41
Capitolo 6 : Persistenza dei dati	42
Definizione delle relazioni	43
Traduzione in tabelle	44
Capitolo 7 : Implementazione	46
Diagramma di deploy	46
Documenti di implementazione	46
Servizi utilizzati	47
Capitolo 8 : Manuale di utilizzo del software.....	48
Configurazione del database	48
Avvio	55
Server	55
Client	55
Screen delle interfacce.....	55
Creazione di una guida interattiva tramite DrExplain	59
Capitolo 9 : Testing.....	62
Capitolo 10 : Sviluppi futuri.....	65

Presentazione del Progetto

Il progetto prevede la realizzazione di un software di gestione e di aggiornamento di cartelle cliniche elettroniche, con il fine di aiutare il personale medico a conservare e organizzare i dati di un paziente.

Il software permette l'interazione di medici e infermieri attraverso differenti interfacce che permettono di effettuare funzioni specifiche per i pazienti ricoverati in ospedali che sfruttano il suddetto software.

Di seguito la cronologia delle revisioni effettuate al documento.

Versione	Data	Descrizione	Autore
Prima bozza del documento	Ago 23, 2020	Prima scrittura del documento.	SDVSS DevTeam
Primo raffinamento	Ago 25, 2020	Raffinamento della bozza iniziale.	SDVSS DevTeam
Aggiornamento 1	Ago 26, 2020	Definizione degli obiettivi del sistema e descrizione dei requisiti funzionali e non funzionali.	SDVSS DevTeam
Aggiornamento 2	Ago 26, 2020	Dichiarazione dei problemi e descrizione delle parti interessate.	SDVSS DevTeam
Aggiornamento 3	Ago 26, 2020	Definizione obiettivi di alto livello e problemi delle parti interessate.	SDVSS DevTeam
Aggiornamento 4	Ago 26, 2020	Aggiornamento della descrizione generale del prodotto.	SDVSS DevTeam
Aggiornamento 5	Ago 27, 2020	Descrizione breve dei casi d'uso.	SDVSS DevTeam
Aggiornamento 6	Ago 28, 2020	Raffinamento dei casi d'uso già presenti e inserimento di nuovi casi d'uso utili al funzionamento del sistema.	SDVSS DevTeam
Aggiornamento 7	Ago 29, 2020	Descrizione dettagliata dei casi d'uso.	SDVSS DevTeam
Aggiornamento 8	Ago 29, 2020	Inserimento di altre sezioni per la descrizione generale del prodotto.	SDVSS DevTeam
Aggiornamento 9	Ago 30, 2020	Inserimento del glossario.	SDVSS DevTeam
Aggiornamento 10	Ago 30, 2020	Raffinamento del documento.	SDVSS DevTeam
Aggiornamento 11	Set 1, 2020	Inserimento di nuove sezioni e aggiunta di grafici rappresentativi del sistema.	SDVSS DevTeam
Aggiornamento 12	Set 1, 2020	Raffinamento della sezione relativa ai casi d'uso.	SDVSS DevTeam

Aggiornamento 13	Set 1, 2020	Inserimento dei diagrammi prodotti nella prima iterazione.	SDVSS DevTeam
Aggiornamento 14	Set 7, 2020	Inserimento dei diagrammi prodotti nella seconda iterazione.	SDVSS DevTeam
Aggiornamento 15	Set 13, 2020	Inserimento dei diagrammi prodotti nella terza iterazione.	SDVSS DevTeam
Aggiornamento 16	Set 18, 2020	Inserimento dei diagrammi prodotti nella quarta iterazione.	SDVSS DevTeam
Aggiornamento 17	Set 18, 2020	Raffinamento del documento e inserimento della sezione relativa alla configurazione.	SDVSS DevTeam
Aggiornamento 18	Set 19, 2020	Inserimento della sezione relativa al testing del sistema.	SDVSS DevTeam
Aggiornamento 19	Set 20, 2020	Inserimento del capitolo conclusivo e correzione finale del documento.	SDVSS DevTeam

Capitolo 1: Descrizione del progetto

Processo di Sviluppo

Per la realizzazione del progetto ci si è avvalsi di tecniche di sviluppo agili, per far fronte alle possibili modifiche in corso d'opera e per ridurre al minimo i possibili fallimenti.

In particolare, è stata adottata una tecnica di sviluppo incrementale e iterativo basata su UP (*Unified Process*), realizzata nell'arco di circa sei settimane.

Gestione del team e suddivisione del lavoro

Il team si compone di cinque membri. Ogni fase di sviluppo si è svolta in gruppo per avere la massima efficienza possibile per lo sviluppo dei vari aspetti del progetto.

Riunioni di progetto

Lo sviluppo dell'intero progetto ha richiesto un tempo di circa sei settimane. Il team si è riunito a cadenza giornaliera per circa 5 ore al giorno, per un totale di 5-6 giorni a settimana, arrivando ad una media di circa 25-30 ore settimanali per ogni membro del team.

In dettaglio, di seguito, le ore spese per lo sviluppo del progetto:

<i>Attività svolta</i>	<i>Sett.1</i>	<i>Sett.2</i>	<i>Sett.3</i>	<i>Sett.4</i>	<i>Sett.5</i>	<i>Sett.6</i>	<i>Totale ore</i>	<i>Totale ore %</i>
<i>Studio di fattibilità</i>	20						20	2.6%
<i>Analisi requisiti</i>	45	20	20	10			95	12.5%
<i>Progettazione</i>	45	65	55	40	45	40	290	38.2%
<i>Implementazione</i>		35	40	40	65	65	245	32.3%
<i>Test</i>				25	15	15	55	7.2%
<i>Gestione documentazione</i>	10	5	15	10	5	10	55	7.2%
<i>Totale ore</i>	120	125	130	125	130	130	760	100%

Stima dei costi di sviluppo

Per il calcolo dei costi di sviluppo del progetto, ci si è basati sulla tecnica di calcolo degli *Use Case Point*. Dopo un'attenta analisi sui possibili use-case del progetto si è fatta una stima dei differenti

indici di stima: come prima approssimazione si è proceduto al calcolo dell'indicatore **UUCP** (*Unadjusted Use Case Point*):

$$UUCP = UUCW + UAW = 50 + 9 = 59.$$

Successivamente sono stati calcolati il fattore di complessità tecnica **TFC** e il fattore ambientale **EF**:

$$TFC = 0,6 + (0,01 * TFactor) = 0,6 + (0,01 * 28) = 0,88$$

$$EF = 1,4 + (-0,03 * EFactor) = 1,4 + (-0,03 * 32) = 0,44$$

Moltiplicando gli indicatori calcolati si ricava l'indicatore **UCP** (*Use Case Point*):

$$UCP = UUCP * TFC * EF = 22,85.$$

Supponendo un rapporto di 25 ore per Use Case Point si ottiene un quantitativo di ore pari a:

$$Ore = 25 * 22,85 \cong 572 \text{ h.}$$

Di seguito le tabelle che mostrano graficamente i calcoli effettuati:

Complessità	Peso	Numero di Use Case	Prodotto
<i>Semplice</i>	5	0	0
<i>Medio</i>	10	2	20
<i>Complesso</i>	15	2	30
Totale		4	50

Tipologia Attore	Peso	Numero Attori	Prodotto
<i>Semplice</i>	1	0	0
<i>Medio</i>	2	0	0
<i>Complesso</i>	3	3	9
Totale		3	9

Fattore	Descrizione	Peso	Valutazione	Prodotto
<i>T1</i>	Sistema distribuito	2.0	1	2
<i>T2</i>	Performance	1.0	4	4
<i>T3</i>	Efficienza per l'utente finale	1.0	4	4
<i>T4</i>	Complessità di elaborazione interna	1.0	2	2
<i>T5</i>	Riusabilità del codice	1.0	1	1
<i>T6</i>	Facile da installare	0.5	1	0.5
<i>T7</i>	Facile da usare	0.5	5	2.5
<i>T8</i>	Portabile	2.0	0	0
<i>T9</i>	Facile da cambiare	1.0	1	1
<i>T10</i>	Elaborazione parallela	1.0	1	1
<i>T11</i>	Caratteristiche di sicurezza	1.0	5	5
<i>T12</i>	Accesso diretto da terzi	1.0	1	1
<i>T13</i>	Formazione per utenti finali	1.0	4	4
Totale				28

Fattore	Descrizione	Peso	Valutazione	Prodotto
<i>E1</i>	Familiarità con il processo di sviluppo	1.5	5	7.5
<i>E2</i>	Esperienza di applicazione	0.5	3	1.5

E3	Esperienza object-oriented del team	1.0	5	5
E4	Capacità di analisi	0.5	4	2
E5	Motivazione del team	1.0	5	5
E6	Stabilità dei requisiti	2.0	3	6
E7	Personale part-time	-1.0	1	-1
E8	Complessità del linguaggio utilizzato	2.0	3	6
Totale				32

Primo workshop dei requisiti

Nella prima settimana si è tenuto il primo workshop dei requisiti, durante il quale sono stati stabiliti gli obiettivi della prima iterazione ed è stata effettuata una raccolta di requisiti funzionali di alto livello. Sono stati poi identificati i principali requisiti non funzionali che il sistema deve rispettare.

Nella prima iterazione si è scelto di concentrarsi sui casi d'uso **"Effettua Login"** e **"Crea Cartella Clinica"**.

Il primo permette l'autenticazione, mentre il secondo permette ad un medico, precedentemente autenticato, di creare una cartella clinica per un paziente ricoverato. I motivi per i quali si è giunti a questa scelta di progetto sono molteplici:

- Il login è un'operazione fondamentale per garantire il funzionamento del sistema: senza autenticazione l'utente non potrà eseguire nessun tipo di operazione;
- La creazione e l'inserimento della cartella clinica associata ad un paziente rappresentano le principali funzionalità del sistema: una volta inserita la cartella i medici e gli infermieri possono visualizzarla ed eventualmente aggiornarla;
- Sia il login che la creazione di una cartella clinica richiedono la comunicazione con un database per la memorizzazione dei dati. È stata quindi necessario ideare una soluzione di persistenza dei dati, e la successiva implementazione di una soluzione per effettuare il collegamento con il sistema.

Secondo workshop dei requisiti

A partire dalla terza settimana il team ha raggiunto tutti gli obiettivi della prima iterazione e ha iniziato la seconda iterazione, fase in cui è stato ultimato il materiale del primo workshop. È stato poi scelto un altro caso d'uso da analizzare: **"Visualizza Cartella Clinica Attiva"**.

- Il caso d'uso permette al medico di visualizzare le cartelle cliniche relative ad uno specifico paziente, anche quelle non attive.

Terzo workshop dei requisiti

Durante la prima parte della quinta settimana si è svolta una riunione in cui il team ha rivisto e raffinato ciò che è stato prodotto nella seconda iterazione, dando poi inizio alla terza iterazione in cui si è analizzato un quinto caso d'uso **"Aggiorna Cartella Clinica"**.

Questo caso d'uso permette al medico di aggiornare la cartella clinica di un paziente aggiungendo una nuova terapia.

Quarto workshop dei requisiti

Durante l'ultima settimana si è svolta l'ultima iterazione in cui sono stati aggiustati tutti i documenti prodotti fino a quel momento, sono stati risolti gli errori di codice e successivamente il team ha analizzato i casi d'uso **"Richiedi Esame"** e **"Visualizza cartelle paziente"**; in particolare:

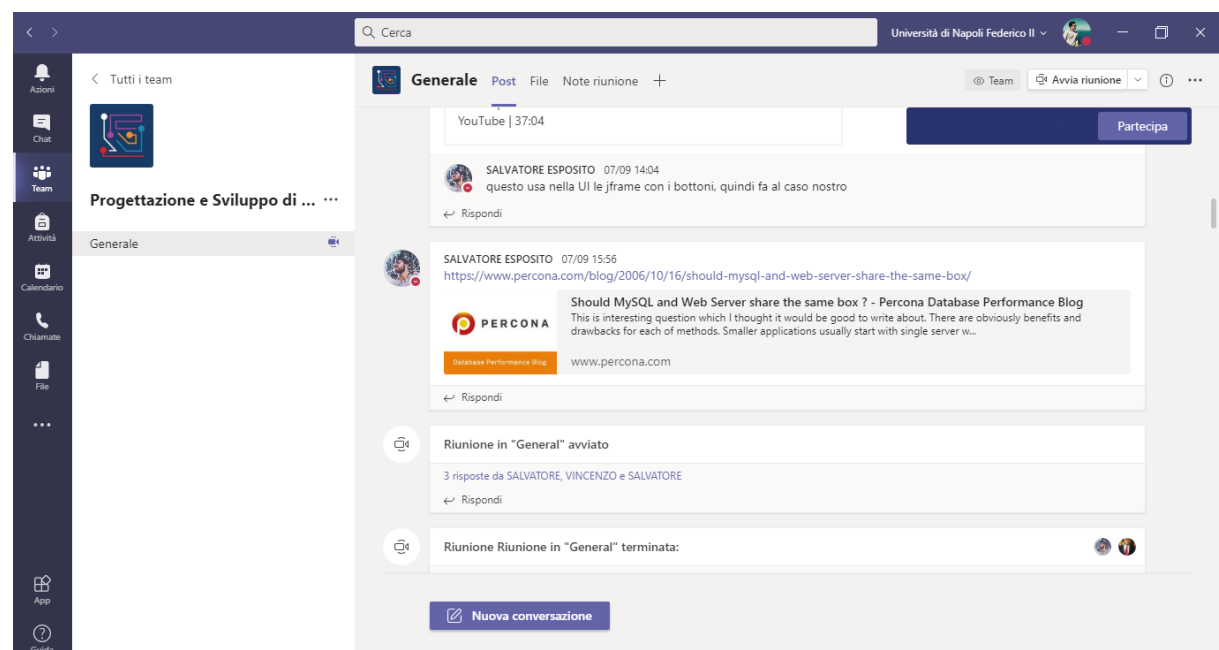
- Il primo permette ad un medico di effettuare la prenotazione un esame per un paziente;
- Il secondo caso d'uso permette all'infermiere di vedere solo la cartella clinica attiva di un paziente.

Strumenti utilizzati

Per far fronte all'emergenza sanitaria, nonché alla considerevole distanza tra i luoghi di residenza dei membri del team, le riunioni sono state effettuate utilizzando sistemi di supporto digitale.

Lo strumento che ha permesso al team di lavorare da remoto è **Microsoft Teams**, all'interno del quale è stato possibile creare uno spazio di lavoro condiviso che ha permesso al team di lavorare in mobilità e in maniera collaborativa.

Di seguito alcune istantanee dello spazio di lavoro:



Per lo sviluppo dell'applicazione sono stati utilizzati ulteriori strumenti di supporto:

- **Eclipse**: ambiente di sviluppo completo per sviluppare applicazioni di diverso tipo, sfruttando differenti linguaggi di programmazione, in particolare il team ha scelto di utilizzare il linguaggio Java per lo sviluppo del software.

- **Window Builder:** plug-in dell'IDE utilizzato, che permette di sviluppare una user-interface in maniera grafica, aggiungendo bottoni, etichette e altri elementi utili. Questo tipo di plug-in sfrutta interessanti framework di java come Swing e AWT.
- **Database MySQL:** database di supporto al sistema, sviluppato secondo una struttura di tipo relazionale, con l'aiuto di un software grafico come **MySQL Workbench**, in particolare nella sua versione 8.0.
- **MySQL JConnector:** libreria di collegamento tra lato server e database. In particolare, si è scelto di sfruttare la versione 4.3.5 perché risultava maggiormente adatta al collegamento tra sistema e dati persistenti.
- **Java RMI:** connettore a chiamata remota implicita utilizzato per la comunicazione tra client e server.
- **Visual Paradigm:** software di supporto per i design di diagrammi UML relativi alla progettazione del sistema.
- **Microsoft Office:** piattaforma utilizzata nella sua forma collaborativa per la redazione della documentazione di supporto al sistema.

Capitolo 2: Specifica dei requisiti

Introduzione

Obiettivi del sistema

Nel seguente capitolo si specificano i requisiti del sistema software “OspedApp”.

Descrizione testuale dei requisiti

Si vuole realizzare un sistema software per la gestione di un ospedale, in particolare si vogliono gestire delle cartelle cliniche associate ai pazienti.

Al fine di effettuare l’accesso al sistema, l’utente inserisce i suoi dati. Il sistema poi verifica se l’utente esiste o meno e, una volta trovato, mostra la giusta interfaccia a seconda che si tratti di un medico o di un infermiere.

A seguito dell’operazione di login, l’infermiere vuole visualizzare la cartella clinica attiva di un paziente. In particolare, l’infermiere inserisce il codice fiscale per ricercare la cartella clinica attiva associata al paziente per il ricovero corrente. Trovata la cartella attiva, il sistema la mostra a video le informazioni tramite apposita interfaccia.

Il medico, dopo essersi autenticato, richiede un esame per il paziente ricoverato. Nella fattispecie, il medico inserisce i dati del paziente e il sistema esegue la ricerca del paziente. Terminata la ricerca, il medico richiede un esame e il sistema registra la richiesta effettuata e mostra a video un messaggio di conferma della prenotazione.

All’arrivo di un paziente nell’ospedale, il medico vuole creare una nuova cartella clinica inserendo i dati anagrafici del paziente, per effettuare il ricovero. Il sistema memorizza i dati inseriti restituendo un messaggio di conferma.

Il medico vuole effettuare l’aggiornamento di una cartella clinica di un paziente. In particolare, egli inserisce il codice fiscale del paziente e il sistema ricerca la cartella clinica attiva per quel paziente. Successivamente il sistema visualizza.

Il medico inoltre, vuole visualizzare tutte le cartelle cliniche relative ad un paziente, attive e non. Inserendo il codice fiscale del paziente, il sistema mostra su video tutte le cartelle cliniche del paziente e il medico sceglie quale visualizzare.

Descrizione generale

Prospettive del prodotto

Lo scopo di questo progetto è la realizzazione di un particolare tipo di software, OspedApp, che si cala in un ambito sanitario per aiutare il personale medico a conservare e organizzare particolari dati di un paziente gestibili attraverso delle cartelle cliniche elettroniche. Volendo rispettare vincoli

di robustezza e adattabilità, l'obiettivo è quello di includere sistemi di interfacce multiple e integrare sistemi di supporto di terze parti.

Opportunità di business

OspedApp è pensato per accompagnare il lavoro del personale sanitario semplificando la gestione dei ricoveri. I prodotti già esistenti non sono adattabili all'attività dei medici, in termini di regole di business variabili e design di rete variabili. C'è quindi una certa insoddisfazione del mercato per questa inflessibilità, e la richiesta di un applicativo che corregga questo stato è estremamente frequente.

Demografia di mercato

Il prodotto è destinato in primis alle strutture ospedaliere, ma può essere utilizzato anche da strutture sanitarie come RSA o case di riposo, che ospitano persone non autosufficienti che non possono essere assistite in casa e che necessitano di specifiche cure mediche e di un'articolata assistenza sanitaria.

Dichiarazione dei problemi

I sistemi ospedalieri tradizionali spesso risultano rigidi, poco tolleranti ai guasti e difficili da integrare con sistemi di terze parti. Risultano dunque poco adatti alla gestione, nonché all'accesso ai dati personali di un paziente e poco performanti nell'esecuzione, in particolare, di operazioni come l'aggiornamento continuo di cartelle cliniche. Diventa necessario dunque apportare modifiche che, però, nella stragrande maggioranza dei casi non soddisfano le esigenze di mercato.

Descrizione delle parti interessate

Sommario delle parti interessate (non utenti)

- *Paziente*: persona che non interagisce col sistema, ma entra in ospedale per essere ricoverato e il personale medico ne tiene traccia mediante il software in questione.

Sommario degli utenti

- *Medico*: specialista in medicina che ha la facoltà di prenotare un esame, creare una cartella clinica ed effettuare delle modifiche aggiungendo terapie ed esami;
- *Infermiere*: professionista sanitario che ha la possibilità di visualizzare la cartella clinica di un determinato paziente.

