



UNIVERSITÀ
POLITECNICA
DELLE MARCHE

SISTEMA INFORMATIVO PER LA GESTIONE DI UNA STRUTTURA OSPEDALIERA

Facoltà di Ingegneria

Corso di Ingegneria del Software



Autori: **Emanuela Saleggia, Valeria Timmer,
Viola Terribili, Francesco Voto**

Prof. Ing. Domenico Ursino

SOMMARIO

INTERVISTA.....	3
DESCRIZIONE del sistema in linguaggio naturale	5
GLOSSARIO.....	6
DIAGRAMMA DEI sistemi	8
ANALISI DEI REQUISITI	9
	9
REQUISITI FUNZIONALI	9
REQUISITI NON FUNZIONALI	12
DIAGRAMMA DEI CASI D'USO	13
DIAGRAMMA COMPLESSIVO DEI CASI D'USO	14
ATTORI: AMMINISTRATORE DELL'UFFICIO DI ACCETTAZIONE -	
AMMINISTRATORE DEL PRONTO SOCCORSO	15
ATTORI: INFERMIERE – MEDICO.....	32
MAPPA DELL'ARCHITETTURA	36
MATRICE DI MAPPING	37
DIAGRAMMI DELLE CLASSI.....	38
DIAGRAMMA COMPLESSIVO DELLE CLASSI	38
AUTENTICAZIONE - HOME	39
OPERATORE	40
PAZIENTE	41
SERVIZIO	42
PRENOTAZIONE.....	43
DIAGRAMMI DELLE SEQUENZE:.....	44
OPERATORI	44
SERVIZI	45
RICOVERI:.....	47
DIAGRAMMI DELLE ATTIVITA'	51
OPERATORE.....	51
PAZIENTE	52
SERVIZIO	53
PRENOTAZIONE.....	54
Il seguenti diagrammi mostrano le attività legate alle prenotazioni.	54
MOCKUP di livello zero	57
Implementazione DEL SOFTWARE.....	61
TESTING DEL SISTEMA.....	71
CONCLUSIONE	74
Note.....	74

INTERVISTA

Per quale scopo viene progettato il software ?

L'amministratore dell'accettazione della struttura ospedaliera presa in analisi è alla ricerca di uno strumento attraverso il quale riesca a gestire, con maggior facilità, i ricoveri all'interno dell'ospedale. Infatti il software dovrà registrare tutti i pazienti ricoverati nella struttura, e anche le prenotazioni future dei ricoveri.

Com'è suddiviso l'ospedale? Quali reparti vi sono?

L'ospedale è suddiviso in cinque reparti, che si trovano in piani diversi. Inoltre, a piano terra si trova l'ufficio di accettazione ed il pronto soccorso. I reparti sono quelli di oncologia, chirurgia, cardiologia, medicina e riabilitazione. Ad ogni reparto è associata una lettera diversa, in modo tale da distinguerli con più precisione: al reparto di *oncologia* corrisponde la lettera "a", a quello di *chirurgia* la lettera "b", a quello di *cardiologia* la lettera "c", a quello di *medicina* la lettera "d" e a quello di *riabilitazione* la lettera "e". Di conseguenza, ogni posto letto di ogni reparto è identificato dalla rispettiva lettera e da un numero.

Quali sono le attività che deve gestire il responsabile?

L'amministratore/responsabile è colui che si occupa dell'ingresso e uscita dei pazienti dalla struttura, dell'assegnazione dei posti letto e di un codice identificativo che deriva dai dati anagrafici (nome e cognome) rilasciati dal paziente al momento dell'accettazione; è colui che si occupa delle prenotazioni dei ricoveri a lungo e breve termine. Con questo codice il responsabile è in grado di conoscere l'effettiva permanenza dei degenti nei vari reparti (oncologia, chirurgia, cardiologia, medicina e riabilitazione) e può pianificare i successivi ricoveri, conosce con esattezza quanti posti letto sono liberi e quanti occupati. Inoltre dovrà essere concesso all'amministratore di ricercare nel software il nome di un paziente, o di uno specifico servizio, in modo da ottimizzare i tempi. Infatti tutte le funzionalità del programma servono proprio per rendere più veloce e affidabile il meccanismo di

salvataggio dei pazienti e di tutte le informazioni connesse ai vari ricoveri.

Vi è un unico amministratore all'interno della struttura ospedaliera ?

No, è presente anche l'amministratore del pronto soccorso che organizza gli arrivi al pronto soccorso, in particolare si occupa dei ricoveri d'emergenza e dell'assegnazione dei posti letto d'emergenza, che sono presenti in ogni reparto della struttura ospedaliera; anche lui è in grado di assegnare un codice identificativo (nome e cognome). I casi che vengono ritenuti meno gravi sono gestiti dall'amministratore principale che si occuperà dello smistamento del paziente nel reparto idoneo alla sua necessità, oltre all'assegnazione del codice identificativo.

Nel caso in cui i posti letto d'emergenza risultano essere tutti occupati, l'amministratore del pronto soccorso si occupa di richiedere un'ambulanza per trasportare il paziente nella struttura ospedaliera più vicina.

Il codice identificativo accompagnerà il paziente durante tutto l'iter ospedaliero

I medici e gli infermieri possono accedere al software ? Se sì, quali sono i loro compiti ?

I medici e gli infermieri possono accedere al software per controllare la lista dei pazienti ricoverati nei vari reparti. Entrambi possono visualizzare i dati anagrafici e clinici di ogni paziente, ma solo il medico può inserire un nuovo referto. Inoltre al medico, come all'amministratore dell'accettazione, è concesso di visualizzare i ricoveri passati, che devono essere salvati nel sistema.

DESCRIZIONE DEL SISTEMA IN LINGUAGGIO NATURALE

Il progetto consiste nella realizzazione di un sistema informativo per una struttura ospedaliera. In particolar modo ci occuperemo della gestione dei ricoveri all'interno dell'ospedale stesso.

Quest'ultimo è suddiviso in cinque reparti, situati su diversi piani. Ogni reparto svolge un'attività specifica, con dei medici specializzati nei vari settori.

Al piano terra si può trovare l'ufficio di accettazione, luogo in cui si devono recare tutti i pazienti al loro ingresso nella struttura ed il pronto soccorso. Nei piani successivi, sono presenti rispettivamente i reparti di oncologia, chirurgia, cardiologia, medicina e riabilitazione. Ciascuno di essi ha dimensioni differenti, quindi il numero di posti letto dipende dal reparto. Inoltre in ogni settore si possono trovare dei posti letto riservati ai ricoveri urgenti, provenienti dal pronto soccorso. Ad ogni reparto è poi associata una lettera diversa che li distingue: al reparto di *oncologia* è associata la lettera "a", a quello di *chirurgia* la lettera "b", a quello di *cardiologia* la lettera "c", a quello di *medicina* la lettera "d" e a quello di *riabilitazione* la lettera "e". Infatti ogni posto letto è identificato dalla rispettiva lettera del reparto a cui appartiene, e da un numero.