Obiettivi di alto livello e problemi delle parti interessate

Obiettivo di alto livello	Attore	Priorità	Problemi e preoccupazioni	Soluzione corrente
Gestione della cartella clinica	Medico e infermiere	Alta	Ci si pone il problema di creare la cartella clinica di un paziente ricoverato, aggiornarla in caso di nuovi esami e trattamenti, permettere all'infermiere di visualizzare la cartella clinica attiva di un paziente e al medico di visualizzare tutte le cartelle cliniche.	Si prevede un'interfaccia specifica alla quale medico e infermiere possono accedere a seguito del login per effettuare la registrazione e la modifica di una cartella clinica.
Prenotazione di un esame	Medico	Media	Bisogna prenotare un esame per un paziente inserendo i dati anagrafici e la tipologia di esame al quale il paziente dovrà sottoporsi.	Registrare la visita di un paziente ed associarla alla sua specifica cartella.

Requisiti funzionali

Un utente, previa necessaria autenticazione tramite apposita interfaccia di Login, può eseguire diverse operazioni, a seconda del ruolo che ricopre all'interno dell'ospedale. L'utente infatti può:

- Effettuare l'accesso al sistema

Si è deciso poi di prevedere due differenti ruoli per l'utente registrato, quali medico e infermiere. In particolare, l'infermiere può:

- Visualizzare la cartella clinica di un paziente a seguito di una ricerca dello stesso per codice fiscale.

Il medico invece interagisce con il nostro sistema effettuando più operazioni. Infatti, egli può:

- Creare una nuova cartella clinica per un paziente, che sia esso già esistente o meno all'interno del sistema;
- Visualizzare lo storico di tutte le cartelle cliniche associate ad un paziente specifico;
- Aggiornare la cartella clinica attiva di un paziente a cui ne è stata associata già una;
- Effettuare la richiesta di un esame per uno specifico paziente.

Requisiti non funzionali

Sicurezza

Per evitare che chiunque possa effettuare operazioni attraverso il sistema, ogni utente che vuole interagire con esso deve essere autenticato.

Usabilità

Il cliente deve essere in grado di visualizzare le schermate dell'applicativo in maniera chiara. Quindi:

- Il testo deve risultare facilmente visibile anche ad un metro di distanza;
- Evitare colori associati a forme comuni di discromatopsie;

La velocità, la facilità e l'elaborazione senza errori sono fondamentali nella creazione e nell'aggiornamento di cartelle cliniche, in quanto si desidera non incappare in alcun errore di elaborazione di esse, dal momento che si potrebbero avere danni gravi alla salute del paziente.

Le informazioni devono essere trasmesse con una grafica user-friendly, in quanto deve risultare il più semplice possibile inserire le informazioni per evitare errori.

Affidabilità

Si garantisce la persistenza dei dati tramite sistemi esterni, in modo da garantire un corretto recupero delle informazioni anche in caso di fallimento del software.

Performance

Come menzionato alla voce fattori umani, i medici vogliono completare l'elaborazione delle cartelle molto rapidamente. Il nostro obiettivo è quello di permettere l'inserimento di nuove istanze in meno di 1 minuto, il 99% delle volte.

Supportabilità

- **ADATTABILITA'**: I diversi medici che usufruiscono di OspedApp hanno esigenze di elaborazione e regole di business uniche durante l'elaborazione. Dunque, in diversi punti definiti dello scenario (ad esempio, quando viene generata una nuova cartella, quando viene aggiunta una nuova voce in una cartella) verrà abilitata la regola di business collegabile ad esso.
- **CONFIGURABILITA'**: I differenti utilizzatori dell'applicazione hanno esigenze differenti ed eseguono diverse operazioni. Pertanto, il cliente si aspetta che ad ogni operatore corrisponda una configurazione di interfacce differenti.

Regole di business

<i>ID</i>	Regola	Modificabilità	Sorgente
<i>REGOLA1</i>	Nel codice fiscale le lettere vanno tutte in maiuscolo	Non è modificabile	Decreto del presidente della Repubblica n. 605 del 29 settembre 1973
<i>REGOLA2</i>	Il medico è l'unico a poter creare una cartella clinica	Non è modificabile	Ministero della Salute
<i>REGOLA3</i>	L'infermiere può visualizzare solo la cartella clinica attiva	Non è modificabile	Ministero della Salute
<i>REGOLA4</i>	Le date devono essere inserite nel formato GG/MM/AAAA	Non è modificabile	Ambiente di sviluppo

Capitolo 3 : Glossario dei termini

Per rendere chiari i concetti successivi si è deciso di redigere un glossario per specificare i termini utilizzati e per definire le entità in gioco. Di seguito la tabella con i termini individuati.

Termine	Definizione e informazioni	Formato	Regole di validazione	Alias
<i>Dati clinici</i>	Elenco di parametri vitali del paziente	Numerico o testuale		
<i>Cartella clinica</i>	Storico dei dati clinici del paziente raccolti nel periodo di degenza, con annessi dati anagrafici del paziente	Testuale	La cartella clinica è identificata da un codice univoco	
<i>Infermiere</i>	Raccoglie i dati clinici del paziente e aggiorna la cartella clinica del paziente		L'infermiere è identificato da un username e una password	
<i>Dati anagrafici</i>	Generalità del paziente, come nome, cognome, codice fiscale, data e luogo di nascita, recapito telefonico, indirizzo, tipo e numero di documento	Testuale e numerico	Il codice fiscale è univoco per ogni paziente	
<i>Richiesta di un esame</i>	Invio elettronico delle generalità del paziente con conseguente iscrizione del paziente alla lista d'attesa per l'esame		Può essere effettuata solo dal medico	
<i>Esame</i>	Prestazione medica specifica per un paziente		Può essere richiesto solo da un medico	
<i>Paziente</i>	Individuo ricoverato in ospedale, a cui sono associate una o più cartelle cliniche		È univocamente identificato dal suo codice fiscale	
<i>Reparto</i>	Zona dell'ospedale adibita alla cura di malattie specifiche			

Medico	Individuo responsabile della cura dei pazienti all'interno di un reparto		Il medico è identificato da un username e una password	
Gestione della cartella clinica	Creazione della cartella clinica di un paziente a seguito di un ricovero, aggiornamento e visualizzazione		La creazione e l'aggiornamento della cartella clinica possono essere eseguite solamente da un medico	
Gestione del paziente	Gestione della cartella clinica associata al paziente ricoverato e richiesta degli esami			
Stato della cartella clinica	Indicazione binaria sull'attivazione o la disattivazione di una cartella clinica	Binario		
Codice fiscale	Identificativo univoco per pazienti			Codice numerico

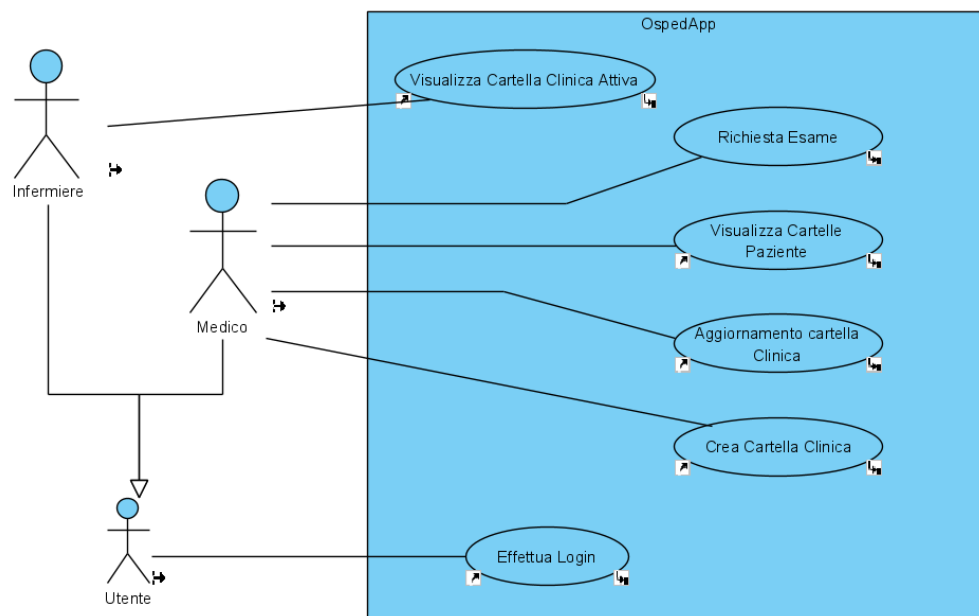
Capitolo 4 : Analisi dei requisiti

Identificazione degli attori

Durante lo sviluppo del sistema sono stati individuati i seguenti attori primari:

- UTENTE
- MEDICO
- INFERMIERE

Diagramma dei casi d'uso



Descrizione breve dei casi d'uso

Caso d'uso 1 : Effettua login

Il caso d'uso permette all'utente di inserire username e password per accedere al sistema. Il sistema poi verifica se l'utente esiste o meno e mostra la giusta interfaccia per l'utente registrato.

Caso d'uso 2 : Creazione di una cartella clinica

Il caso d'uso permette al medico di creare una nuova cartella clinica inserendo i dati anagrafici del paziente, per effettuare il ricovero. Il sistema memorizza i dati inseriti restituendo un messaggio di conferma.

Caso d'uso 3 : Visualizza cartella clinica attiva

Tramite il caso d'uso, si permette ad un infermiere di visualizzare la cartella clinica attiva di un paziente. Egli inserisce il codice fiscale per ricercare la cartella clinica del paziente e il sistema mostra a video le giuste informazioni.

Caso d'uso 4 : Aggiorna cartella clinica

Per l'aggiornamento, il medico effettua la ricerca di una cartella clinica attiva di un paziente tramite il codice fiscale e il sistema permette la modifica dei campi relativi ai dati anagrafici e di ricovero del paziente, nonché l'aggiunta di una nuova terapia.

Caso d'uso 5 : Visualizza cartelle paziente

Il medico è interessato a visualizzare tutte le cartelle cliniche relative ad un paziente, attive e non. Inserendo il codice fiscale del paziente, il sistema mostra tutte le cartelle cliniche del paziente e il medico sceglie quale visualizzare.

Caso d'uso 6 : Richiesta esame

Il medico effettua la richiesta di un esame per il paziente ricoverato. Dopo aver effettuato la ricerca di un paziente, il medico richiede un esame e il sistema registra la richiesta effettuata.

Tabella Attori – Obiettivi

Attore	Obiettivi
Utente	<ul style="list-style-type: none">• Effettua il login all'interno del sistema
Medico	<ul style="list-style-type: none">• Crea la cartella clinica per un paziente• Effettua l'aggiornamento della cartella attiva per un paziente• Visualizza tutte le cartelle cliniche che sono state associate ad un paziente

Infermiere

- Effettua la richiesta di un esame per un paziente
- Visualizza la cartella clinica attiva di un paziente

Descrizione di dettaglio dei casi d'uso

Caso d'uso 1 : Effettua login

Nome del caso d'uso	Effettua login
Ambito	Applicazione OspedApp
Livello	Obiettivo utente
Attore primario	Utente
Parti interessate	Utente che vuole accedere al sistema.
Precondizioni	L'utente deve essere preregistrato all'interno del sistema
Garanzia di successo	Il sistema mostra la corretta interfaccia
Principale scenario di successo	<ol style="list-style-type: none">1. L'utente inserisce ID e Password2. Il sistema verifica la validità dei dati3. L'utente accede all'interfaccia specifica
Estensioni	Se l'utente non è registrato 2a. Il sistema risponde con un messaggio d'errore
Requisiti particolari	Sistema di comunicazione tra terminale e base dati, che risponda in maniera affidabile il 90% delle volte
Tecnologie e variazioni dei dati	
Frequenza di occorrenza	Potenzialmente continuo
Varie	

Caso d'uso 2 : Creazione di una cartella clinica

Nome del caso d'uso	Creazione di una cartella clinica
Ambito	Applicazione OspedApp
Livello	Obiettivo utente
Attore primario	Medico

Parti interessate	Medico: deve creare la cartella clinica del paziente con terapie e diagnosi
Precondizioni	Il medico deve essere autenticato
Garanzia di successo	La cartella clinica è registrata
Principale scenario di successo	<ol style="list-style-type: none"> 1. Il medico raggiunge l'apposita postazione 2. Il medico avvia la creazione di una cartella clinica 3. Il sistema mostra a video l'interfaccia corretta 4. Il medico inserisce i dati del paziente 5. Il sistema mostra i dati del paziente 6. Il medico richiede la conferma di memorizzazione 7. Il sistema memorizza la nuova cartella 8. Il sistema disattiva le precedenti cartelle 9. Il sistema restituisce un messaggio di conferma
Estensioni	<p>4.a. Il paziente non è registrato nel database perché è la prima volta che effettua un ricovero</p> <ol style="list-style-type: none"> 1. Il sistema registra il paziente e la cartella clinica appena creata
Post-condizioni	Al paziente risulta associata un'unica cartella attiva
Requisiti particolari	Sistema di comunicazione tra terminale e base dati, che risponda in maniera affidabile il 99% delle volte
Tecnologie e variazioni dei dati	<ul style="list-style-type: none"> • La cartella clinica è identificata con un proprio codice • Il paziente è identificato attraverso il codice fiscale
Frequenza di occorrenza	Potenzialmente continuo
Varie	

Caso d'uso 3 : Visualizza cartella clinica attiva

Nome del caso d'uso	Visualizzazione cartella clinica attiva
Ambito	Applicazione OspedApp
Livello	Obiettivo utente
Attore primario	Infermiere

Parti interessate	<p>Infermiere: vuole somministrare la terapia corretta al paziente</p> <p>Paziente: gli viene somministrato il giusto trattamento</p> <p>Medico: vuole risolvere problemi/dubbi dell'infermiere</p>
Precondizioni	<p>Il paziente deve essere ricoverato</p> <p>L'infermiere deve essere autenticato e riconosciuto</p>
Garanzia di successo	La cartella clinica viene mostrata sul terminale
Principale scenario di successo	<ol style="list-style-type: none"> 1. L'infermiere si reca al terminale 2. Il sistema mostra l'interfaccia di visualizzazione 3. L'infermiere inserisce il codice fiscale del paziente 4. Il sistema effettua la ricerca del paziente 5. Il sistema ricerca la cartella attiva del paziente 6. Il sistema mostra a video la cartella clinica attiva
Estensioni	<p>2.a. Non esiste nessun paziente con quel codice fiscale</p> <ol style="list-style-type: none"> 1. il sistema segnala l'errore 2. Il sistema chiede di reinserire i dati <p>3.a. Non esiste nessuna cartella attiva per quel paziente</p> <ol style="list-style-type: none"> 1. Il sistema segnala che il paziente non è ricoverato al momento
Requisiti particolari	<ol style="list-style-type: none"> 1. Terminale di input dei dati 2. Sistema di comunicazione tra terminale e base dati, che risponda in maniera affidabile il 99% delle volte
Tecnologie e variazioni dei dati	<ul style="list-style-type: none"> • Il paziente è identificato attraverso il codice fiscale
Frequenza di occorrenza	Potenzialmente continuo
Varie	

Caso d'uso 4 : Aggiorna cartella clinica

Nome del caso d'uso	Aggiorna cartella clinica
Ambito	Applicazione OspedApp

Livello	Obiettivo utente
Attore primario	Medico
Parti interessate	Medico: vuole modificare la cartella clinica di un paziente
Precondizioni	Il medico deve essere autenticato
Garanzia di successo	La cartella clinica viene aggiornata
Principale scenario di successo	<ol style="list-style-type: none"> 1. Il medico si reca al terminale 2. Il sistema mostra la giusta interfaccia 3. Il medico inserisce il codice fiscale del paziente 4. Il sistema ricerca il paziente con il relativo codice 5. Il sistema ricerca la cartella clinica attiva per quel paziente 6. Il sistema mostra la cartella al medico 7. Il medico modifica i dati nella cartella 8. Il sistema registra le modifiche effettuate 9. Il sistema risponde con un messaggio di conferma
Estensioni	<p>4.a. Non esiste alcun paziente ricoverato con quel codice</p> <ol style="list-style-type: none"> 1. Il sistema mostra un messaggio di errore 2. Il sistema chiede di reinserire i dati <p>5.a. Non esiste una cartella attiva per quel paziente</p> <ol style="list-style-type: none"> 1. Il sistema mostra un messaggio di errore
Requisiti particolari	Sistema di comunicazione tra terminale e base dati, che risponda in maniera affidabile il 99% delle volte
Tecnologie e variazioni dei dati	<ul style="list-style-type: none"> • La cartella clinica è identificata con un proprio codice • Il paziente è identificato attraverso il codice fiscale
Frequenza di occorrenza	Ad ogni esame effettuato da un paziente
Varie	

Caso d'uso 5 : Visualizza cartelle paziente

Nome del caso d'uso	Visualizza cartelle paziente
Ambito	Applicazione OspedApp

Livello	Obiettivo utente
Attore primario	Medico
Parti interessate	Medico: vuole visualizzare informazioni di un paziente quali esami, terapie, anamnesi oppure precedenti diagnosi.
Precondizioni	Il paziente deve essere ricoverato e il medico deve essere autenticato
Garanzia di successo	La cartella clinica viene mostrata sul terminale
Principale scenario di successo	<ol style="list-style-type: none"> 1. Il medico si reca all'apposito terminale 2. Il sistema visualizza la corretta interfaccia 3. Il medico inserisce il codice fiscale del paziente 4. Il sistema effettua la ricerca del paziente 5. Il sistema ricerca la cartella attiva del paziente 6. Il medico visualizza la cartella clinica del paziente
Estensioni	<p>3.a Se il paziente non esiste</p> <ol style="list-style-type: none"> 1. il sistema fornisce un messaggio d'errore <p>5.a Se non esistono cartelle cliniche per quel paziente</p> <ol style="list-style-type: none"> 1. Il sistema dà un messaggio d'errore
Requisiti particolari	<ul style="list-style-type: none"> • Terminale di input dei dati • Sistema di comunicazione tra tablet e base dati, che risponda in maniera affidabile il 99% delle volte
Tecnologie e variazioni dei dati	<ul style="list-style-type: none"> • Il paziente è identificato attraverso il codice fiscale
Frequenza di occorrenza	Saltuario
Varie	

Caso d'uso 6 : Richiesta esame

Nome del caso d'uso	Richiesta esame
Ambito	Applicazione OspedApp
Livello	Obiettivo utente
Attore primario	Medico

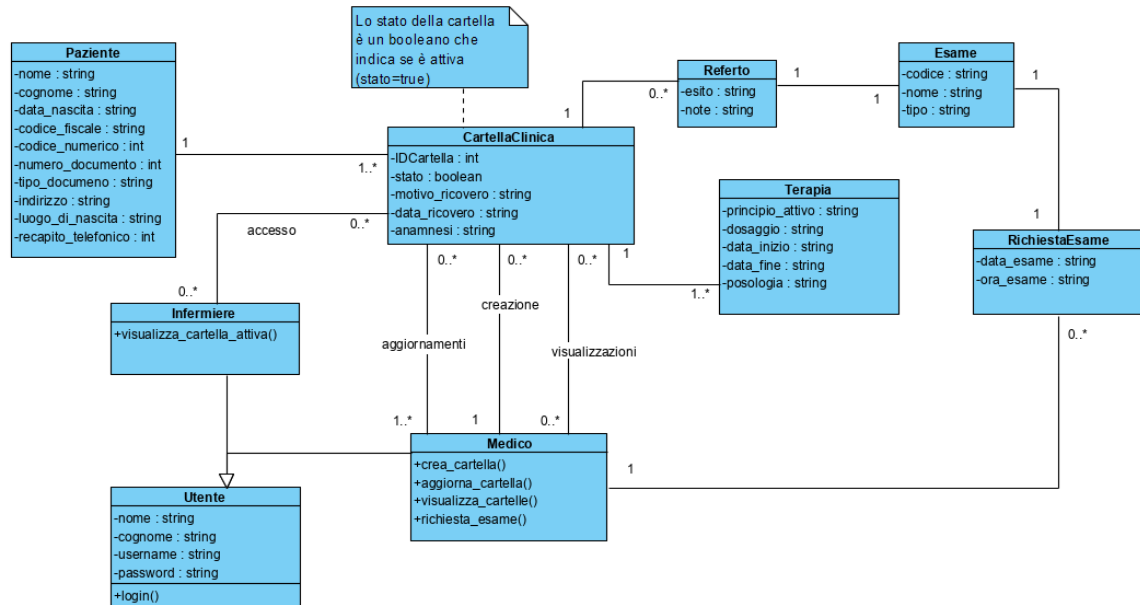
Parti interessate	<p>Medico: deve richiedere una prestazione medica per un paziente</p> <p>Paziente: effettua la visita medica</p>
Precondizioni	Il medico deve essere autenticato e il paziente deve risultare ricoverato
Garanzia di successo	L'esame viene registrato nella cartella clinica
Principale scenario di successo	<ol style="list-style-type: none"> 1. Il medico si reca al terminale 2. Il sistema mostra l'interfaccia 3. Il medico inserisce i dati anagrafici del paziente 4. il sistema effettua la ricerca del paziente 5. il medico richiede un esame per il paziente 6. Il sistema verifica se la richiesta può essere registrata 7. il sistema registra la richiesta 8. il sistema dà un messaggio di conferma
Estensioni	<p>4.a. Non esiste alcun paziente ricoverato con quei dati</p> <ol style="list-style-type: none"> 1. Il sistema mostra un messaggio di errore 2. Il sistema chiede di reinserire i dati <p>5.a. La richiesta non può essere registrata</p> <ol style="list-style-type: none"> 1. Il sistema mostra un messaggio di errore
Requisiti particolari	<ul style="list-style-type: none"> • Terminale di input dei dati • Sistema di comunicazione tra terminale e base dati, che risponda in maniera affidabile il 99% delle volte
Tecnologie e variazioni dei dati	<ul style="list-style-type: none"> • Il paziente è identificato attraverso il codice fiscale
Frequenza di occorrenza	Potenzialmente continuo
Varie	

Modello di dominio del sistema

Al fine di mettere a fuoco i concetti fondamentali del sistema e definire un vocabolario specifico, si è pensato di introdurre il modello di dominio del sistema stesso.