I dipendenti della struttura sono due amministratori, uno o più infermieri e medici. Uno dei responsabili si trova all'accettazione, mentre l'altro gestisce il pronto soccorso. Il primo si occupa dell'ingresso e l'uscita dei pazienti dalla struttura, della gestione delle prenotazioni dei ricoveri e anche dell'assegnamento dei posti letto. Il secondo coordina gli arrivi al pronto soccorso registrando i singoli dati e destinando i pazienti in condizioni gravi ai vari reparti. Gli infermieri, invece, si occupano di accompagnare i pazienti nei rispettivi settori della struttura ospedaliera. I paramedici possono inoltre visualizzare la lista dei ricoveri e i dati anagrafici e clinici di ogni paziente, mentre i medici possono anche inserire un referto. A ogni dipendente della struttura deve essere concesso di identificarsi nel sistema, per mezzo di un username e di una password. Di ogni dipendente si devono registrare: nome, cognome, codice fiscale, luogo e data di nascita, email, password e il ruolo che svolge all'interno dell'ospedale. In ugual modo, per ogni paziente, al momento

dell'accettazione, si devono registrare: nome, cognome, luogo e data di nascita, codice fiscale, numero di telefono ed e-mail. Recandosi nell'ufficio di amministrazione una settimana prima del ricovero, ogni paziente deve effettuare la prenotazione e l'accettazione, indicando la data di inizio e di fine permanenza nel reparto desiderato della struttura. La disdetta può essere compiuta fino ad un giorno prima dell'ingresso in ospedale. Nel momento in cui il paziente entra in ospedale o in pronto soccorso per il ricovero gli viene assegnato un codice identificativo dall'amministratore, ovvero l'id.

Ciascun paziente potrà effettuare più prenotazioni in reparti e periodi differenti .

Inoltre, il programma dovrà permettere ai vari dipendenti di ricercare un paziente, un servizio o una prenotazione, per rendere così più veloce la ricerca di informazioni e l'inserimento di nuovi dati. Infatti, il software è pensato proprio per ottimizzare il lavoro dei dipendenti dell'ospedale.

GLOSSARIO

- ***Struttura Ospedaliera***

- Descrizione: Edificio pubblico destinato al ricovero e all'assistenza sanitaria dei pazienti.
- Tipo: Business.
- Sinonimo : Ospedale.

- ***Amministratore***

- Descrizione: Gestore del sistema della struttura. Si occupa dell'accettazione e dell'assegnamento dei ricoveri.
- Tipo: Business.
- Sinonimo: Responsabile.

- ***Codice identificativo***

- Descrizione: Sequenza di lettere costituita dall'unione del nome e del cognome del paziente.
- Tipo: Tecnico.
- Sinonimo: CI e ID.

- ***Infermiere***

- Descrizione: Dipendente della struttura ospedaliera. Si occupa di accompagnare i pazienti nei rispettivi settori dell'ospedale e di visualizzare i loro dati.
- Tipo: Business.
- Sinonimo: Paramedico.

- ***Medico***

- Descrizione: Dipendente della struttura ospedaliera. Colui che può accedere alla cartella clinica dei pazienti.
- Tipo: Business.
- Sinonimo: Nessuno.

- ***Reparto***

- Descrizione: Piano dell'ospedale specializzato. Luogo in cui vengono ricoverati i pazienti.
- Tipo: Business.
- Sinonimo: Settore.

- ***Paziente***

- Descrizione: Colui che viene ricoverato nella struttura ospedaliera.
- Tipo: Business.
- Sinonimo: Degente.

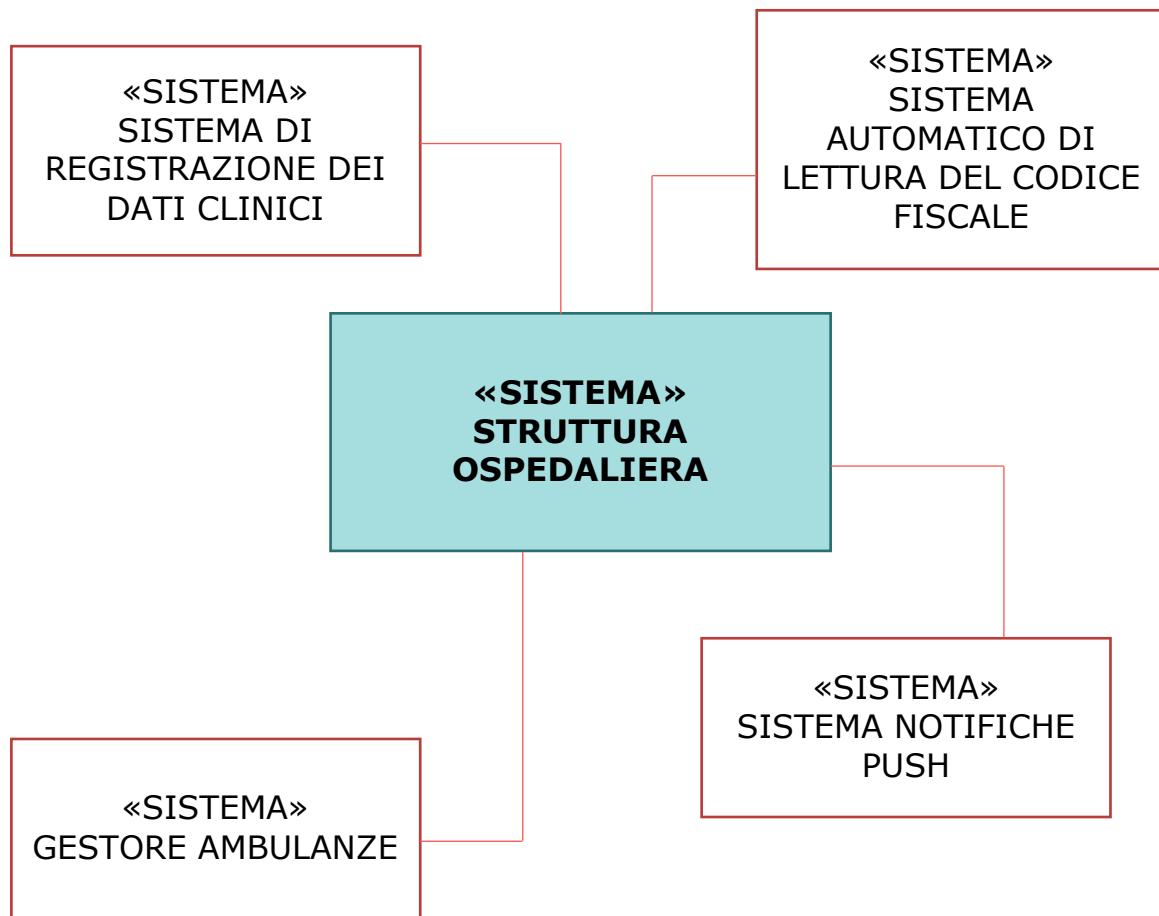
- ***Ricovero***

- Descrizione: Periodo di permanenza di un paziente all'interno di un reparto dell'ospedale
- Tipo: Business.
- Sinonimo: Nessuno.

- ***Operatore***

- Descrizione: Colui che attualmente è un impiegato della struttura ospedaliera
- Tipo: Business.
- Sinonimo: Dipendente.

DIAGRAMMA DEI SISTEMI



ANALISI DEI REQUISITI

Analisi dei requisiti

Funzionali

- + Gestione Pazienti
- + Gestione Operatori
- + Gestione Attività

Non Funzionali

- + Requisiti Non Funzionali

REQUISITI FUNZIONALI

Requisiti Funzionali

Gestione Pazienti

- + RF1 Visualizzazione Paziente
- + RF2 Inserimento Nuovo Paziente
- + RF3 Ricerca Paziente
- + RF4 Modifica Paziente

Gestione Operatori

- + RF5 Visualizzazione Operatore
- + RF6 Inserimento Nuovo Operatore
- + RF7 Eliminazione Operatore
- + RF8 Ricerca Operatore
- + RF9 Modifica Operatore

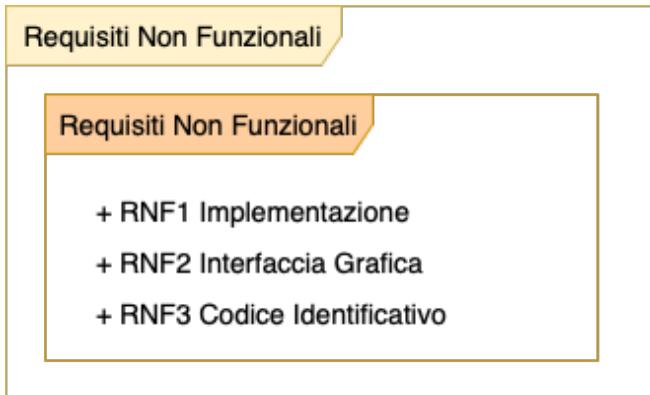
Gestione Attività

- + RF10 Visualizzazione Ricovero
- + RF11 Prenotazione Nuovo Ricovero
- + RF12 Visualizzazione Ricovero di Emergenza
- + RF13 Prenotazione Ricovero di Emergenza
- + RF14 Disdetta Prenotazione Ricovero
- + RF15 Modifica Ricovero
- + RF16 Visualizzazione Ricoveri Archiviati
- + RF17 Visualizzazione Disponibilità
- + RF18 Ricerca Ricovero
- + RF19 Ricerca Ricovero di Emergenza
- + RF20 Visualizzazione Dati Paziente
- + RF21 Modifica Dati Paziente
- + RF22 Visualizzazione Pazienti nel Reparto
- + RF23 Visualizzazione Servizio
- + RF24 Visualizzazione Servizio di Emergenza
- + RF25 Inserimento Nuovo Servizio
- + RF26 Eliminazione Servizio
- + RF27 Ricerca Servizio
- + RF28 Ricerca Servizio di Emergenza

• Requisiti Funzionali	• Descrizione
• RF1 Visualizzazione Paziente	• <i>Il sistema dovrà consentire la visualizzazione dei dati anagrafici di un paziente.</i>
• RF2 Inserimento Nuovo Paziente	• <i>Il sistema dovrà consentire l'inserimento di un nuovo paziente.</i>
• RF3 Ricerca Paziente	• <i>Il sistema dovrà consentire di ricercare un paziente all'interno della lista dei pazienti attraverso nome, cognome o codice fiscale.</i>
• RF4 Modifica Paziente	• <i>Il sistema dovrà consentire di modificare i dati anagrafici di un paziente.</i>
• RF5 Visualizzazione Operatore	• <i>Il sistema dovrà consentire la visualizzazione di un operatore.</i>
• RF6 Inserimento Nuovo Operatore	• <i>Il sistema dovrà consentire l'inserimento di un nuovo operatore.</i>
• RF7 Eliminazione Operatore	• <i>Il sistema dovrà consentire l'eliminazione di un operatore dalla lista degli operatori.</i>
• RF8 Ricerca Operatore	• <i>Il sistema dovrà consentire la ricerca di un operatore all'interno della lista degli operatori attraverso nome o cognome.</i>
• RF9 Modifica Operatore	• <i>Il sistema dovrà consentire di modificare i dati anagrafici di un operatore.</i>
• RF10 Visualizzazione Ricovero	• <i>Il sistema dovrà permettere di visualizzare le informazioni relative ad un determinato ricovero.</i>
• RF11 Prenotazione Nuovo Ricovero	• <i>Il sistema dovrà consentire di prenotare un nuovo ricovero nella struttura.</i>
• RF12 Visualizzazione Ricovero di Emergenza	• <i>Il sistema dovrà permettere di visualizzare le informazioni relative ad un determinato ricovero di emergenza.</i>
• RF13 Prenotazione Ricovero di Emergenza	• <i>Il sistema dovrà consentire di prenotare un ricovero di emergenza nella struttura.</i>
• RF14 Disdetta Prenotazione Ricovero	• <i>Il sistema dovrà consentire di disdire una prenotazione di un ricovero non ancora in corso.</i>