Un importante merito di questo tipo di modello è quello di fornire una base comune di concetti su cui ragionare e una terminologia condivisa rigorosa e specifica. Tale modello dà quindi una descrizione statica della struttura del sistema.

Di seguito il modello di dominio del sistema OspedApp.

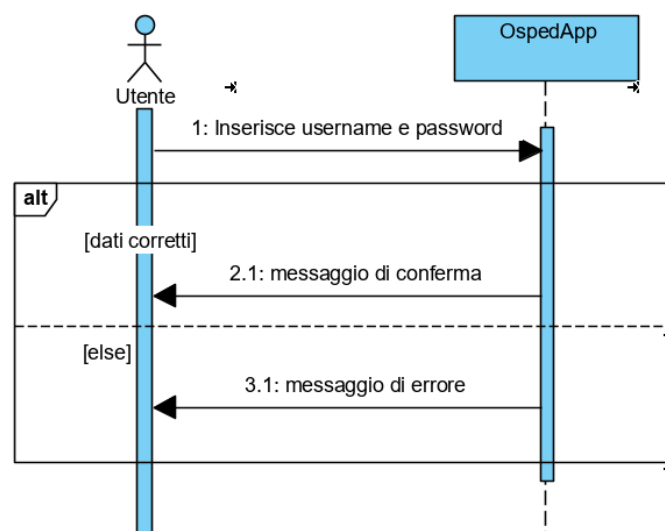


Sequence diagram di analisi

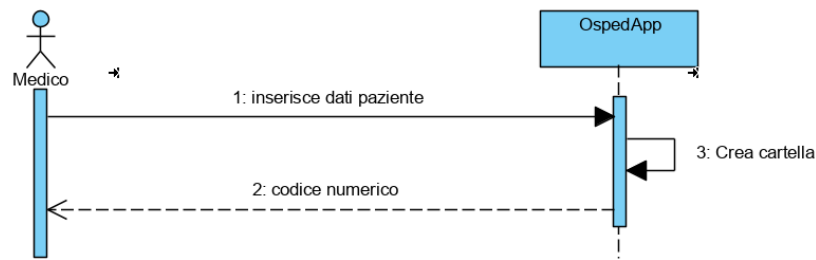
Un diagramma utile per visualizzare le interazioni tra un attore e il sistema è il diagramma di sequenza, che mette in luce una determinata sequenza di azioni in cui tutte le scelte sono state già effettuate.

Di seguito i sequence diagram di analisi delle funzionalità identificate per il sistema.

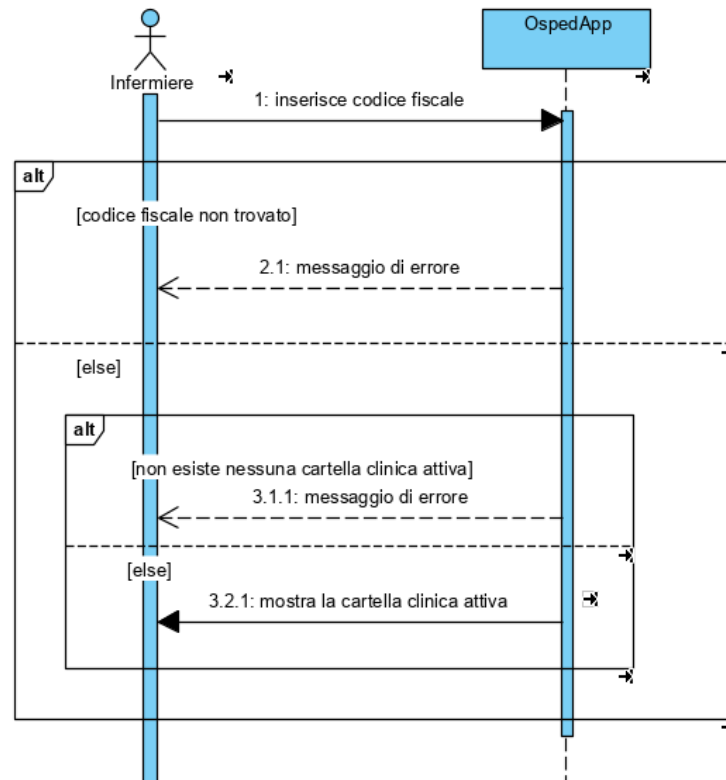
Effettua login



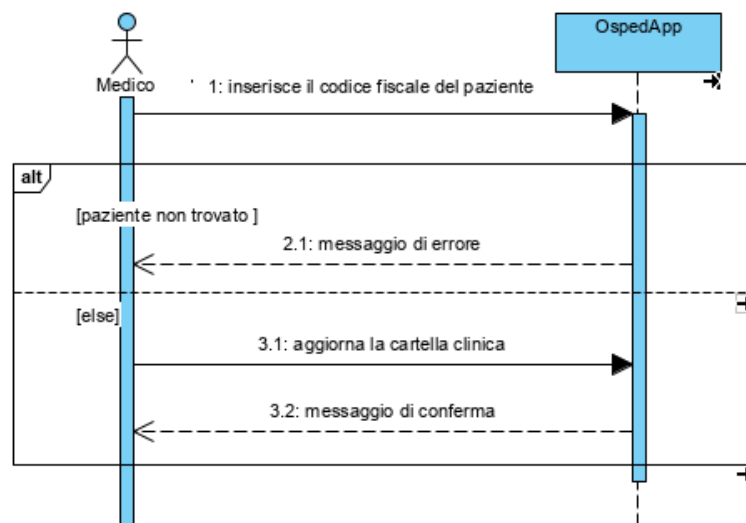
Creazione di una cartella clinica



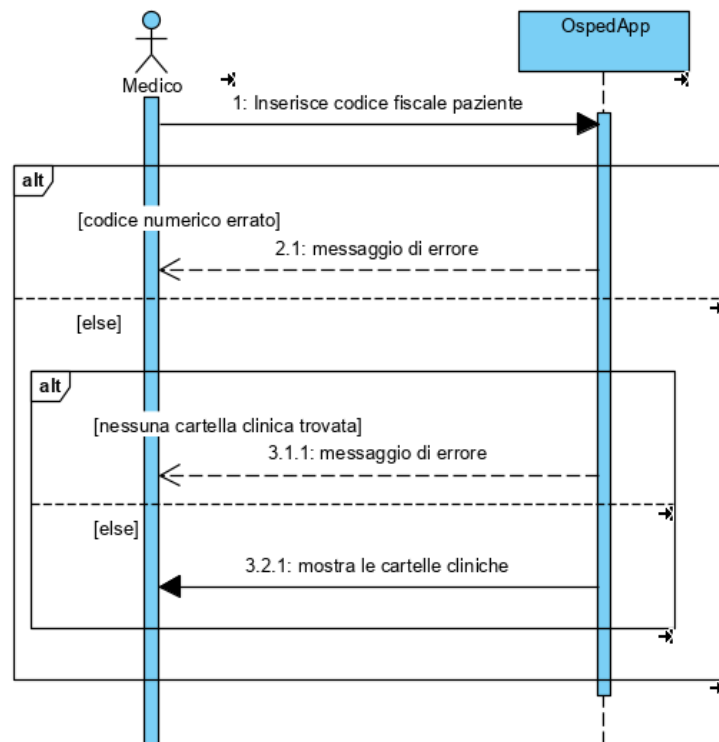
Visualizza cartella clinica attiva



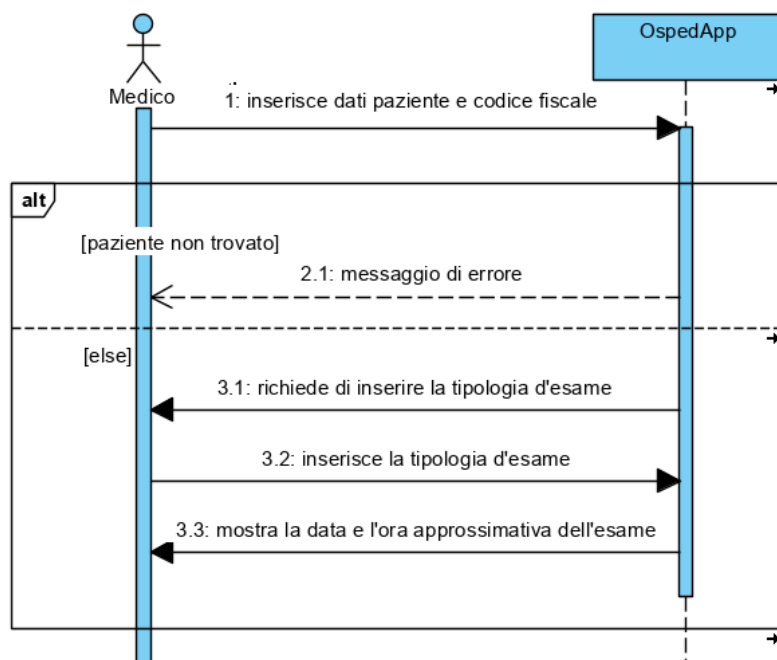
Aggiorna cartella clinica



Visualizza cartelle paziente



Richiesta esame

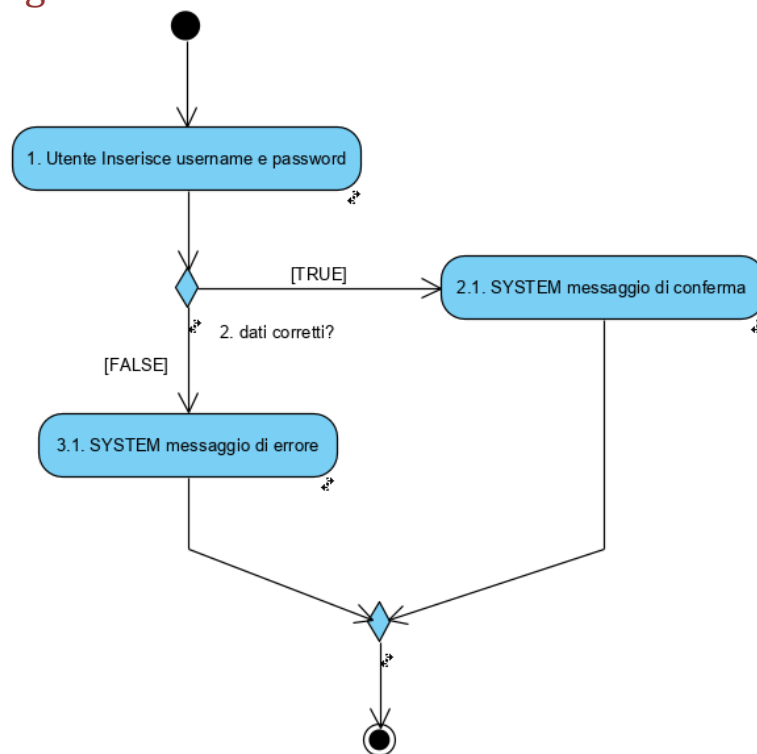


Activity diagram di analisi

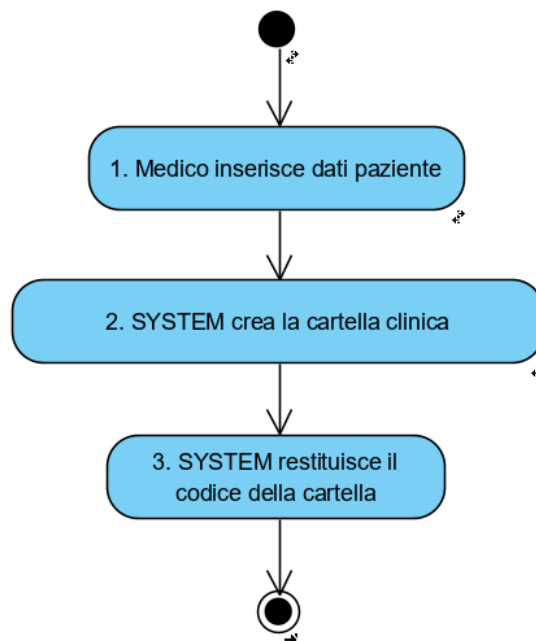
Altro diagramma utile che permette di descrivere un processo attraverso dei grafi in cui i nodi rappresentano le attività e gli archi l'ordine con cui vengono eseguite, è il diagramma di attività.

Di seguito i diagrammi di attività delle funzionalità del sistema OspedApp.