<ul style="list-style-type: none"> RF15 Modifica Ricovero 	<ul style="list-style-type: none"> <i>Il sistema dovrà consentire di modificare la data di fine di un ricovero in corso (consente di liberare il servizio occupato).</i>
<ul style="list-style-type: none"> RF16 Visualizzazione Ricoveri Archiviati 	<ul style="list-style-type: none"> <i>Il sistema dovrà consentire la visualizzazione dei ricoveri passati.</i>
<ul style="list-style-type: none"> RF17 Visualizzazione Disponibilità 	<ul style="list-style-type: none"> <i>Il sistema dovrà consentire di visualizzare il numero dei servizi disponibili e occupati.</i>
<ul style="list-style-type: none"> RF18 Ricerca Ricovero 	<ul style="list-style-type: none"> <i>Il sistema dovrà consentire la ricerca di un ricovero all'interno della lista dei ricoveri.</i>
<ul style="list-style-type: none"> RF19 Ricerca Ricovero di Emergenza 	<ul style="list-style-type: none"> <i>Il sistema dovrà consentire la ricerca di un ricovero all'interno della lista dei ricoveri di emergenza.</i>
<ul style="list-style-type: none"> RF20 Visualizzazione Dati Paziente 	<ul style="list-style-type: none"> <i>Il sistema dovrà consentire di visualizzare i dati clinici.</i>
<ul style="list-style-type: none"> RF21 Modifica Dati Paziente 	<ul style="list-style-type: none"> <i>Il sistema dovrà consentire di modificare i dati clinici di un determinato paziente.</i>
<ul style="list-style-type: none"> RF22 Visualizzazione Pazienti nel Reparto 	<ul style="list-style-type: none"> <i>Il sistema dovrà consentire la visualizzazione di tutti i pazienti presenti in un reparto.</i>
<ul style="list-style-type: none"> RF23 Visualizzazione Servizio 	<ul style="list-style-type: none"> <i>Il sistema dovrà consentire la visualizzazione dei dati di ciascun servizio della struttura.</i>
<ul style="list-style-type: none"> RF24 Visualizzazione Servizio di Emergenza 	<ul style="list-style-type: none"> <i>Il sistema dovrà consentire la visualizzazione dei dati di un servizio di emergenza.</i>
<ul style="list-style-type: none"> RF25 Inserimento Nuovo Servizio 	<ul style="list-style-type: none"> <i>Il sistema dovrà consentire l'inserimento di un nuovo servizio.</i>
<ul style="list-style-type: none"> RF26 Eliminazione Servizio 	<ul style="list-style-type: none"> <i>Il sistema dovrà consentire l'eliminazione di un servizio.</i>
<ul style="list-style-type: none"> RF27 Ricerca Servizio 	<ul style="list-style-type: none"> <i>Il sistema dovrà consentire la ricerca di un determinato servizio all'interno della lista dei servizi tramite posto letto e reparto.</i>
<ul style="list-style-type: none"> RF28 Ricerca Servizio di Emergenza 	<ul style="list-style-type: none"> <i>Il sistema dovrà consentire la ricerca di un determinato servizio all'interno della lista dei servizi di emergenza tramite posto letto e reparto.</i>

REQUISITI NON FUNZIONALI



• Requisiti Non Funzionali	• Descrizione
• RFN1 Implementazione	• <i>Il sistema verrà realizzato in tecnologia Python 3.</i>
• RFN2 Interfaccia Grafica	• <i>Il sistema sarà dotato di interfaccia grafica</i>
• Codice identificativo	• <i>Il sistema dovrà consentire di utilizzare un codice univoco per i servizi della struttura (tipo, reparto e posto letto), per i pazienti e gli operatori (codice fiscale) e per i ricoveri (codice fiscale del paziente e nome del servizio).</i>

DIAGRAMMA DEI CASI D'USO

Diagramma dei casi d'uso

Attori

- + Amministratore dell'ufficio di accettazione
- + Amministratore del pronto soccorso
- + Infermiere
- + Medico

Gestione Pazienti

- + Visualizzazione Paziente
- + Inserimento Nuovo Paziente
- + Ricerca Paziente
- + Modifica Paziente

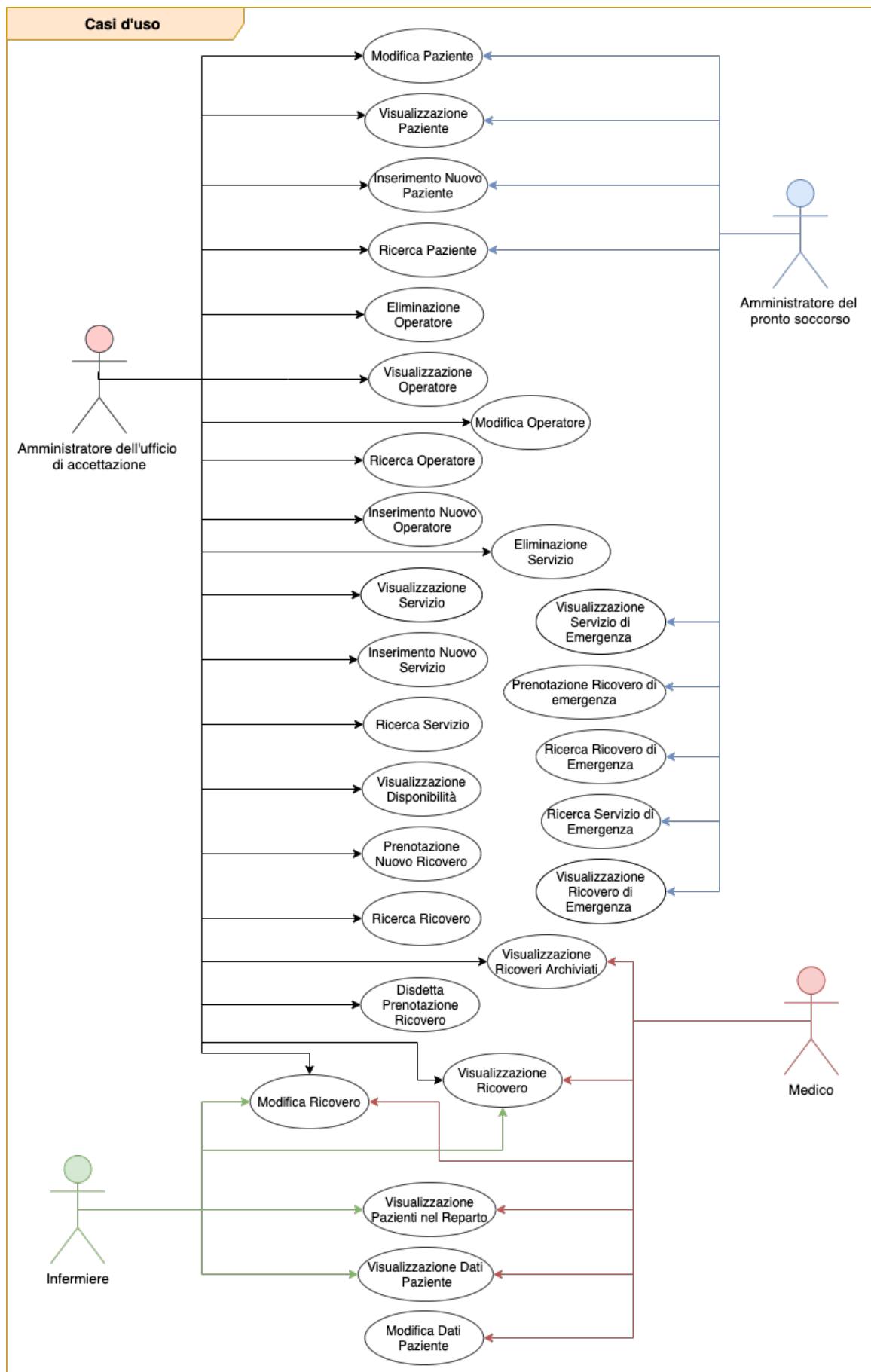
Gestione Operatori

- + Visualizzazione Operatore
- + Inserimento Nuovo Operatore
- + Eliminazione Operatore
- + Ricerca Operatore
- + Modifica Operatore