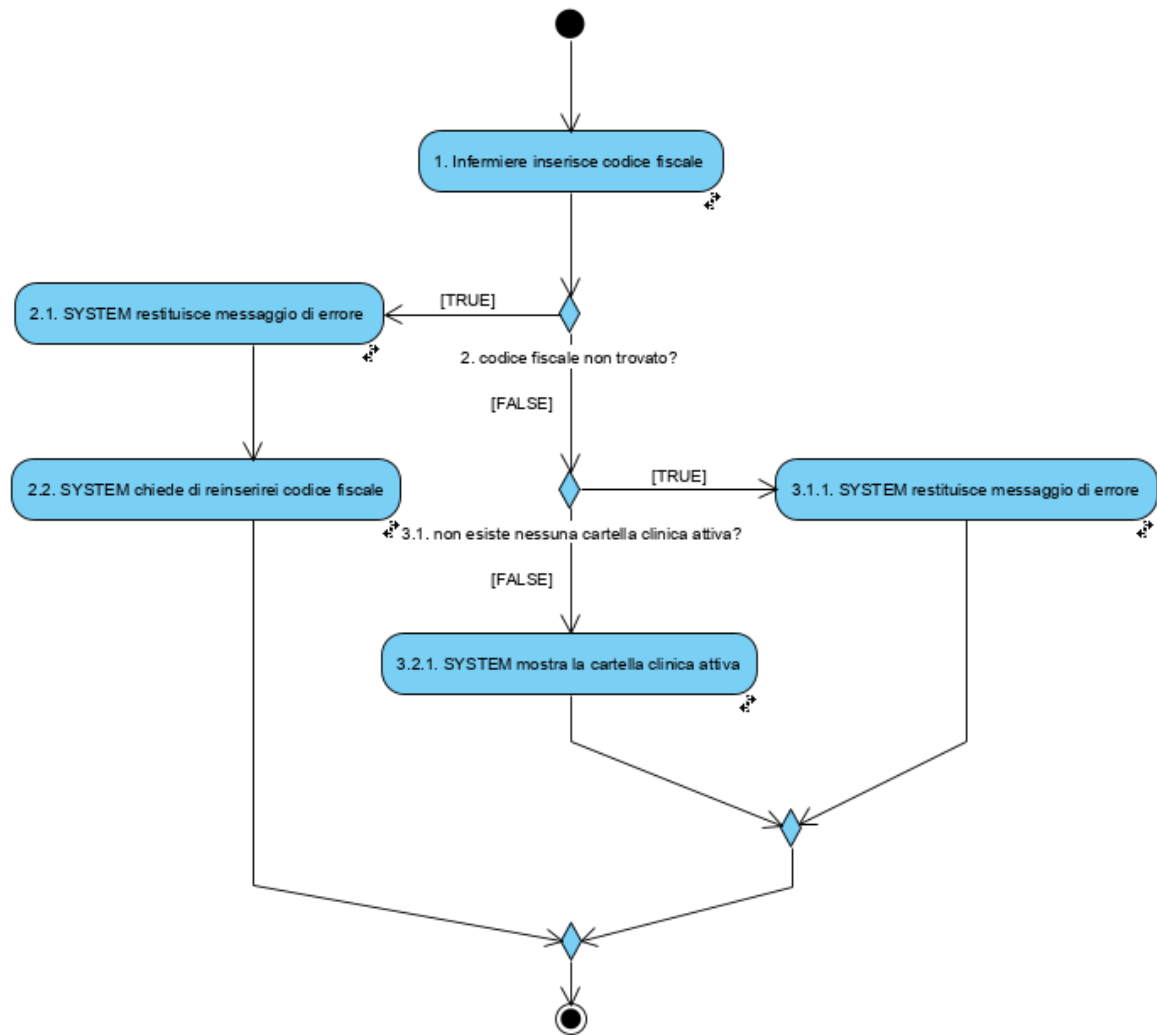
Effettua login



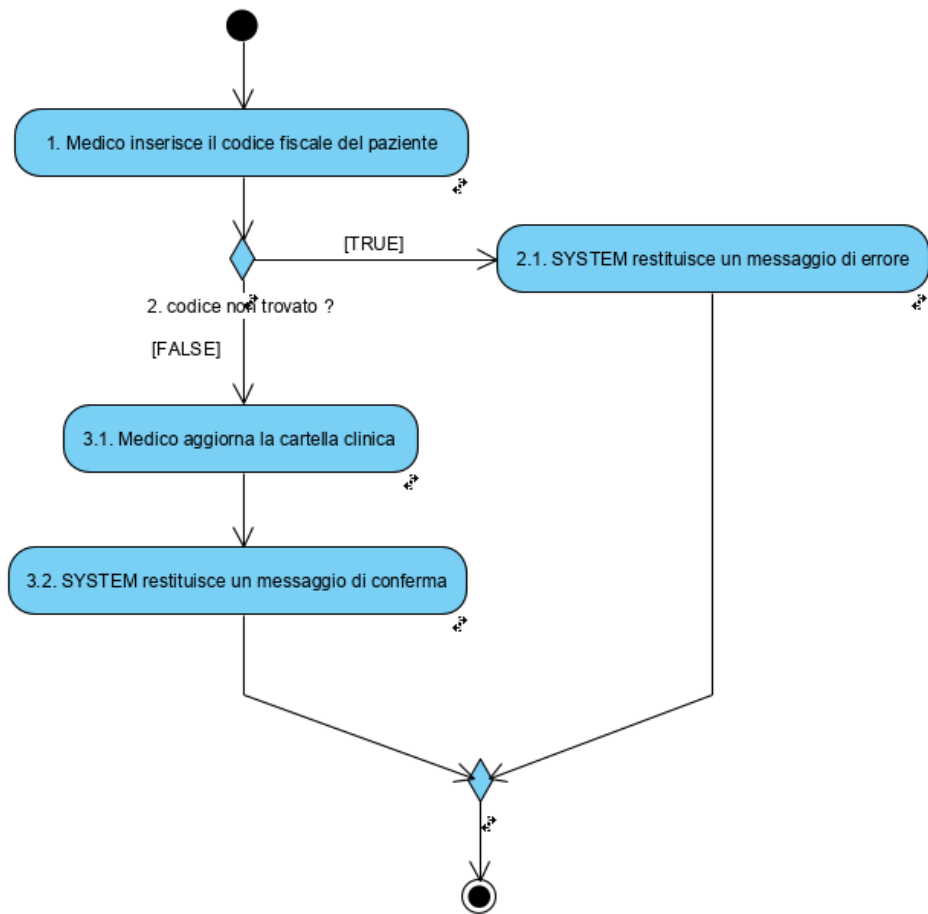
Creazione di una cartella clinica



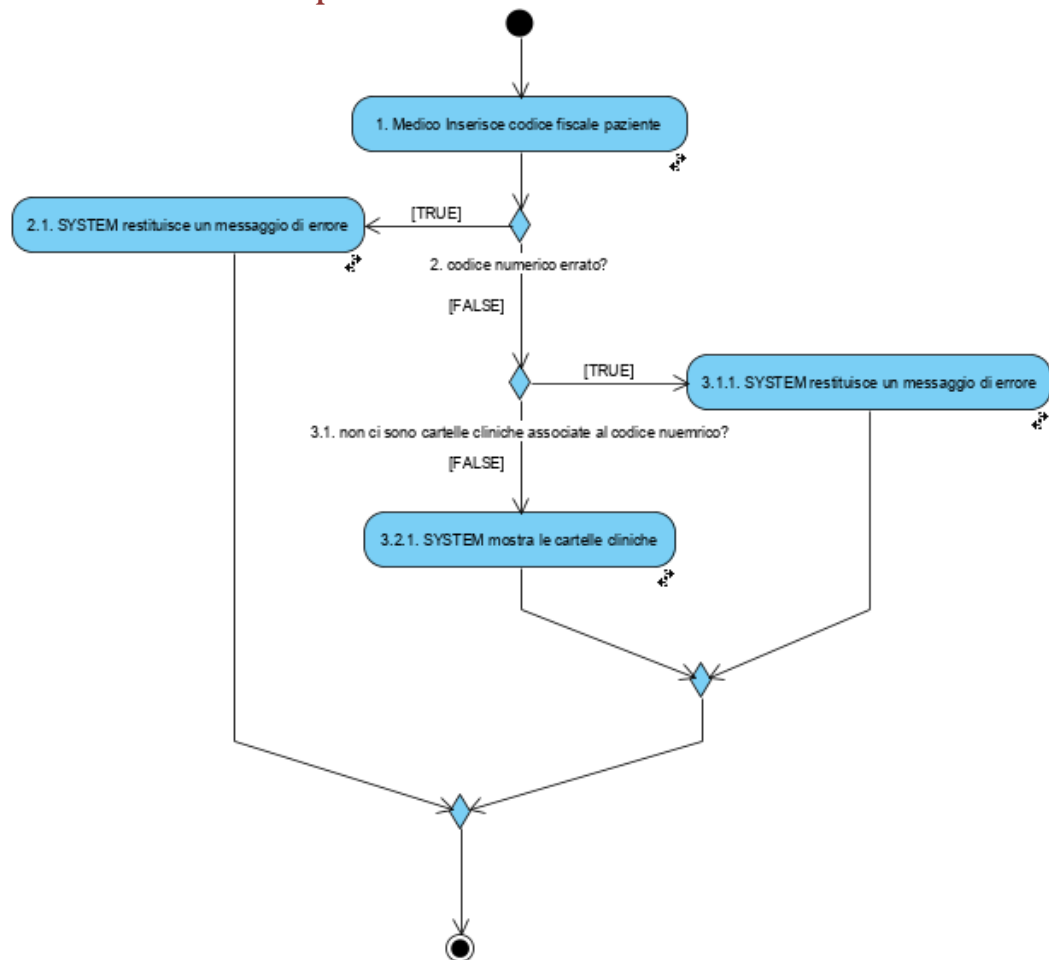
Visualizza cartella clinica attiva



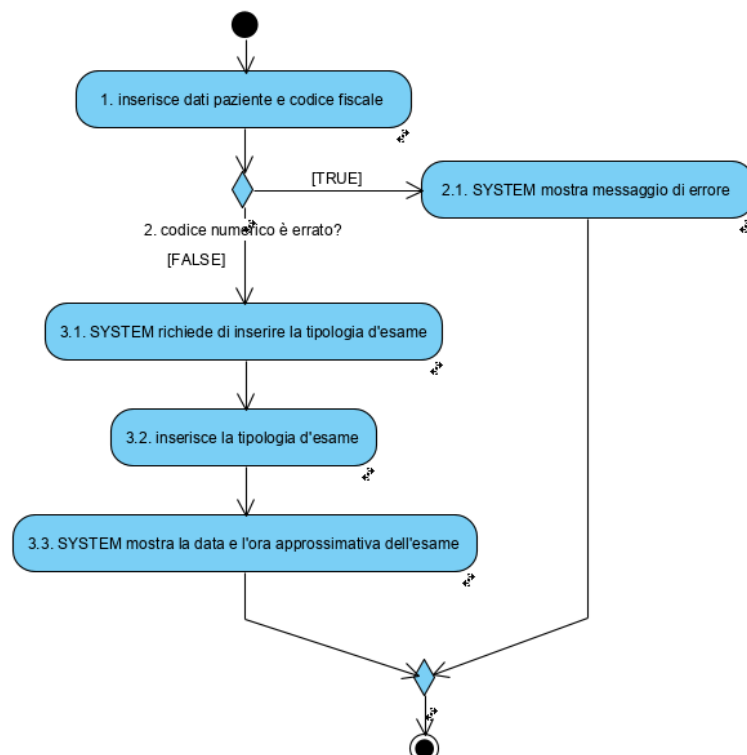
Aggiorna cartella clinica



Visualizza cartelle paziente



Richiesta esame



Capitolo 5 : Architettura Software

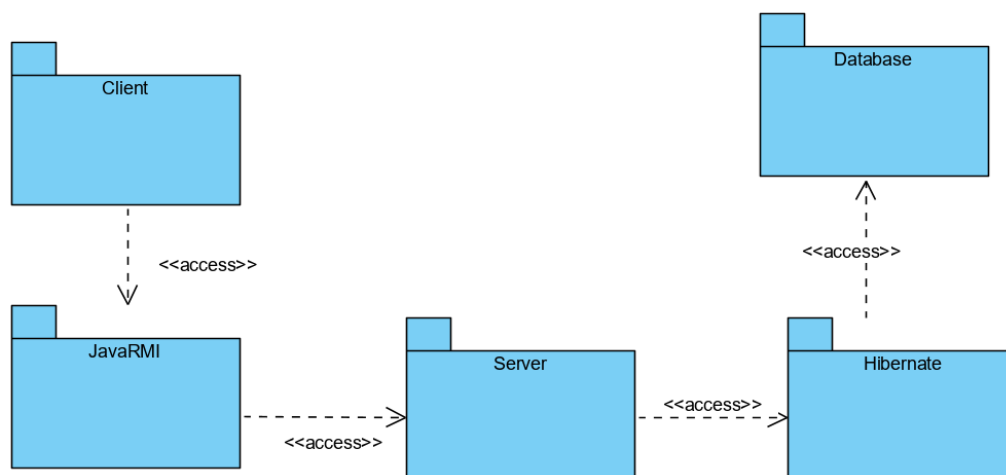
All'interno di un sistema software è importante avere una vista di come sarà l'architettura di quest'ultimo. In particolare, una architettura software è definita dai suoi componenti, dalle relazioni reciproche tra i componenti e con l'ambiente e dai principi che ne governano la progettazione e l'evoluzione.

Infatti, descrivere l'architettura software di un sistema significa elencarne le sottoparti costituenti ed illustrarne i rapporti inter-funzionali.

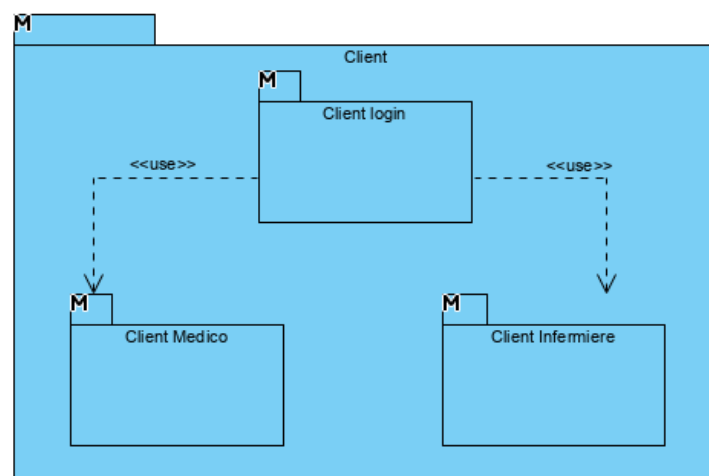
Per la descrizione dell'architettura utilizzata ci si è avvalso di uno stile architetturale di tipo Client-Server; in particolare tale stile permette di ripartire l'elaborazione tra client e server ed evitare così scarse prestazioni, e inoltre permette l'interoperabilità tra sistemi, dando la possibilità a diversi componenti di lavorare assieme.

Di seguito le viste di alto livello dell'architettura del sistema OspedApp.

Vista architetturale di alto livello



Vista architetturale di alto livello lato client



Vista architetturale di alto livello lato server

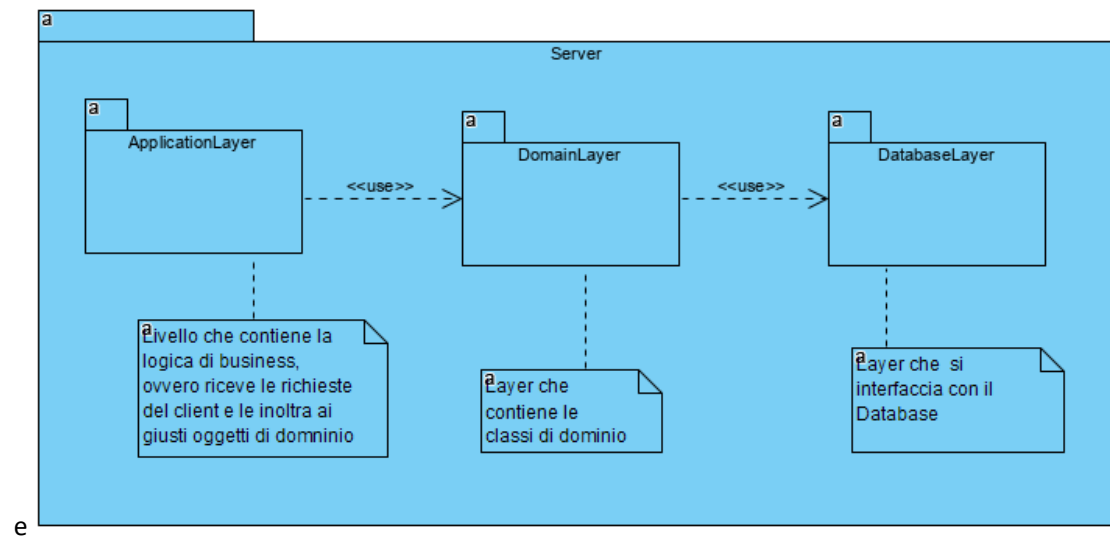
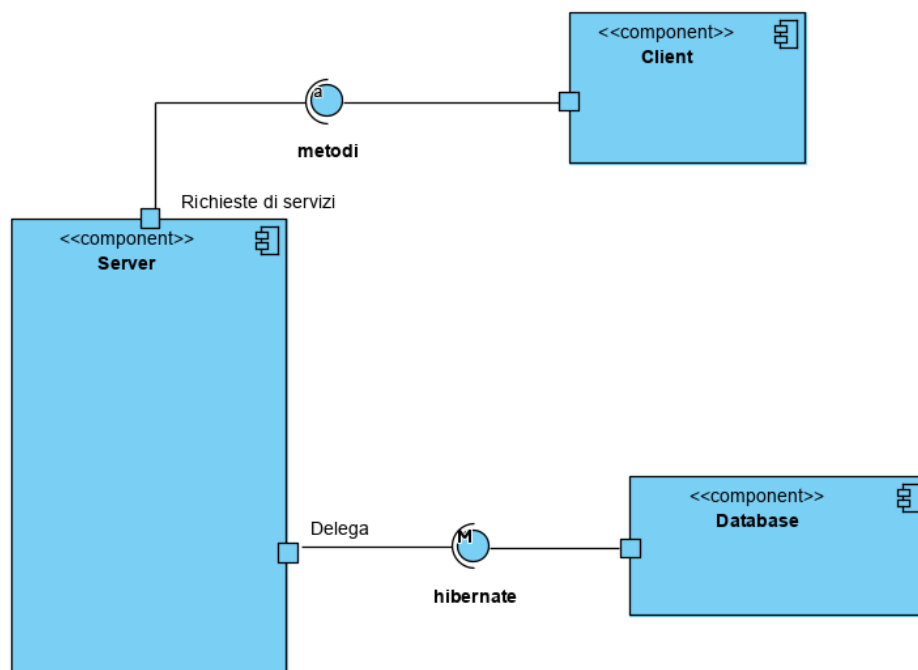


Diagramma dei componenti

Per avere una vista della struttura interna del software modellato attraverso le interazioni tra i vari componenti del sistema stesso, è utile stendere un diagramma dei componenti.

Di seguito la soluzione implementata per il software OspedApp.



Descrizione dell'architettura nel dettaglio

Scendendo più nel dettaglio dell'architettura, come pattern architetturale si è scelto il 2-tiered: sono presenti 2 tier separati, il client e il server, e la comunicazione tra i due tier avviene da remoto mediante il connettore **Java RMI** (*Remote Server Invocation*). Il server a sua volta comunica con un database MySQL mediante il framework *Hibernate*.

Dal punto di vista logico, il Client rappresenta il Presentation Layer, ovvero il livello che si occupa di interfacciarsi con l'utente, mentre il Server è diviso in tre livelli:

- **Application Layer**, che contiene la logica di business, ovvero si occupa di prendere le richieste del Client e inoltrarle ai giusti oggetti del Domain Layer;
- **Domain Layer**, che contiene gli oggetti di dominio che si occupano di realizzare le responsabilità richieste;
- **Database Layer**, che è il livello che si occupa di comunicare con il Database MySQL per accedere ai dati.

I livelli sono di tipo *Closed*, ovvero un livello può comunicare solamente con il livello subito inferiore e non può bypassarlo: questa scelta può portare ad ulteriori rallentamenti, ma si è scelto di operare in questo modo in quanto il numero di livelli è relativamente basso (4), e mediante questa scelta è possibile ottenere il massimo disaccoppiamento.

Il client è di tipo *thin*, ovvero non contiene la logica applicativa, e ciò permette di avere un client più leggero.

Per problemi di visualizzazione dei diagrammi all'interno del documento, è convenuto separare le implementazioni del lato client e del lato server in due diagrammi differenti.

Di seguito mostrati i suddetti diagrammi; per una più corretta visualizzazione si rimanda al file in Visual Paradigm contenente i diagrammi inseriti nel documento.

Diagramma architetturale di dettaglio lato client

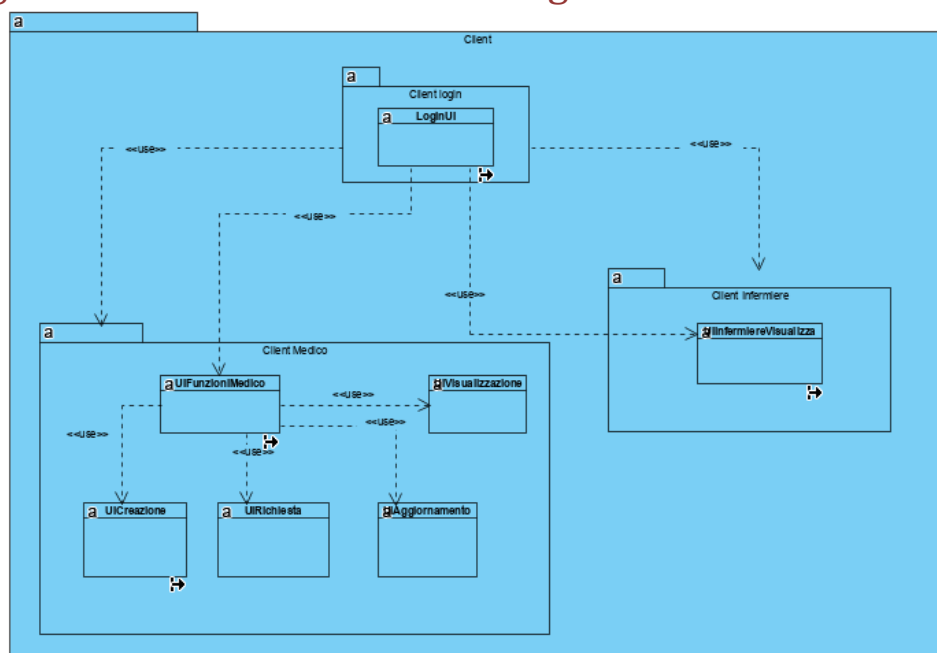
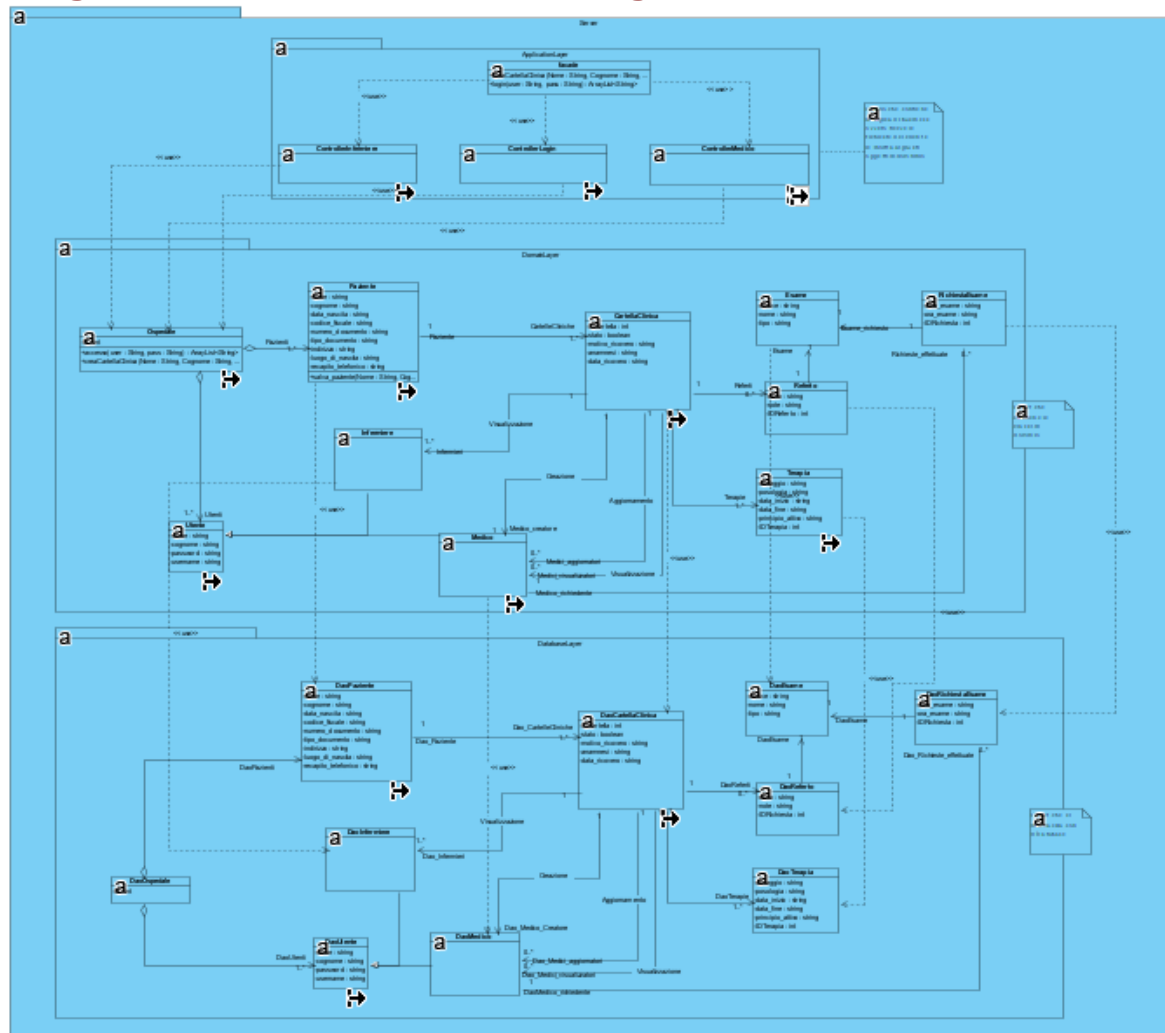