Gestione Attività

- + Visualizzazione Ricovero
- + Prenotazione Nuovo Ricovero
- + Visualizzazione Ricovero di Emergenza
- + Prenotazione Ricovero di Emergenza
- + Disdetta Prenotazione Ricovero
- + Modifica Ricovero
- + Visualizzazione Ricoveri Archiviati
- + Visualizzazione Disponibilità
- + Ricerca Ricovero
- + Ricerca Ricovero di Emergenza
- + Visualizzazione Dati Paziente
- + Modifica Dati Paziente
- + Visualizzazione Pazienti nel Reparto
- + Visualizzazione Servizio
- + Visualizzazione Servizio di Emergenza
- + Inserimento Nuovo Servizio
- + Eliminazione Servizio
- + Ricerca Servizio
- + Ricerca Servizio di Emergenza

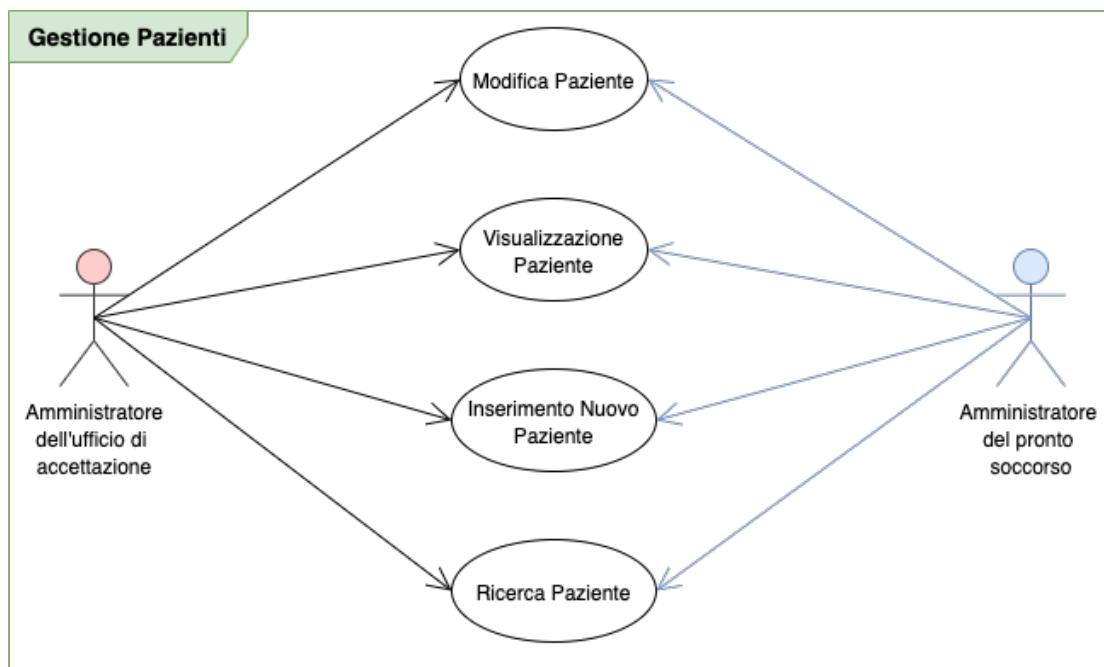
DIAGRAMMA COMPLESSIVO DEI CASI D'USO



DESCRIZIONE DEI CASI D'USO

ATTORI: AMMINISTRATORE DELL'UFFICIO DI ACCETTAZIONE - AMMINISTRATORE DEL PRONTO SOCCORSO

CASI D'USO: Gestione Pazienti



Caso d'Uso AMMINISTRATORE DELL'UFFICIO DI ACCETTAZIONE - AMMINISTRATORE DEL PRONTO SOCCORSO: Inserimento Nuovo Paziente

Questo caso d'uso si verifica quando un amministratore deve inserire un nuovo paziente nel sistema.

Pre-condizioni: Il paziente non esiste nel sistema.

Post-condizioni: Il nuovo paziente esiste nel sistema (a meno che si sia verificata l'impossibilità di aggiungerlo).

Sequenza degli eventi principali:

- 1) Il caso d'uso inizia quando un amministratore deve inserire i dati di un nuovo paziente
- 2) Il sistema visualizza la schermata di inserimento dati del nuovo paziente

- 3) L'amministratore fornisce i dati richiesti del paziente, ovvero: nome e cognome, sesso, data e luogo di nascita, codice fiscale ed email
- 4) L'amministratore avvia la procedura di inserimento del nuovo paziente
- 5) Il paziente viene inserito nel sistema

Sequenza degli eventi alternativa: La sequenza alternativa inizia dal punto 4:

- 6) Le informazioni inserite dall'amministratore sono incomplete
- 7) L'inserimento del nuovo paziente fallisce a causa dell'inserimento di informazioni non corrette

Caso d'Uso AMMINISTRATORE DELL'UFFICIO DI ACCETTAZIONE – AMMINISTRATORE DEL PRONTO SOCCORSO: Visualizzazione Paziente

Questo caso d'uso si verifica quando un amministratore vuole visualizzare i dati (anagrafici) di un determinato paziente.

Pre-condizioni : Il paziente esiste nel sistema.

Post-condizioni : Nessuna.

Sequenza degli eventi principali :

- 1) Il caso d'uso inizia quando un amministratore vuole visualizzare i dati di un determinato paziente
- 2) Il sistema legge le informazioni del paziente scelto dall' amministratore
- 3) Il sistema visualizza a schermo le informazioni del paziente

Sequenza degli eventi alternativa : Nessuna

Caso d'Uso AMMINISTRATORE DELL'UFFICIO DI ACCETTAZIONE – AMMINISTRATORE DEL PRONTO SOCCORSO: Modifica Paziente

Questo caso d'uso si verifica qualora un amministratore debba modificare i dati anagrafici di un determinato paziente già presente nel sistema.

Pre-condizioni: Il paziente esiste nel sistema.