Diagramma architetturale di dettaglio lato server



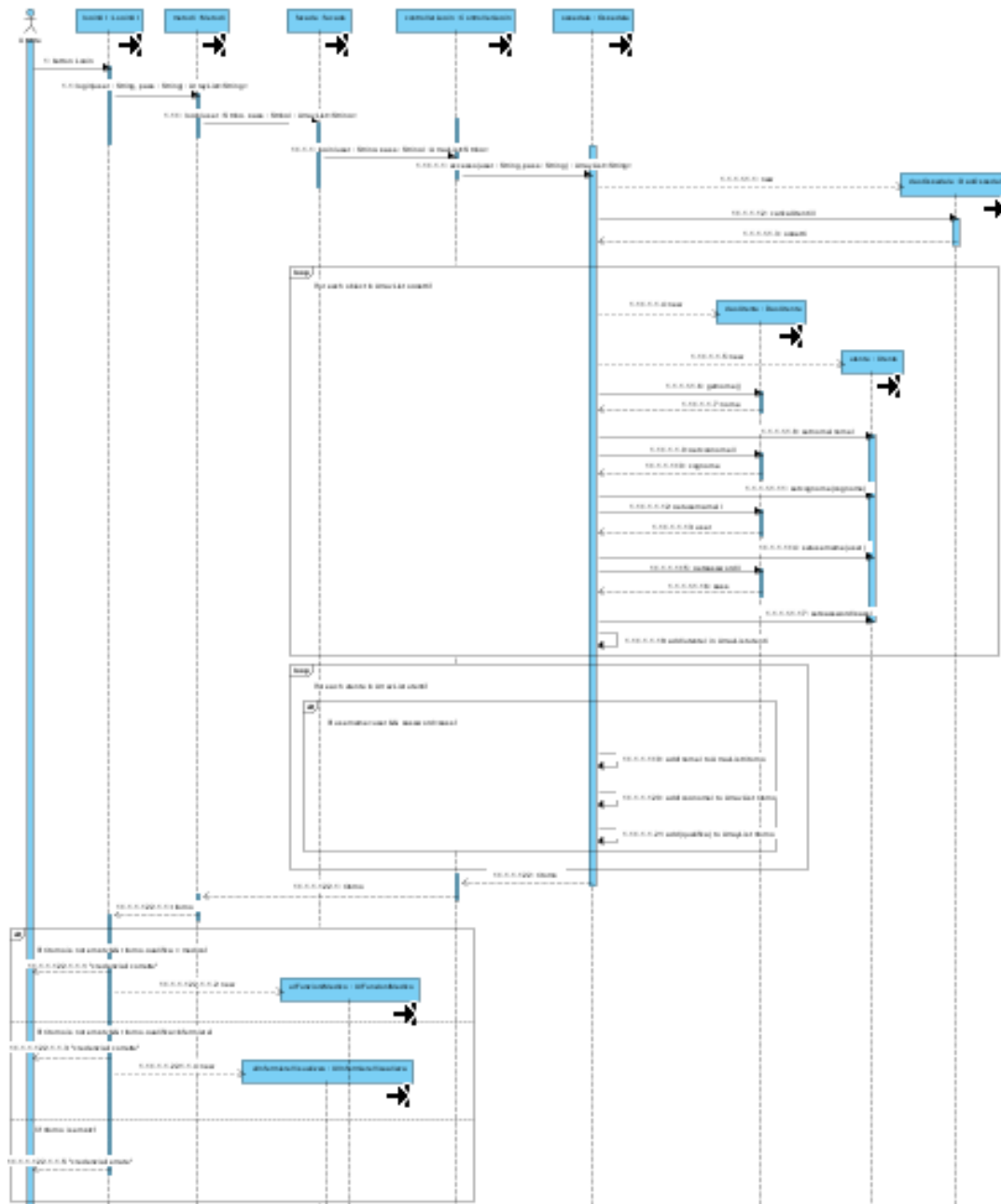
Diagrammi di sequenza di dettaglio

Un diagramma di sequenza di dettaglio può essere usato per definire gli input e gli output del sistema da realizzare. Esso rappresenta un particolare scenario di caso d'uso, dove per scenario si intende una determinata sequenza di azioni in cui tutte le scelte sono già state effettuate.

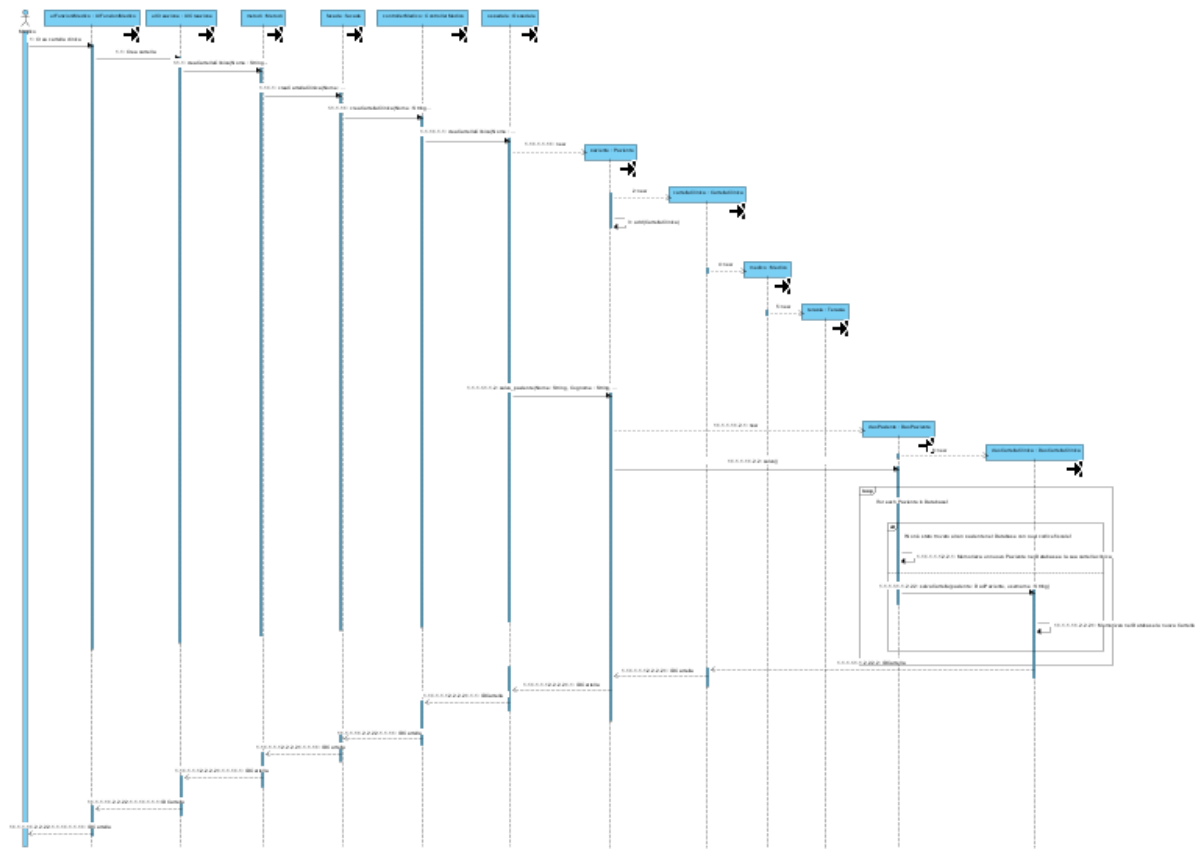
Il diagramma di sequenza di dettaglio descrive le relazioni che intercorrono in termini di messaggi tra Attori, Oggetti di business, Oggetti o Entità del sistema che si sta rappresentando.

In UP questi diagrammi vanno associati solo ai casi d'uso e agli scenari che sono oggetto della specifica iterazione. Devono essere definiti in fase di elaborazione per determinare gli eventi e le operazioni del sistema e per scrivere i contratti delle operazioni di sistema.

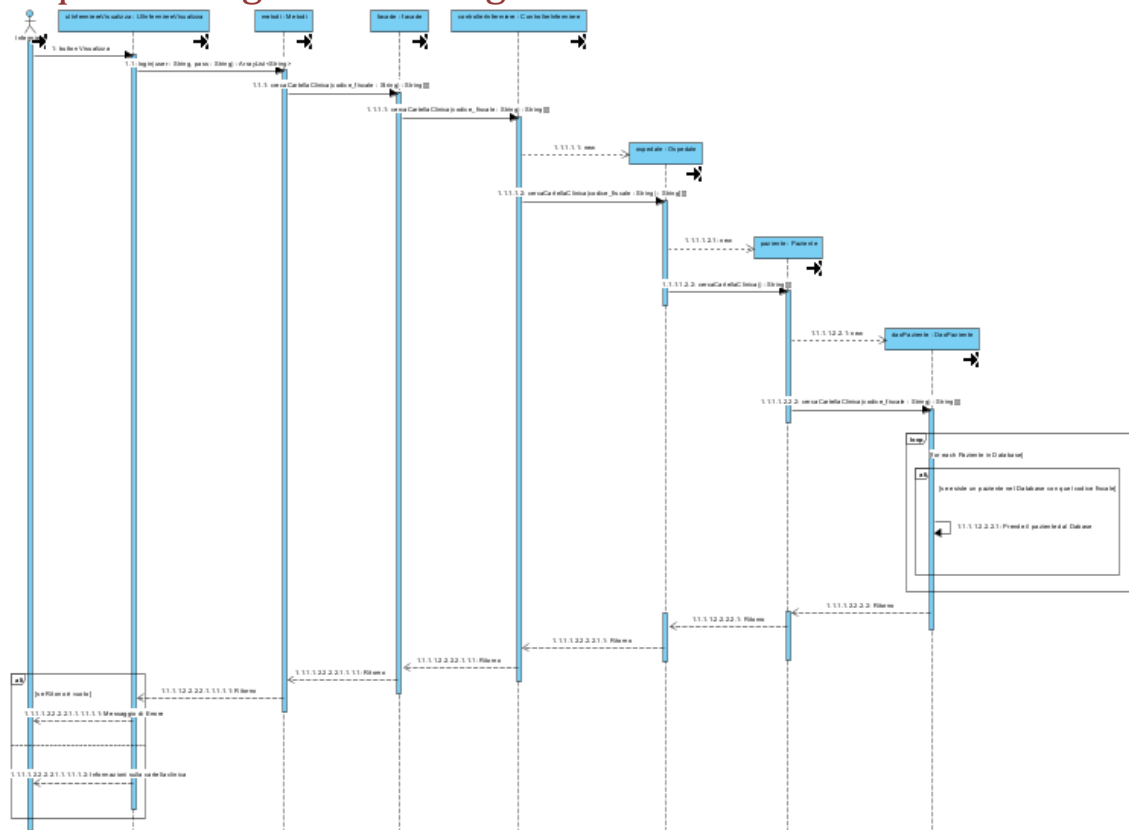
Sequence diagram di dettaglio – Effettua Login



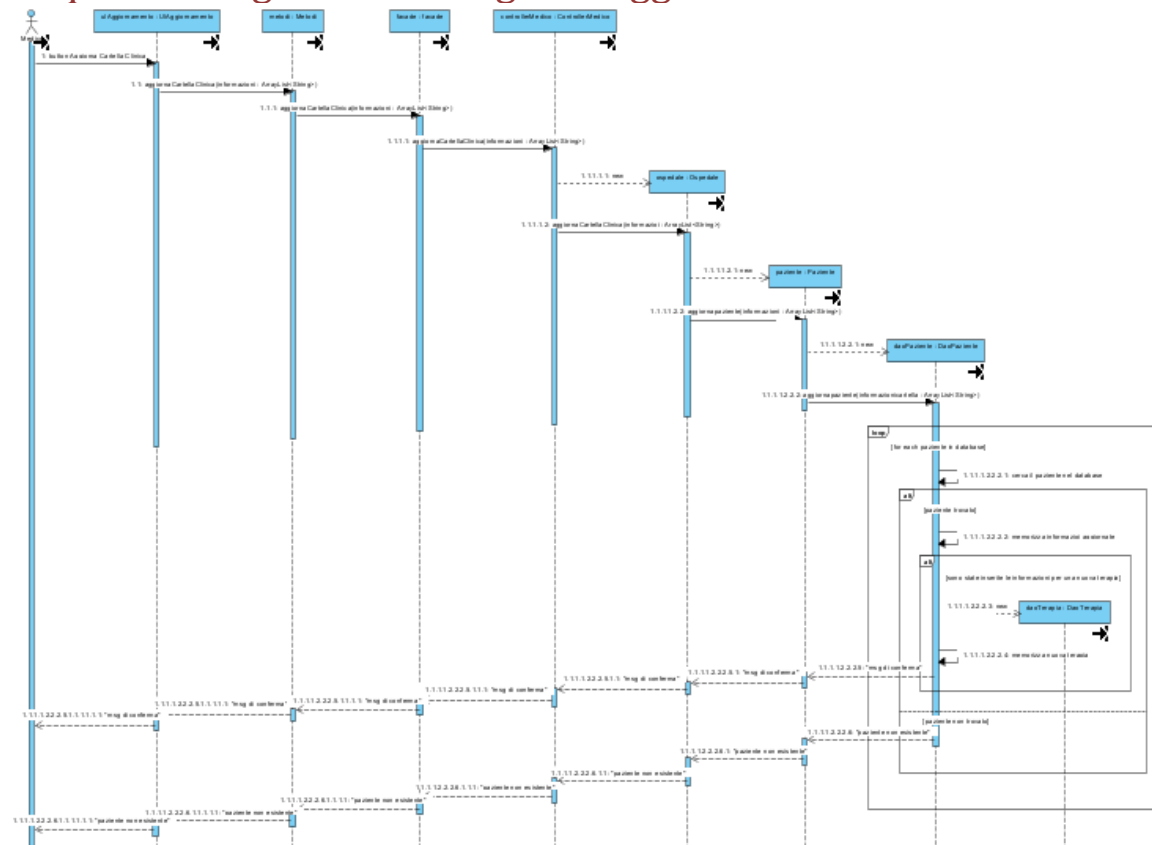
Sequence diagram di dettaglio – Creazione cartella clinica



Sequence diagram di dettaglio – Visualizza cartella clinica



Sequence diagram di dettaglio – Aggiorna cartella clinica

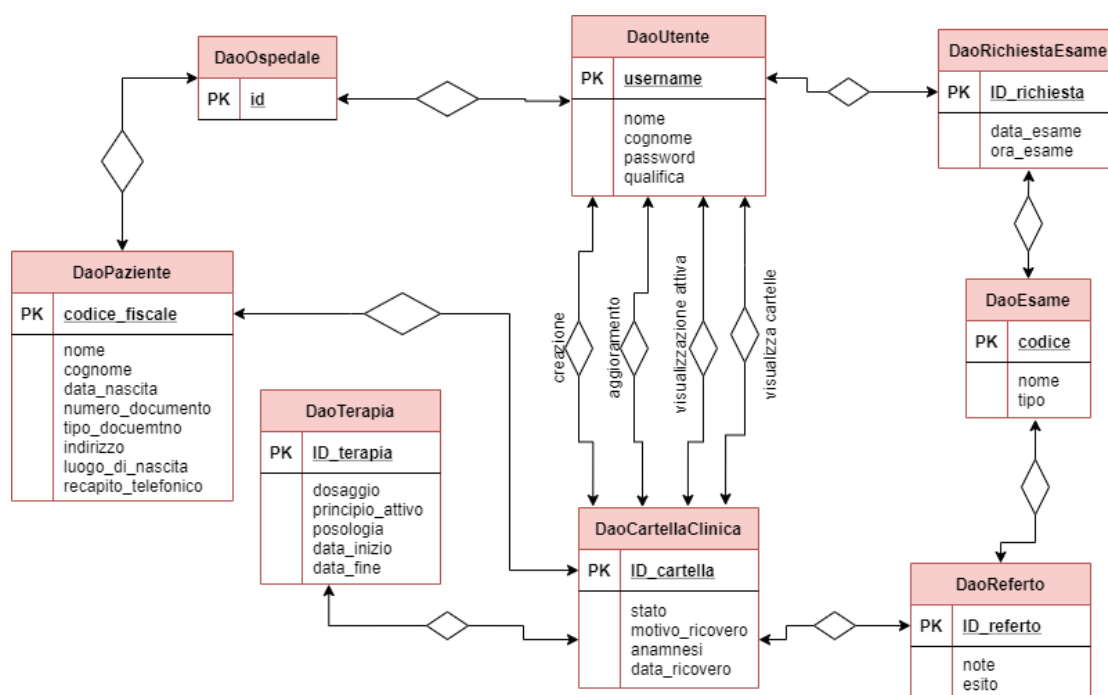


Capitolo 6 : Persistenza dei dati

Per effettuare i salvataggi persistenti dei dati è stato implementato un database di tipo relazionale utilizzando MySQL. In particolare, per definire in maniera semplice le entità e le relazioni della base di dati utilizzando una grafica intuitiva senza dover ricorrere all'esecuzione di specifiche istruzioni SQL, si è optato per un software gratuito quale *MySQL Workbench* per la definizione e la costruzione della base dati stessa.

In dettaglio, si è deciso in primis di effettuare un diagramma entità-relazioni per la progettazione della base dati persistente e, successivamente, si è proceduto alla costruzione del database tramite il tool indicato in precedenza.

Di seguito il diagramma E/R rappresentante le entità in gioco.



Dal diagramma si evincono quindi le seguenti scelte effettuate a livello di persistenza dei dati:

- Introduzione di una classe fittizia che rappresenta l'intero sistema, per rendere il mapping tra oggetti e tabelle relazionali più completo;
- Definizione di associazioni multiple tra le tabelle *DaoUtente* e *DaoCartellaClinica*, per tenere traccia di diversi legami tra le due entità, come l'utente che effettua l'aggiornamento di una cartella, l'utente che ne effettua la creazione o ancora l'utente che effettua la visualizzazione della cartella clinica;
- Si è pensato di evitare l'inserimento di tabelle multiple riguardanti i medici e gli infermieri all'interno dell'ospedale, in primis per facilitare l'operazione di login andando a ricercare una tupla all'interno di un'unica tabella, e inoltre per evitare la duplicazione e l'eccessiva frammentazione dei dati all'interno del database stesso; se si fosse giunti alla conclusione di separare le informazioni relative a medici e infermieri, infatti, avremmo ottenuto due tabelle formate dagli stessi campi, ciascuna con un numero di tuple

irrisorio considerato il numero medio di personale medico e sanitario all'interno di un ospedale;

- Si è pensato di predisporre la base dati per un eventuale aggiunta di un esame all'interno dell'ospedale con l'aggiunta della tabella *DaoEsame*, che contiene tutti i possibili esami per cui, all'interno dell'ospedale, è possibile effettuare una richiesta;

Definizione delle relazioni

- **DAOOSPEDALE – DAOPAZIENTE** : ad ogni paziente è associato un ospedale, e ogni ospedale può includere più pazienti
- **DAOOSPEDALE – DAOUTENTE** : ad ogni utente è associato un ospedale, mentre ad ogni ospedale sono associati più utenti
- **DAOPAZIENTE – DAOCARTELLECLINICHE** : ad ogni paziente corrispondono più cartelle cliniche, ma ogni cartella clinica è associata ad un solo paziente
- **DAOCARTELLACLINICA – DAOTERAPIA** : ad ogni cartella sono associate più terapie, mentre una specifica terapia è associata ad un'unica cartella
- **DAOCARTELLACLINICA – DAOREFERTO** : ad ogni cartella clinica sono relativi più referti, mentre ogni referto è relativo ad un'unica cartella clinica
- **DAOUTENTE – DAOCARTELLACLINICA**
 - **CREAZIONE** : Per ogni cartella clinica c'è un solo utente che effettua la creazione, ma un utente può effettuare la creazione di più cartelle cliniche;
 - **AGGIORNAMENTO** : Per ogni cartella clinica esiste un unico medico che ha per ultimo aggiornato quella cartella, mentre ogni utente potrebbe aver aggiornato più cartelle cliniche;
 - **VISUALIZZAZIONE ATTIVA** : Per ogni cartella clinica un singolo infermiere effettua l'ultima visualizzazione di una cartella attiva, e un utente potrebbe visualizzare più di una cartella clinica attiva;
 - **VISUALIZZA CARTELLE** : Per ogni cartella clinica un singolo medico effettua la visualizzazione delle cartelle cliniche di un paziente, mentre un medico può visualizzare le cartelle cliniche di più pazienti;

Andando a dettagliare il diagramma aggiungendo le relative chiavi esterne e i tipi degli attributi presenti nelle varie entità, è stato possibile generare un diagramma raffinato, utilizzando il tool Visual Paradigm. Il risultato è specificato di seguito.