Post-condizioni: I dati del paziente vengono modificati e aggiornati nel sistema.

Sequenza degli eventi principali :

- 1) Il caso d'uso inizia quando un amministratore vuole visualizzare i dati di un paziente già registrato
- 2) Il sistema legge le informazioni del paziente scelto dall'amministratore
- 3) Il sistema visualizza a schermo le informazioni del paziente
- 4) L'amministratore modifica i dati del paziente
- 5) L'amministratore conferma le nuove modifiche

Sequenza degli eventi alternativa: Nessuna

Caso d'Uso AMMINISTRATORE DELL'UFFICIO DI ACCETTAZIONE – AMMINISTRATORE DEL PRONTO SOCCORSO: Ricerca Paziente

Questo caso d'uso si verifica quando un amministratore vuole cercare un determinato paziente all'interno della lista dei degenzi registrati nella struttura.

Pre-condizioni : Il paziente esiste nel sistema.

Post-condizioni : Nessuna.

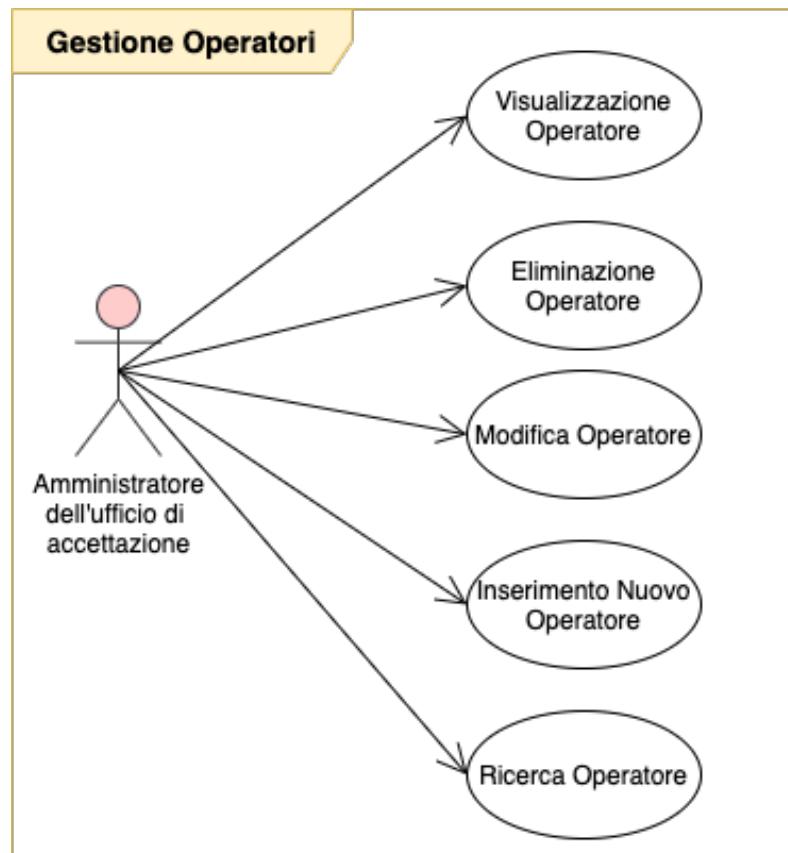
Sequenza degli eventi principali :

- 1) Il caso d'uso inizia quando un amministratore vuole cercare un determinato paziente nella lista dei degenzi

- 2) Il sistema legge i dati relativi al paziente inseriti dall'amministratore (nome, cognome, codice fiscale)
- 3) Il sistema seleziona tutti i pazienti presenti a sistema con le caratteristiche inserite dall'amministratore
- 4) L'amministratore può visualizzare tutte le informazioni raccolte dal sistema

Sequenza degli eventi alternativa : Nessuna

CASI D'USO: Gestione Operatori



Caso d'Uso AMMINISTRATORE DELL'UFFICIO DI ACCETTAZIONE: Visualizzazione Operatore

Questo caso d'uso si verifica quando l'amministratore dell'ufficio di accettazione vuole visualizzare le informazioni relative a un operatore della struttura ospedaliera.

Pre-condizioni: L'operatore esiste nel sistema

Post-condizioni: Nessuna

Sequenza degli eventi principali:

- 1) Il caso d'uso inizia quando l'amministratore dell'ufficio di accettazione vuole visualizzare le informazioni relative ad un operatore
- 2) Il sistema legge le informazioni dell'operatore scelto dall'amministratore
- 3) Il sistema visualizza a schermo le informazioni dell'operatore

Sequenza alternativa: Nessuna

Caso d'Uso AMMINISTRATORE DELL'UFFICIO DI ACCETTAZIONE: Inserimento Nuovo Operatore

Questo caso d'uso si verifica quando l'amministratore dell'ufficio di accettazione deve inserire un nuovo operatore che lavorerà all'interno della struttura ospedaliera

Pre-condizioni : L'operatore non esiste nel sistema

Post-condizioni : L'operatore esiste nel sistema (a meno che si sia verificata l'impossibilità di aggiungerlo)

Sequenza degli eventi principali :

- 1) Il caso d'uso inizia quando l'amministratore deve inserire i dati di un nuovo operatore all'interno del sistema
- 2) Il sistema visualizza la schermata di inserimento dati del nuovo operatore
- 3) L'amministratore fornisce le informazioni richieste
- 4) L'amministratore avvia il processo di inserimento del nuovo operatore
- 5) Il sistema aggiunge il nuovo operatore

Sequenza alternativa : La sequenza alternativa inizia dal punto 4:

- 5) Le informazioni inserite dall'amministratore incomplete
- 6) L'operazione di inserimento del nuovo operatore fallisce

Caso d'Uso AMMINISTRATORE DELL'UFFICIO DI ACCETTAZIONE: Eliminazione Operatore

Questo caso d'uso si ha quando l'amministratore deve eliminare un operatore dal sistema.

Pre-condizioni : L'operatore esiste nel sistema.

Post-condizioni : L'operatore non esiste più nel sistema .

Sequenza degli eventi principali :

- 1) Il caso d'uso inizia quando l'amministratore dell'accettazione deve eliminare le informazioni relative ad un operatore, cioè un infermiere o un medico
- 2) Il sistema legge i dati dell'operatore scelto dall'amministratore
- 3) Il sistema visualizza a schermo i dati dell'operatore
- 4) L'amministratore avvia il processo di eliminazione
- 5) Il sistema elimina le informazioni dell'operatore

Sequenza alternativa : Nessuna

Caso d'Uso AMMINISTRATORE DELL'UFFICIO DI ACCETTAZIONE: Modifica Operatore

Questo caso d'uso si ha quando l'amministratore deve modificare i dati anagrafici di un operatore presente nel sistema.