Traduzione in tabelle

Nella fase finale di progettazione logica della base di dati si è tradotto ogni entità in una tabella, avente per nome quello dell'entità stessa, e per colonne gli attributi delle varie entità. Di seguito le tabelle identificate tramite la fase di traduzione.

- ❖ **DaoOspedale** (id)
- ❖ **DaoUtente** (username, nome, cognome, password, qualifica)
- ❖ **DaoPaziente** (codice_fiscale, nome, cognome, data_nascita, numero_documento, tipo_documento, indirizzo, luogo_di_nascita, recapito_telefonico, DaoOspedaleId : DaoOspedale)
- ❖ **DaoCartellaClinica** (ID_Cartella, stato, motivo_ricovero, anamnesi, data_ricovero, DaoPazienteCodice_fiscale : DaoPaziente, Utente_creazione : DaoUtente, Medico_aggiornamento : DaoUtente, Medico_visualizzazione : DaoUtente, Infermiere_visualizzazione : DaoUtente)
- ❖ **DaoTerapia** (ID_terapia, dosaggio, principio_attivo, data_inizio, data_fine, posologia, DaoCartellaClinicaID_Cartella : DaoCartellaClinica)
- ❖ **DaoRichiestaEsame** (IDRichiesta, data_esame, ora_esame, DaoUtenteusername : DaoUtente, DaoEsamecodice : DaoEsame)

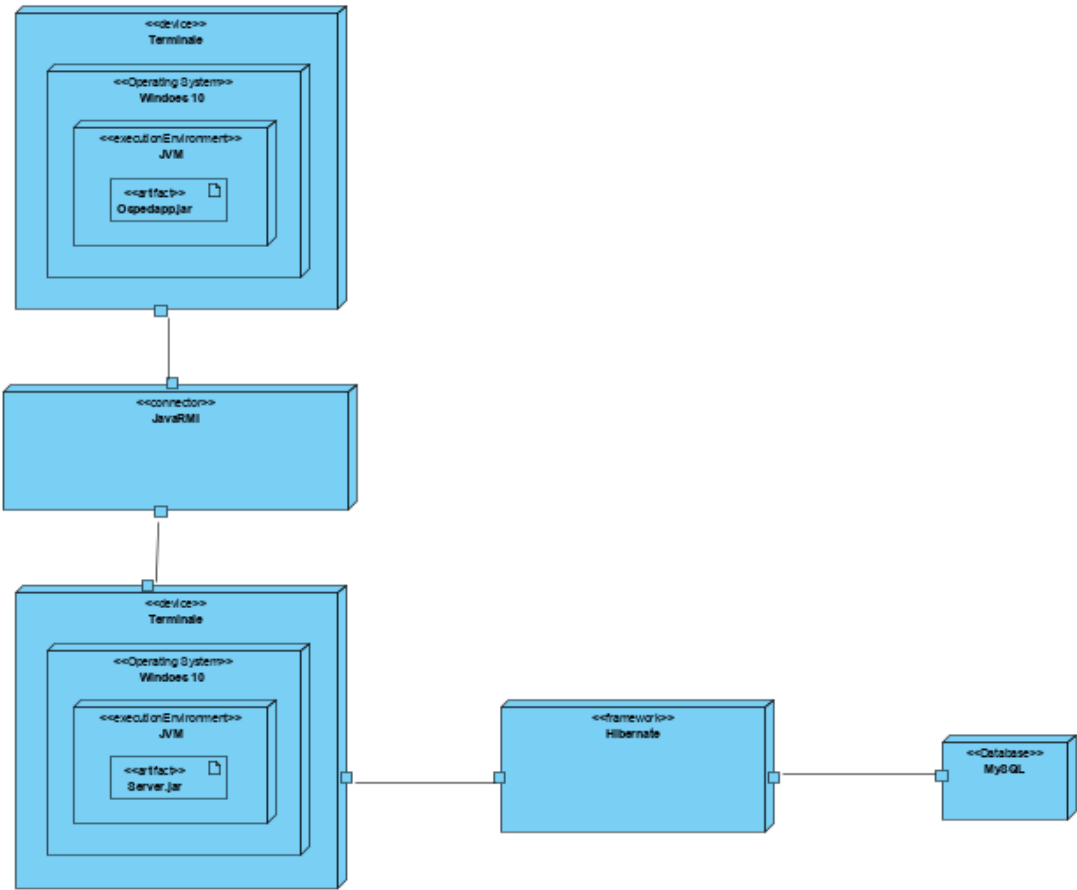
❖ **DaoEsame** (codice, nome, tipo)

❖ **DaoReferto** (IDReferto, note, esito, DaoEsamecodice : DaoEsame,
DaoCartellaClinicaIDCartella : DaoCartellaClinica)

Capitolo 7 : Implementazione

Diagramma di deploy

Per rendere più chiaro il rilascio e le modalità con cui esso avviene, è sempre utile fornire un diagramma di deploy del sistema. Di seguito il diagramma sviluppato per OspedApp.



Documenti di implementazione

Di seguito viene riportato uno schema in cui si tiene traccia del lavoro svolto quotidianamente a livello implementativo.

ID	Implementazione	Inizio	Fine	Durata	Status
1	Iterazione 1	Ago 23, 2020	Set 1, 2020	10 d	Completato
2	Esplorazione veloce dei requisiti	Ago 23, 2020	Ago 25, 2020	3 d	
3	Prototipazione rapida UI	Ago 26, 2020	Ago 27, 2020	1 d	

4	Implementazione caso d'uso "Login"	Ago 27, 2020	Ago 28, 2020	1 d	
5	Implementazione caso d'uso "Crea Cartella Clinica"	Ago 28, 2020	Set 1, 2020	5 d	
6	Iterazione 2	Set 2, 2020	Set 9, 2020	8 d	Completato
7	Aggiornamento rapido UI	Set 2, 2020	Set 3, 2020	1 d	
8	Implementazione caso d'uso "Visualizza Cartella Clinica Attiva"	Set 4, 2020	Set 9, 2020	6 d	
9	Iterazione 3	Set 10, 2020	Set 15, 2020	6 d	Completato
10	Aggiornamento rapido UI	Set 10, 2020	Set 11, 2020	1 d	
11	Implementazione caso d'uso "Aggiorna Cartella Clinica"	Set 11, 2020	Set 15, 2020	5 d	
12	Iterazione 4	Set 16, 2020	Set 20, 2020	5d	Completato
13	Analisi rapida dei casi d'uso "Richiedi Esame" e "Visualizza Cartelle Paziente"	Set 16, 2020	Set 17, 2020	2 d	
14	Progettazione di dettaglio della UI	Set 17, 2020	Set 20, 2020	4 d	

Servizi utilizzati

Il sistema è stato progettato secondo il paradigma OO (Object-Oriented) utilizzando il linguaggio di programmazione Java. Essendo l'architettura di tipo Client/Server, per implementare la comunicazione tra i Client e il Server è stato utilizzato JavaRMI (*Remote Method Invocation*). Questa tecnologia include una API (*Application Programming Interface*) il cui scopo esplicito è quello di rendere trasparenti al programmatore tutti i dettagli della comunicazione su rete. Infatti, essa consente di invocare un metodo di un oggetto "remoto", cioè appartenente ad un diverso processo e quindi situato su un'altra macchina, quasi come se tale oggetto fosse locale. L'utilizzo di questo meccanismo di invocazione remota di metodi comporta un grande vantaggio nei sistemi OO poiché consente di modellare le interazioni fra processi distribuiti usando lo stesso strumento concettuale che si utilizza per rappresentare le interazioni fra i diversi oggetti di un'applicazione, ossia la chiamata di metodi.

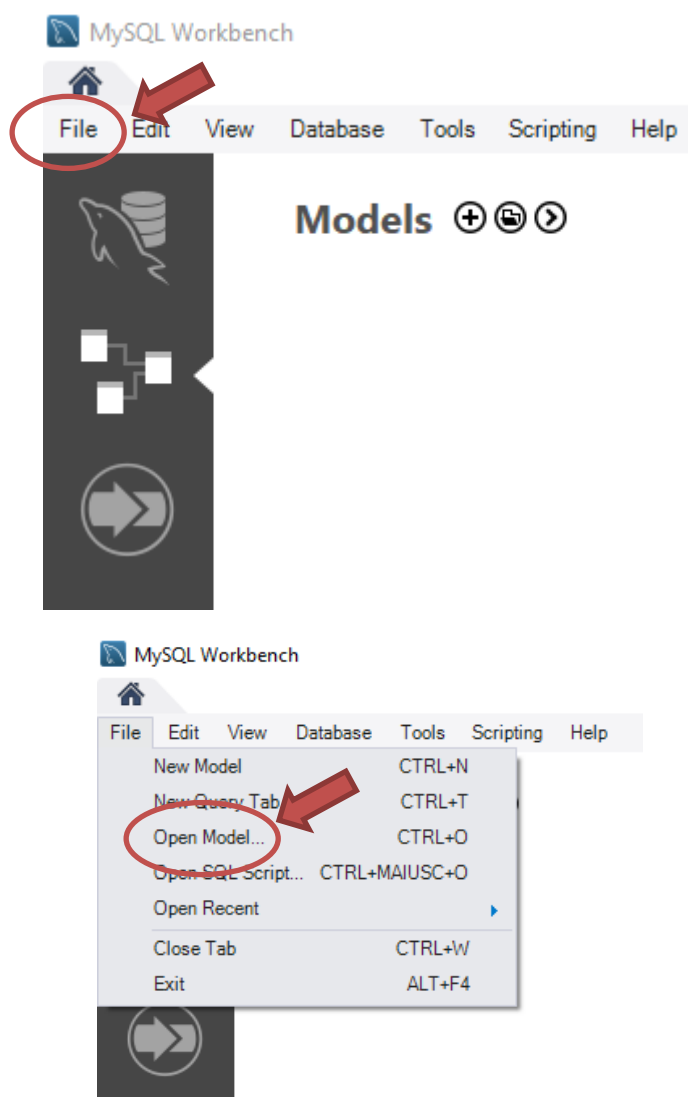
Per quanto riguarda invece la connessione del Server al Database MySQL, è stato utilizzato il connettore MySQL JConnector supportato da Hibernate. Hibernate è una piattaforma middleware per lo sviluppo di applicazioni Java che fornisce un servizio di ORM (*Object Relational Mapping*) per la gestione della persistenza dei dati sul database attraverso la rappresentazione e il mantenimento su DB relazionale di un sistema di oggetti Java. L'obiettivo principale di Hibernate è quello di esonerare lo sviluppatore dall'intero lavoro relativo alla persistenza dei dati.

Capitolo 8 : Manuale di utilizzo del software

Configurazione del database

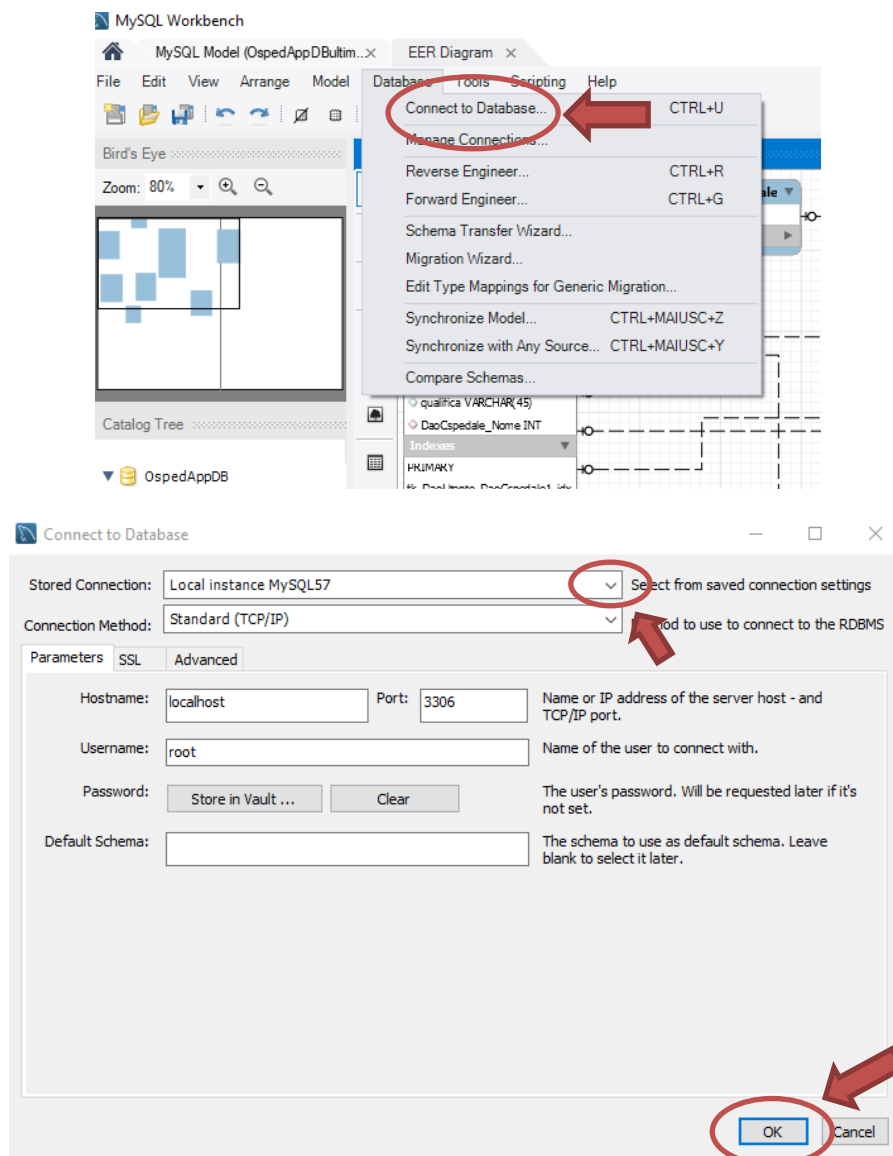
Una delle operazioni preliminari per poter osservare il corretto funzionamento del sistema OspedApp, è quella di configurare il database ed effettuare le operazioni di start di quest'ultimo.

Per fare ciò è necessario effettuare l'import del workspace per la creazione delle tabelle. Per effettuare la creazione delle tabelle è necessario recarsi nella scheda **File** all'interno di MySQL Workbench, cliccare su **Open Model...** e selezionare il file dato a disposizione con estensione **mwb**. Di seguito le operazioni viste nel dettaglio.



Successivamente l'operazione che si effettua è collegare lo schema relazionale appena importato all'istanza di MySQL creata in automatico quando viene installato il tool. Per fare ciò bisogna recarsi nella scheda in alto con la dicitura **Database** e cliccare su **Connect to database**.

Dopo aver cliccato sull'apposito comando, si aprirà una finestra di controllo per la configurazione della connessione con il database. Qui è necessario controllare se alla voce **Stored Connection** sia presente l'istanza di MySQL corretta. Successivamente cliccare su **Ok**. Di seguito le operazioni nel dettaglio.



Una volta effettuato la connessione tra database e modello, è necessario sincronizzare il modello al database a cui è stata appena effettuata la connessione. Per fare ciò è necessario recarsi alla voce **Database** all'interno della scheda del modello di MySQL, e cliccare su **Synchronize model....**

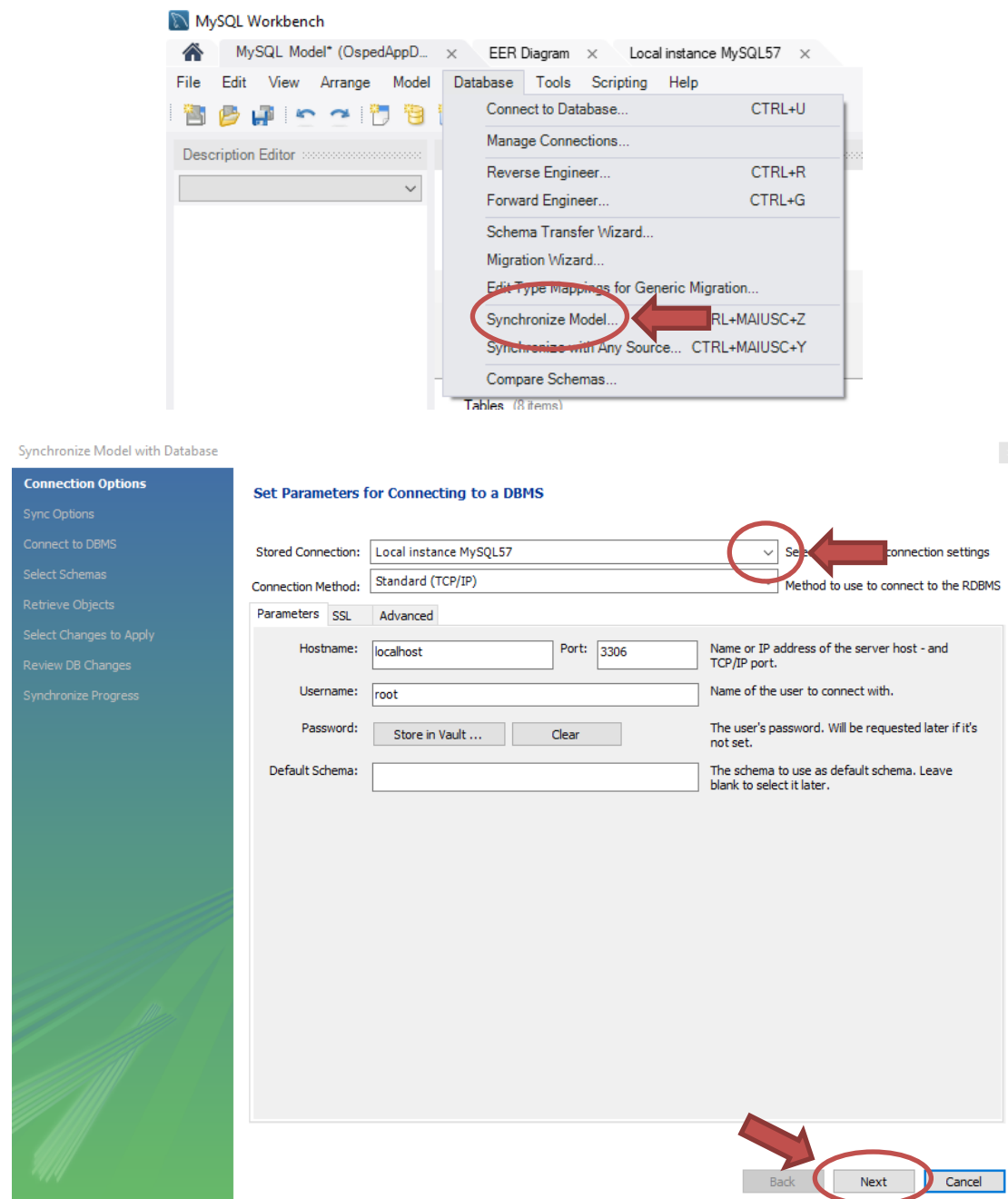
Questa operazione permette al tool di effettuare operazioni in linguaggio SQL al fine di andare a creare effettivamente lo schema all'interno del database. Una volta effettuato il click, comparirà una scheda, nella quale è necessario, come effettuato in precedenza, controllare che nel campo **Stored Connection** sia stata inserita la connessione corretta. Fatto ciò cliccare su **Next**.

Successivamente comparirà il secondo menù, per settare le opzioni di connessione; si consiglia di non modificare nulla e cliccare ancora una volta su **Next**.

Attendere l'esecuzione della fase di configurazione e cliccare su **Next**. Selezionare lo schema corretto e proseguire con la fase successiva. Attendere ancora una volta la fase di reverse engineering del database, e cliccare su **Next**.

Successivamente comparirà una schermata che mostra i cambiamenti che verranno effettuati all'interno dello schema del database. Se le scelte rispecchiano i cambiamenti che vogliamo ottenere clicchiamo su **Next**, altrimenti è possibile effettuare una configurazione dei cambiamenti tramite il click sull'operazione sospetta e cliccando su **Table Mapping**....

Infine, il tool mostra un riassunto delle operazioni in SQL che verranno eseguite all'atto del click sul bottone **Execute**. Cliccando sul tasto in questione vengono eseguiti i comandi specificati e la sincronizzazione del modello con il database. Cliccando su **Close** è possibile chiudere la finestra di dialogo. Di seguito alcune istantanee delle operazioni appena descritte.



Connection Options

Sync Options

Connect to DBMS

Select Schemas

Retrieve Objects

Select Changes to Apply

Review DB Changes

Synchronize Progress

Select the Schemas to be Synchronized



Select the Schemata to be Synchronized:

Model Schema	RDBMS Schema
<input checked="" type="checkbox"/> ospedAppDB	ospedappdb

Override target schema to be synchronized with: ospedappdb

Override Target

Next

Cancel

Connection Options

Sync Options

Connect to DBMS

Select Schemas

Retrieve Objects

Select Changes to Apply

Review DB Changes

Synchronize Progress

Connect to DBMS and Fetch Information

The following tasks will now be executed. Please monitor the execution.
Press Show Logs to see the execution logs.

- ☒ Connect to DBMS
- ☒ Retrieve Schema List from Database
- ☒ Check Common Server Configuration Issues

Execution Completed Successfully
Fetch finished.

Show Logs

Back

Next

Cancel

Connection Options

Sync Options

Connect to DBMS

Select Schemas

Retrieve Objects

Select Changes to Apply

Review DB Changes

Synchronize Progress

Model and Database Differences

Double click arrows in the list to choose whether to ignore changes, update the model with database changes or vice-versa. You can also apply an action to multiple selected rows.

Model	Update	Source
▼ ospedapdb (OspedAppDB)		ospedapdb
DaoPaziente	→	daopaziente
DaoCartellaClinica	→	N/A
DaoTerapia	→	N/A
DaoReferto	→	N/A
DaoEsame	→	N/A
DaoRichiestaEsame	→	N/A
DaoUtente	→	N/A
DaoOspedale	←	daoospedale

Update Model Ignore Update Source

Table Mapping... Column Mapping...

Back **Next** Cancel

Connection Options

Sync Options

Connect to DBMS

Select Schemas

Retrieve Objects

Select Changes to Apply

Review DB Changes

Synchronize Progress

Model and Database Differences

Double click arrows in the list to choose whether to ignore changes, update the model with database changes or vice-versa. You can also apply an action to multiple selected rows.

Model	Update	Source
▼ ospedapdb (OspedAppDB)		ospedapdb
DaoPaziente	→	daopaziente
DaoCartellaClinica	→	N/A
DaoTerapia	→	N/A
DaoReferto	→	N/A
DaoEsame	→	N/A
DaoRichiestaEsame	→	N/A
DaoUtente	→	N/A
DaoOspedale	←	daoospedale

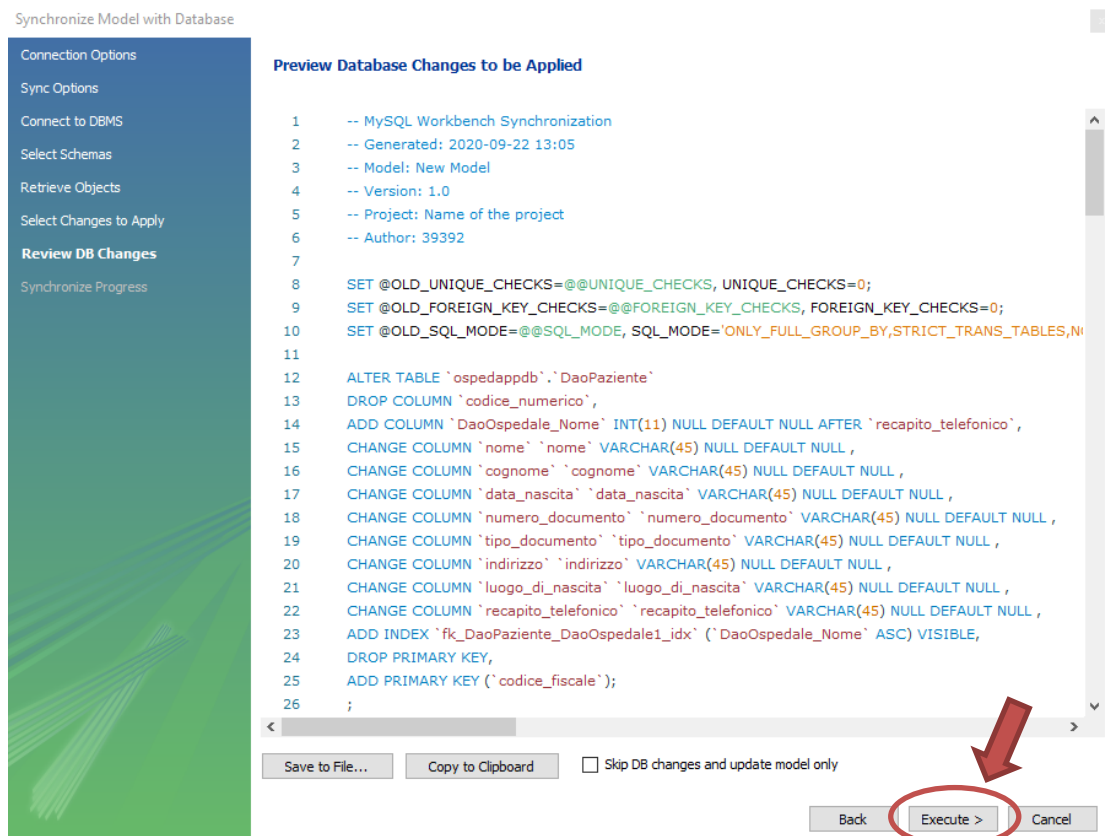
```
CREATE TABLE IF NOT EXISTS `ospedapdb`.`DaoTerapia` (
  `IDTerapia` INT(11) NOT NULL AUTO_INCREMENT,
  `principio_attivo` VARCHAR(255) NULL DEFAULT NULL,
  `data_fine` VARCHAR(45) NULL DEFAULT NULL,
  `data_inizio` VARCHAR(45) NULL DEFAULT NULL,
  `posologia` VARCHAR(45) NULL DEFAULT NULL,
  `dosaggio` VARCHAR(255) NULL DEFAULT NULL,
  `CartellaClinica_IDCartella` INT(11) NULL DEFAULT NULL,
  PRIMARY KEY (`IDTerapia`),
  INDEX `fk_Terapia_CartellaClinica1_idx` (`CartellaClinica_IDCartella` ASC) VISIBLE,
  CONSTRAINT `fk_Terapia_CartellaClinica1`
  FOREIGN KEY (`CartellaClinica_IDCartella`)

```

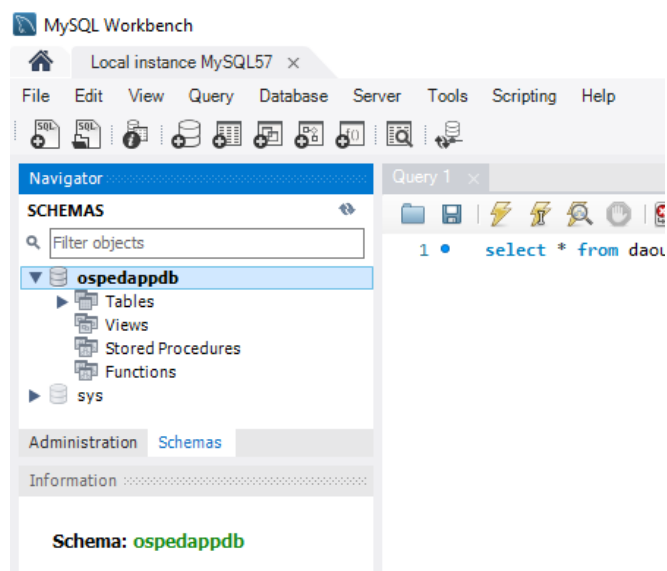
Update Model Ignore Update Source

Table Mapping... Column Mapping...

Back **Next** Cancel



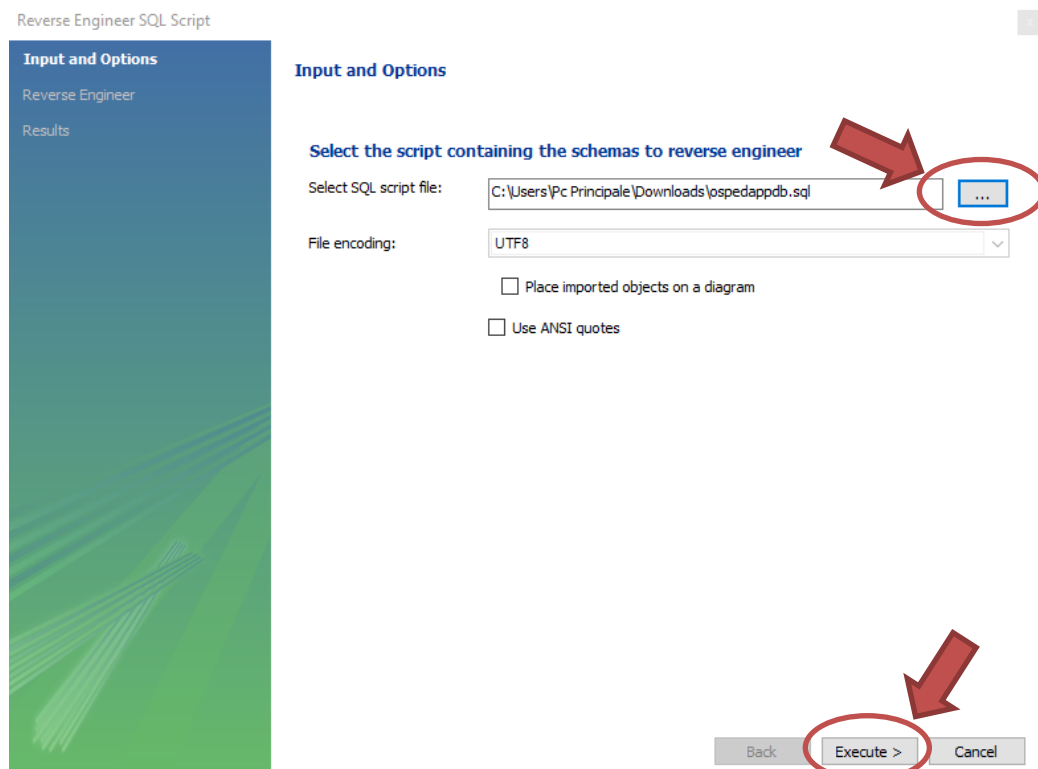
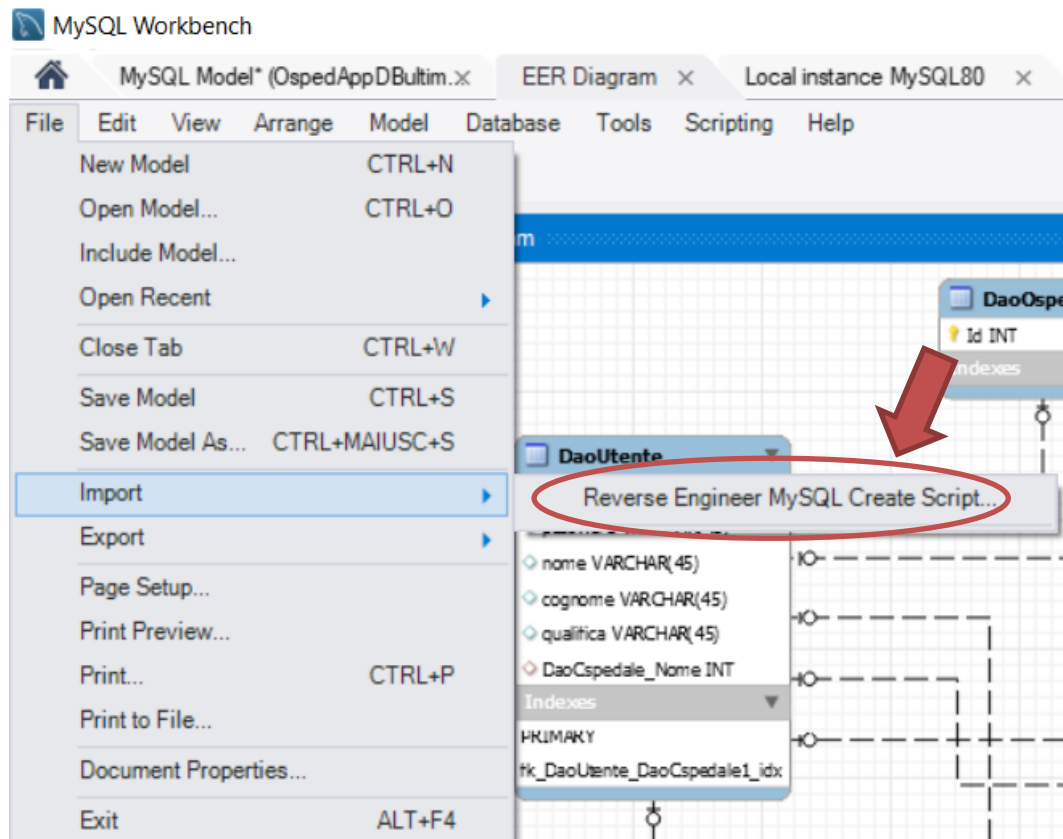
Per rendere effettiva l'attivazione dello schema all'interno del workspace MySQL, è necessario controllare che nella finestra a lato sia presente lo schema e sia evidenziato in grassetto, come mostrato di seguito.



Una volta creato il DB, bisogna creare una tupla nella tabella utente che permetta di effettuare l'accesso. Per semplicità vengono forniti i file con estensione .sql da importare nel DB per effettuare il popolamento delle tabelle dello schema relazionale creato in precedenza.

Per fare ciò è necessario andare nella scheda **File** e selezionare la voce **Import**. Da lì scegliere la voce **Reverse Engineer MySQL Create Script...**. Dal menù selezionare il file sql corretto e procedere con l'esecuzione del file.

Di seguito le istantanee dei passaggi descritti.



Per poter accedere però al DB, è necessario modificare il file di configurazione di hibernate (*hibernate.cfg.xml*) inserendo:

- nel campo “password” la password scelta all’atto dell’installazione di MySQL Workbench
<propertyname="hibernate.connection.password">password</property>;
- nel campo “url” il nome del server, il porto utilizzato e il nome dello schema
<propertyname="hibernate.connection.url">jdbc:mysql://localhost:3306/ospedap
pdb</property>;
- nel campo “username” l’username utilizzata all’atto della creazione del DB
<property name="hibernate.connection.username">root</property>.

Avvio

Server


Per poter utilizzare il sistema è necessario prima di tutto avviare il server mediante il file “Server.jar”: all’avvio verrà mostrato un messaggio di conferma. Per verificare che il server sia stato effettivamente avviato è possibile usare il comando:

```
netstat -ano | findstr : 1099
```

Per terminare il server è possibile utilizzare il comando:

```
taskkill /PID numeroPID /F
```

In alternativa, è possibile terminare l’attività dal task manager.

 OpenJDK Platform binary	0%	106,5 MB	0 MB/s	0 Mbps	0%	Molto basso
---	----	----------	--------	--------	----	-------------

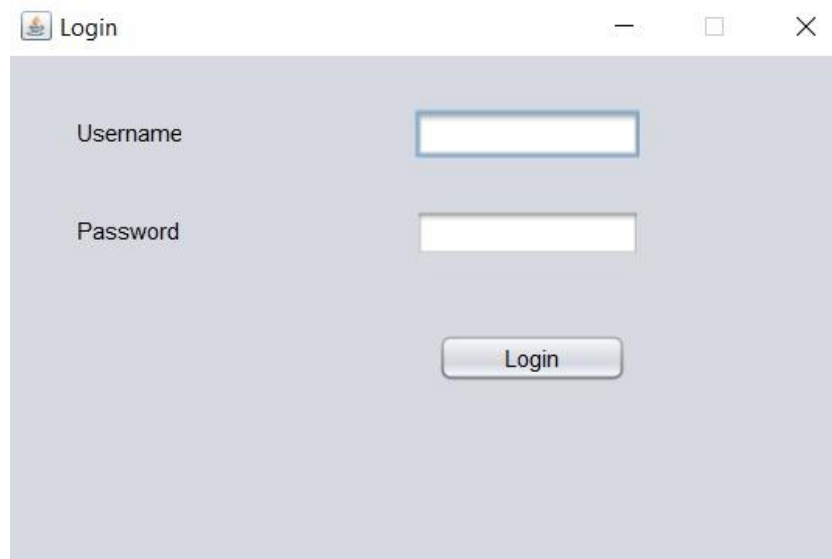
Client

Per avviare l’applicazione è necessario avere installato *Java* e *Java Development Kit*.

Una volta avviata l’applicazione, l’utente dovrà inserire username e password necessari per autenticarsi; a seconda che esso sia un medico o un infermiere, verrà aperta l’apposita interfaccia.

Screen delle interfacce

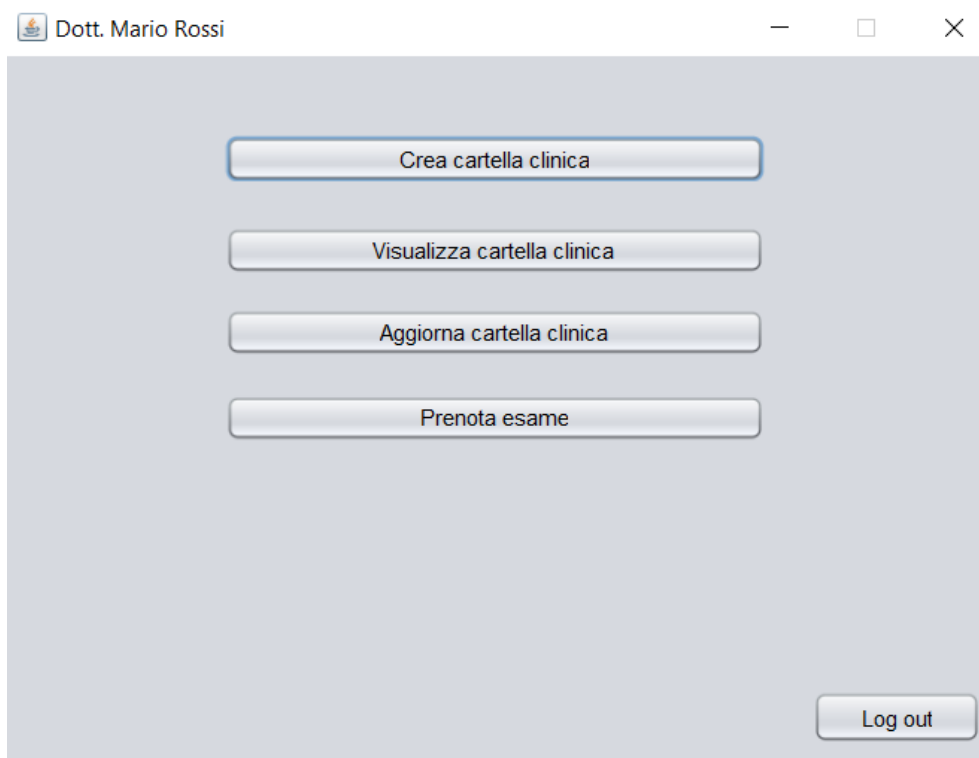
La prima schermata che compare all’atto dell’esecuzione del sistema è quella per effettuare il login, nella quale è possibile inserire username e password.



A screenshot of a login window titled "Login". It features a light blue background. On the left, the labels "Username" and "Password" are positioned next to their respective input fields. The "Username" field is a white rectangle with a blue border. The "Password" field is a white rectangle with a grey border. Below these fields is a "Login" button with a blue gradient and white text. The window has standard OS controls (minimize, maximize, close) in the top right corner.

Se l'utente loggato è un medico, l'interfaccia permetterà diverse operazioni:


- creazione della cartella clinica;
- visualizzazione delle cartelle cliniche di un paziente;
- aggiornamento di una cartella clinica;
- prenotazione di un esame per un paziente.