Pre-condizioni : L'operatore esiste nel sistema.

Post-condizioni : L'operatore esiste a sistema ed i suoi dati risultano essere modificati.

Sequenza degli eventi principali :

- 1) Il caso d'uso inizia quando l'amministratore dell'accettazione deve modificare le informazioni relative ad un operatore
- 2) Il sistema legge i dati dell'operatore scelto dall'amministratore

- 3) Il sistema visualizza a schermo i dati dell'operatore
- 4) L'amministratore avvia il processo di modifica
- 5) Il sistema modifica le informazioni dell'operatore

Sequenza alternativa : Nessuna

Caso d'Uso AMMINISTRATORE DELL'UFFICIO DI ACCETTAZIONE: Ricerca Operatore

Questo caso d'uso si verifica quando un amministratore vuole cercare un determinato operatore all'interno della lista degli impiegati attuali nella struttura.

Pre-condizioni : L'operatore esiste nel sistema.

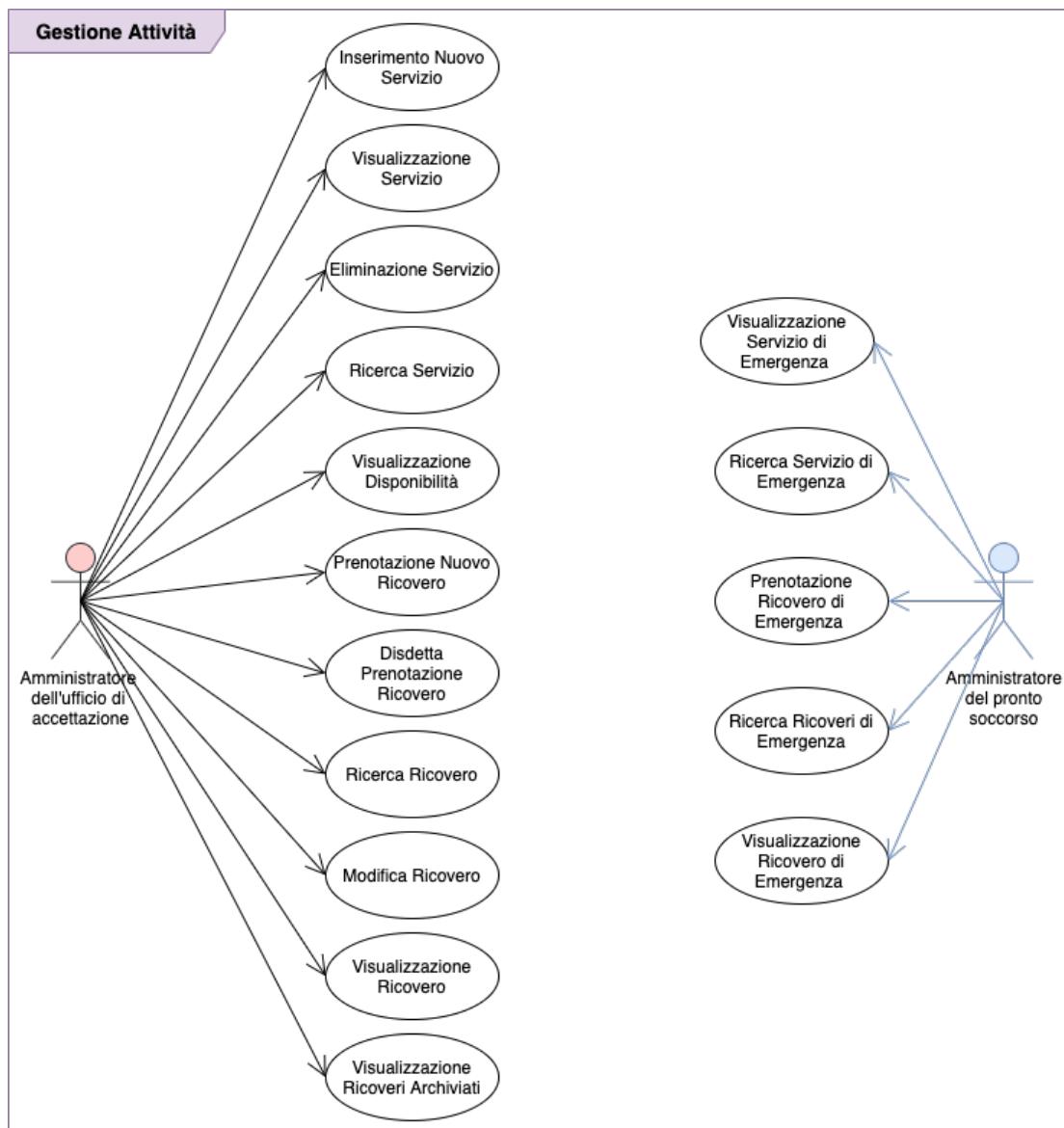
Post-condizioni : Nessuna.

Sequenza degli eventi principali :

- 1) Il caso d'uso inizia quando un amministratore vuole cercare un determinato operatore
- 2) Il sistema legge i dati relativi all'operatore inseriti dall'amministratore (nome, cognome, codice fiscale)
- 3) Il sistema seleziona tutti gli operatori presenti a sistema con le caratteristiche inserite dall'amministratore
- 4) L'amministratore può visualizzare tutte le informazioni raccolte dal sistema

Sequenza degli eventi alternativa : Nessuna

CASI D'USO: Gestione Attività



Caso d'Uso AMMINISTRATORE DELL'UFFICIO DI ACCETTAZIONE: Inserimento Nuovo Servizio

Questo caso d'uso si verifica quando l'amministratore dell'ufficio di accettazione deve inserire a sistema un nuovo servizi (ricovero o ricovero di emergenza).

Pre-condizioni : Il servizio non esiste a sistema.

Post-condizioni : Il nuovo servizio esiste a sistema.

Sequenza degli eventi principali:

- 1) Il caso d'uso inizia quando l'amministratore deve inserire i dati di un nuovo servizio per la struttura ospedaliera

- 2) Il sistema visualizza la schermata di inserimento dati del nuovo servizio
- 3) L'amministratore fornisce le informazioni richieste
- 4) L'amministratore avvia il processo di inserimento del nuovo servizio
- 5) Il sistema aggiunge il nuovo servizio

Sequenza alternativa: la sequenza alternativa inizia dal punto 4:

- 5) Le informazioni inserite dall'amministratore non sono sufficienti
- 6) L'operazione di inserimento del nuovo servizio fallisce

Caso d'Uso AMMINISTRATORE DELL'UFFICIO DI ACCETTAZIONE: Visualizzazione Servizio

Questo caso d'uso si verifica quando l'amministratore dell'ufficio di accettazione vuole visualizzare le informazioni relative ad un servizio presente nella struttura ospedaliera.