A screenshot of a medical interface window titled "Dott. Mario Rossi". It has a light blue background. In the center, there are four buttons stacked vertically: "Crea cartella clinica", "Visualizza cartella clinica", "Aggiorna cartella clinica", and "Prenota esame". All buttons have a blue gradient and white text. In the bottom right corner, there is a "Log out" button with a blue gradient and white text. The window has standard OS controls (minimize, maximize, close) in the top right corner.

Di seguito vengono mostrate le varie interfacce utilizzabili da un medico.

Interfaccia di creazione della cartella clinica

 Crea cartella clinica

Dati anagrafici

Nome

Cognome

Recapito

Data di nascita

Luogo di nascita

Numero documento

Codice fiscale

Indirizzo

Tipo documento

Carta di identità

Informazioni ricovero

Motivo del ricovero

Anamnesi patologica prossima

Terapia

Principio attivo


Posologia

Dosaggio

Torna indietro

Crea cartella

Interfaccia di visualizzazione della cartella da parte di un medico

 Visualizza cartella clinica

Codice fiscale paziente da ricercare

Cerca paziente

Dati anagrafici

Nome

Gennaro

Cognome

Esposito

Recapito

081123471

Data di nascita

19/09/2001

Luogo di nascita

Napoli

Numero documento

NA0817548

Codice fiscale

GNNSPS01P19F839G

Indirizzo

Via Tribunli 21

Tipo documento

Carta di identità

Terapie

Terapie effettuate

Terapia 1

:Principio attivo: omeoprazolo Posologia: 3 volte al giorno Dosaggio: 15mg Data inizio: 2020/09/22 17:07:48;

Informazioni paziente

Anamnesi patologica prossima

il paziente presentava sintomi di dolore addominale da alcuni giorni

Motivo ricovero

dolore addominale

Data ricovero

2020/09/22 17:07:48

Torna indietro

Interfaccia di aggiornamento della cartella clinica

Aggiorna cartella clinica

Codice fiscale paziente: GNNSPS01P19F839G Cerca paziente

Dati anagrafici
Nome: Gennaro Cognome: Esposito Recapito: 081123471
Data di nascita: 19/09/2001 Luogo di nascita: Napoli Numero documento: NA0817548
Codice fiscale: INSPS01P19F839G Indirizzo: Via Tribunali 21 Tipo documento: Carta di identità

Informazioni ricovero
Motivo del ricovero: dolore addominale Data ricovero: 2020/09/22 17:07:48
Anamnesi patologica prossima: il paziente presentava sintomi di dolore addominale da alcuni giorni
☒ Cartella attiva

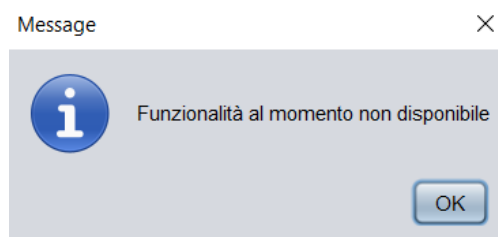
Terapie
Terapie in corso: Terapia 1
:Principio attivo: omeoprazolo Posologia: 3 volte al giorno Dosaggio: 15mg Data inizio: 2020/09/22

Aggiungi terapia
Principio attivo: Posologia: Dosaggio:

Torna indietro Aggiorna cartella

Interfaccia di prenotazione di un esame

Questa funzionalità non è stata ancora implementata, e pertanto rappresenta un possibile sviluppo futuro.



Interfaccia di visualizzazione della cartella da parte di un infermiere

Se invece l'utente loggato è un infermiere potrà solamente visualizzare la cartella clinica attiva del paziente di interesse.

Visualizza cartella clinica

Codice fiscale paziente da ricercare

Dati anagrafici

Nome	Cognome	Recapito
Data di nascita	Luogo di nascita	Numero documento
Codice fiscale	Indirizzo	Tipo documento

Terapie

Terapie effettuate

Informazioni paziente

Anamnesi patologica prossima

Motivo ricovero

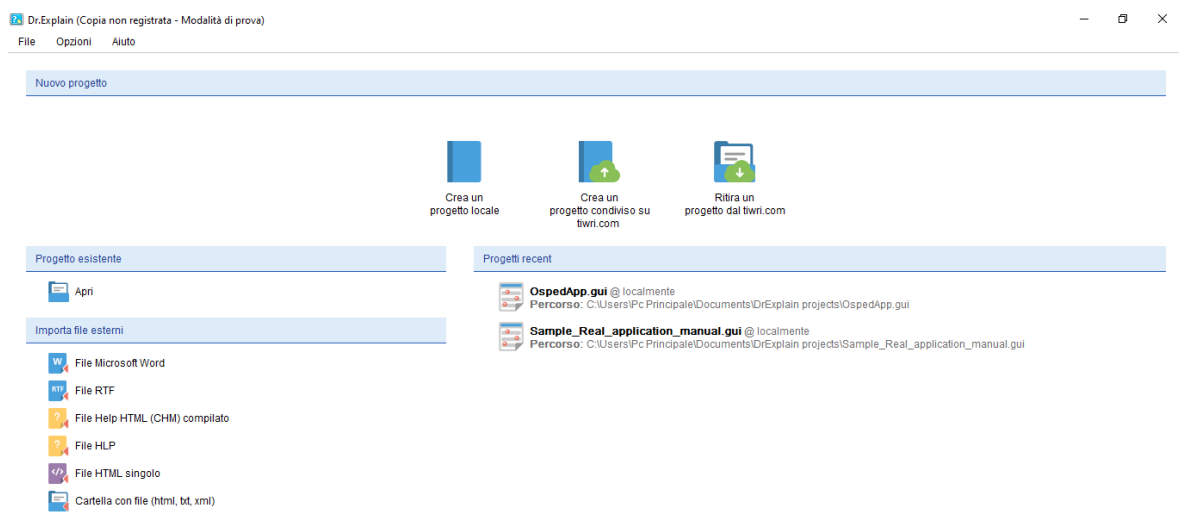
Data ricovero

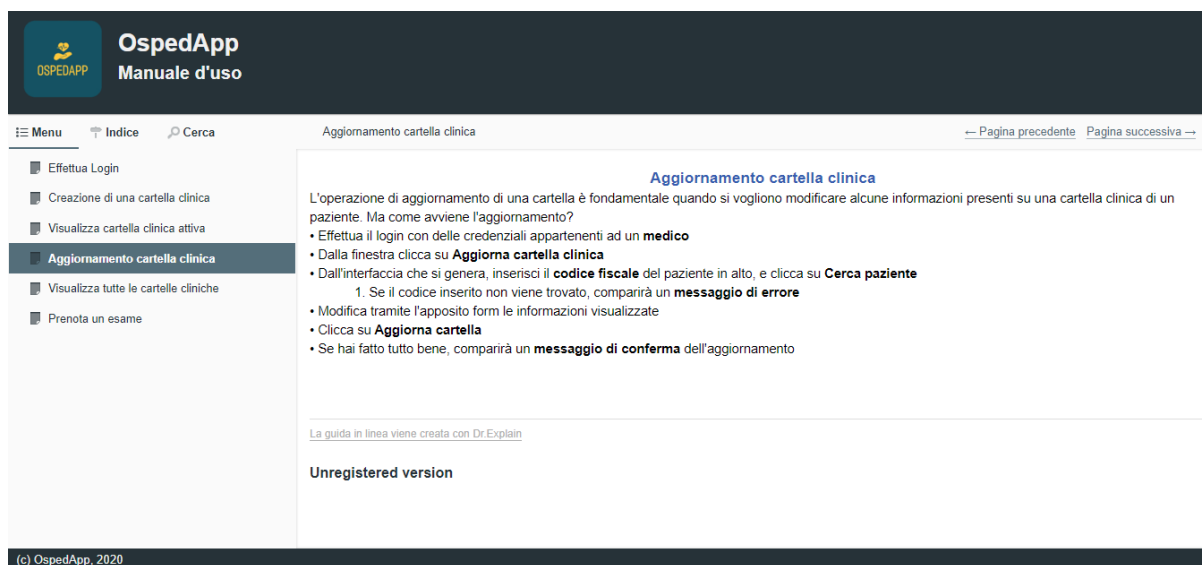
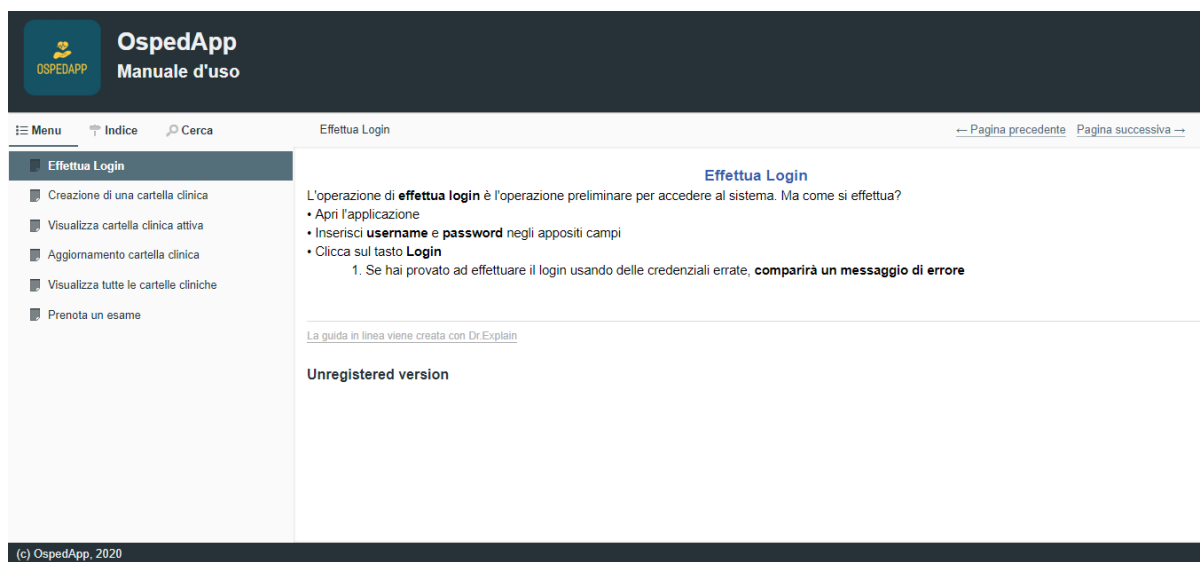
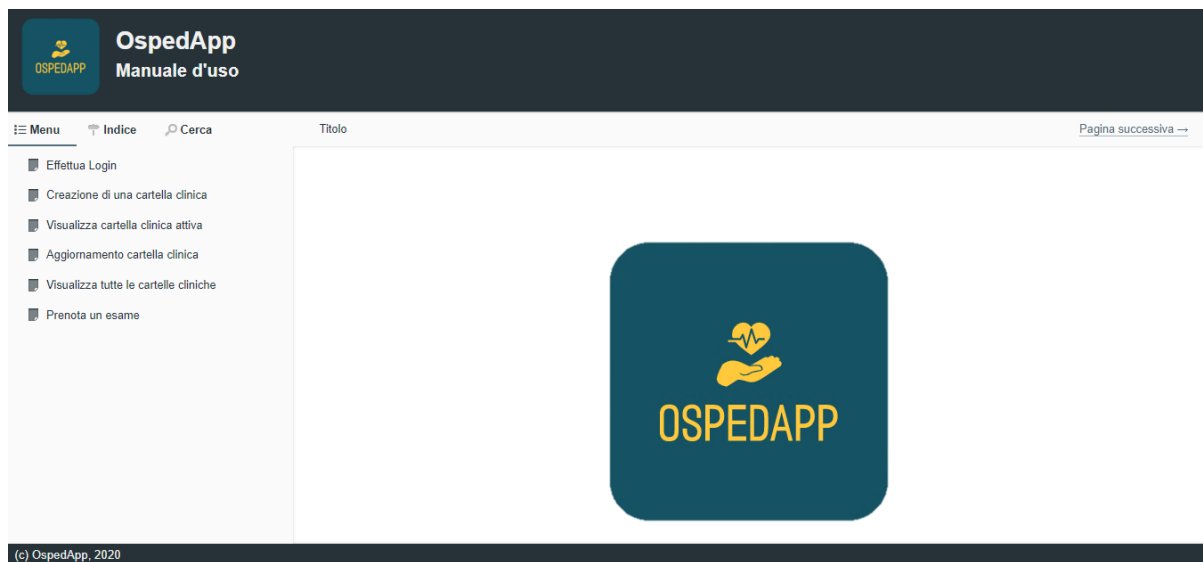
Creazione di una guida interattiva tramite DrExplain

Per creare una documentazione di supporto all'utente è possibile fare uso di strumenti che generano delle guide per un software, come nel caso di *DrExplain*.

Grazie all'utilizzo di questo tool grafico è stato possibile definire un file in formato HTML, rilasciato in allegato, all'interno del quale è possibile visualizzare una guida interattiva dell'intero sistema.

Di seguito alcune schermate dell'applicativo, e di quello che è stato realizzato.





Andando a vedere nel dettaglio il manuale, nel menù a sinistra è possibile scorgere differenti sezioni, ognuna delle quali spiega in maniera riassuntiva le operazioni da eseguire per portare a termine le varie operazioni.

In allegato è possibile trovare i file HTML all'interno di una cartella specifica. Per far sì che il manuale venga visualizzato in maniera corretta è possibile avviare la pagina iniziale, cioè quella che presenta la dicitura **index.html**. Una volta aperta la pagina iniziale, è possibile navigare all'interno del file tramite gli appositi link all'interno dell'interfaccia.

Capitolo 9 : Testing

Il testing è un'attività fondamentale del ciclo di sviluppo del sistema software e deve essere eseguita alla fine di ciascuna iterazione per verificare la presenza di errori o imperfezioni nel codice che potrebbero portare ad un sistema poco funzionale e poco affidabile.

Nel corso dello sviluppo di *OspedApp* sono stati eseguiti diversi test, che sono di seguito documentati.

ID Test	Azione	Risultati attesi	Risultati ottenuti	Esito
1	Login Utente	Apertura dell'interfaccia associata al medico/infermiere e/o visualizzazione del messaggio di errore se l'utente non è presente nel database.	Apertura dell'interfaccia corretta e visualizzazione del messaggio di errore se l'utente non è presente nel database.	✓
2	Creazione cartella clinica	Inserimento nel DB di una nuova cartella clinica.	JDBCConnection Exception	✗
3	Creazione cartella clinica	Inserimento nel DB di una nuova cartella clinica.	SQL Exception: Unknown column in field list.	✗
4	Creazione cartella clinica	Inserimento nel DB di una nuova cartella clinica.	org.hibernate.InstantiationException	✗
5	Creazione cartella clinica	Inserimento nel DB di una nuova cartella clinica	Dati inseriti correttamente nel DB.	✓
6	Visualizza cartella clinica attiva	Visualizzazione dei dati memorizzati nella cartella clinica attiva del paziente.	Visualizzazione dei dati anagrafici del paziente ma non delle terapie.	✗
7	Visualizza cartella clinica attiva	Visualizzazione dei dati memorizzati nella cartella clinica attiva del paziente.	OutOfBoundException.	✗
8	Visualizza cartella clinica attiva	Visualizzazione dei dati memorizzati nella cartella clinica attiva del paziente.	Visualizzazione dei dati anagrafici del paziente ma non delle terapie.	✓
9	Aggiorna cartella clinica	Visualizzazione della cartella clinica associata al codice fiscale inserito e modifica dei dati memorizzati nel DB.	SQL Exception: Unknown column in field list.	✗

10	Aggiorna cartella clinica	Visualizzazione della cartella clinica associata al codice fiscale inserito e modifica dei dati memorizzati nel DB.	Java.lang.IllegalArgoumentException	✗
11	Aggiorna cartella clinica	Visualizzazione della cartella clinica associata al codice fiscale inserito e modifica dei dati memorizzati nel DB.	JDBCConnection Exception	✗
12	Aggiorna cartella clinica	Visualizzazione della cartella clinica associata al codice fiscale inserito e modifica dei dati memorizzati nel DB.	La ricerca del paziente per codice fiscale va a buon fine ma non vengono mostrati i campi	✗
13	Aggiorna cartella clinica	Visualizzazione della cartella clinica associata al codice fiscale inserito e modifica dei dati memorizzati nel DB.	Tutti i campi vengono mostrati correttamente ma le modifiche non vengono salvate nel DB.	✗
14	Aggiorna cartella clinica	Salvataggio dei dati modificati nel DB.	Hibernate: ids for this class must be manually assigned before calling saveOrUpdate ().	✗
15	Aggiorna cartella clinica	Visualizzazione della cartella clinica associata al codice fiscale inserito e modifica dei dati memorizzati nel DB.	Le modifiche vengono memorizzate correttamente nel DB.	✓
16	Medico clicca su "Crea cartella clinica"	Apertura dell'interfaccia associata alla creazione della cartella clinica.	Apertura dell'interfaccia corretta.	✓
17	Medico clicca su "Aggiorna cartella clinica"	Apertura dell'interfaccia associata all'aggiornamento della cartella clinica.	Apertura dell'interfaccia corretta.	✓
18	Inserimento codice fiscale nel DB	Il codice fiscale viene memorizzato nel database con tutti caratteri maiuscoli anche se inserito con caratteri minuscoli.	Il codice viene memorizzato nel database con tutti caratteri maiuscoli anche se inserito con caratteri minuscoli.	✓
19	Ricerca di un paziente per aggiornamento o visualizzazione cartella clinica	Se il codice fiscale non è presente nel database viene mostrato un messaggio di errore.	Se il codice fiscale non è presente nel database viene mostrato un messaggio di errore.	✓

20	Inserimento parametri scorretti	Visualizzazione di un messaggio d'errore che implica il reinserimento dei parametri.	Visualizzazione del messaggio di errore.	✓
21	Inserimento nome contenente valori numerici	Visualizzazione di un messaggio d'errore che segnala l'errato inserimento.	Visualizzazione del messaggio di errore corretto.	✓
22	Inserimento cognome contenente valori numerici	Visualizzazione di un messaggio d'errore che segnala l'errato inserimento.	Visualizzazione del messaggio di errore corretto.	✓
23	Inserimento recapito telefonico contenente lettere	Visualizzazione di un messaggio d'errore che segnala l'errato inserimento.	Visualizzazione del messaggio di errore corretto.	✓
24	Inserimento anamnesi con un numero di caratteri maggiore di 255	Visualizzazione di un messaggio d'errore che segnala la lunghezza eccessiva del campo anamnesi.	Visualizzazione del messaggio di errore corretto.	✓
25	Aggiornamento di una cartella clinica	Visualizzazione di un messaggio che segnala l'avvenuto aggiornamento della cartella.	Visualizzazione del messaggio di errore corretto.	✓
26	Inserimento di un codice fiscale errato	Visualizzazione di un messaggio di errore che segnala il codice fiscale errato.	Visualizzazione del messaggio di errore corretto.	✓
27	Testing di integrazione	Funzionamento corretto di tutte le funzionalità offerte dal sistema.	Funzionamento corretto.	✓

Conseguentemente ai test effettuati, tutti gli eventuali errori e bug riscontrati sono stati eliminati.

Capitolo 10 : Sviluppi futuri

Nelle successive iterazioni saranno implementati:

- il caso d'uso "*Visualizza Cartelle Paziente*" che permette ad un medico di visualizzare tutte le cartelle cliniche associate ad un paziente ricoverato;
- il caso d'uso "*Richiedi Esame*" che permette al medico di prenotare un esame per un paziente.

Inoltre, per come è stato implementato il database, è possibile implementare sia estensioni del sistema, sia la comunicazione con altri sistemi responsabili di altri compiti. Il database infatti è predisposto per la memorizzazione dei medici e degli infermieri che visualizzano/modificano la cartella clinica e per la memorizzazione della data in cui termina una terapia. A tal proposito potrebbe essere utile anche l'inserimento di nuovi attori nel sistema, come il primario, il tecnico di laboratorio, il personale amministrativo e il personale di sala operatoria. Per sviluppi futuri si può anche pensare di ampliare la cartella clinica, aggiungendo nuovi campi, e aggiungere una serie di controlli per rendere il sistema maggiormente affidabile e sicuro.