Pre-condizioni: Il servizio esiste nel sistema.

Post-condizioni: Nessuna.

Sequenza degli eventi principali:

- 1) Il caso d'uso inizia quando l'amministratore dell'ufficio di accettazione vuole visualizzare le informazioni relative ad un servizio della struttura
- 2) Il sistema legge le informazioni del servizio selezionato dall'amministratore
- 3) Il sistema visualizza a schermo le informazioni del servizio

Sequenza alternativa: Nessuna

Caso d'Uso AMMINISTRATORE DEL PRONTO SOCCORSO: Visualizzazione Servizio di Emergenza

Questo caso d'uso si verifica quando l'amministratore del pronto soccorso vuole visualizzare le informazioni relative ad un servizio di emergenza presente nella struttura ospedaliera.

Pre-condizioni: Il servizio esiste nel sistema.

Post-condizioni: Nessuna.

Sequenza degli eventi principali:

- 1) Il caso d'uso inizia quando l'amministratore del pronto soccorso vuole visualizzare le informazioni relative ad un servizio di emergenza della struttura
- 2) Il sistema legge le informazioni del servizio selezionato dall'amministratore
- 3) Il sistema visualizza a schermo le informazioni del servizio

Sequenza alternativa: Nessuna

Caso d'Uso AMMINISTRATORE DELL'UFFICIO DI ACCETTAZIONE: Eliminazione Servizio

Questo caso d'uso si ha quando l'amministratore deve eliminare un servizio dal sistema, perché non più presente in struttura.

Pre-condizioni : Il servizio esiste nel sistema.

Post-condizioni : Il servizio non esiste più nel sistema .

Sequenza degli eventi principali :

- 1) Il caso d'uso inizia quando l'amministratore dell'accettazione vuole eliminare un servizio non più disponibile in struttura
- 2) Il sistema legge i dati del servizio scelto dall'amministratore
- 3) Il sistema visualizza a schermo i dati del servizio
- 4) L'amministratore avvia il processo di eliminazione
- 5) Il sistema elimina le informazioni del servizio

Sequenza alternativa : Nessuna

Caso d'Uso AMMINISTRATORE DELL'UFFICIO DI ACCETTAZIONE: Ricerca Servizio

Questo caso d'uso si verifica quando un amministratore vuole cercare un determinato servizio presente in struttura per visualizzarne le informazioni.

Pre-condizioni : Il servizio esiste nel sistema.

Post-condizioni : Nessuna.

Sequenza degli eventi principali :

- 1) Il caso d'uso inizia quando un amministratore vuole cercare un determinato servizio
- 2) Il sistema legge i dati relativi al servizio inseriti dall'amministratore (reparto e/o posto letto)
- 3) Il sistema seleziona tutti i servizi presenti a sistema con le caratteristiche inserite dall'amministratore
- 4) L'amministratore può visualizzare tutte le informazioni raccolte dal sistema

Sequenza degli eventi alternativa : Nessuna

Caso d'Uso AMMINISTRATORE DEL PRONTO SOCCORSO: Ricerca Servizio di Emergenza

Questo caso d'uso si verifica quando l'amministratore del pronto soccorso vuole cercare un determinato servizio all'interno della lista dei servizi di emergenza presenti nella struttura ospedaliera.

Pre-condizioni : Il servizio esiste nel sistema.

Post-condizioni : Nessuna.

Sequenza degli eventi principali :

- 1) Il caso d'uso inizia quando un amministratore vuole cercare un determinato servizio di emergenza
- 2) Il sistema legge i dati relativi al servizio inseriti dall'amministratore (reparto e/o posto letto)

- 3) Il sistema seleziona tutti i servizi presenti a sistema con le caratteristiche inserite dall'amministratore
- 4) L'amministratore può visualizzare tutte le informazioni raccolte dal sistema

Sequenza degli eventi alternativa : Nessuna

Caso d'Uso AMMINISTRATORE DELL'UFFICIO DI ACCETTAZIONE: Visualizzazione Disponibilità

Questo caso d'uso si ha quando l' amministratore dell'ufficio di accettazione vuole verificare la disponibilità dei posti letto all'interno dei vari reparti dell'ospedale

Pre-condizioni e post-condizioni: Nessuna

Sequenza degli eventi principali:

- 1) Il caso d'uso inizia quando l'amministratore vuole visualizzare la disponibilità dei posti letto all'interno di un reparto
- 2) Il sistema legge il numero di posti letto occupati e disponibili nel reparto desiderato
- 3) Il sistema comunica all'amministratore la situazione dei servizi nei vari reparti dell'ospedale.

Sequenza alternativa: Nessuna

Caso d'Uso AMMINISTRATORE DELL'UFFICIO DI ACCETTAZIONE: Prenotazione Nuovo Ricovero

Questo caso d'uso si verifica quando l'amministratore dell'ufficio di accettazione deve inserire la prenotazione di un ricovero.

Pre-condizioni : Il paziente non è ricoverato all'interno della struttura

Post-condizioni : Il paziente viene ricoverato all'interno della struttura (a meno che non si verifichi qualche problema, come la mancata disponibilità di posti letto all'interno del reparto)

Sequenza degli eventi principali:

- 6) Il caso d'uso inizia quando l'amministratore deve inserire la prenotazione di un nuovo ricovero da parte di un paziente
- 7) Il sistema ricerca le informazioni del paziente (se quest'ultimo non è presente nel sistema, viene inserito dall'amministratore in quanto nuovo paziente)
- 8) Si verifica la disponibilità dei posti letto
- 9) Si procede con la prenotazione del ricovero, indicando il periodo e il reparto desiderato

Sequenza alternativa: La sequenza alternativa inizia dal punto 3 :

- 3) Non vi è disponibilità di posti letto al momento

Caso d'Uso AMMINISTRATORE DEL PRONTO SOCCORSO: Prenotazione Ricovero di Emergenza

Questo caso d'uso si verifica quando si presenta al pronto soccorso un'emergenza, dunque l'amministratore del pronto soccorso vuole inserire una nuova prenotazione.

Pre-condizioni: la presenza di un caso d'emergenza.

Post-condizione: al paziente viene assegnato un ricovero di emergenza ed un posto letto di emergenza.

Sequenza degli eventi principali:

- 1) Il caso d'uso inizia quando al pronto soccorso si presenta un'urgenza che necessita di ricovero
- 2) Il sistema ricerca le informazioni del paziente (se quest'ultimo non è presente nel sistema, viene inserito dall'amministratore in quanto nuovo paziente)
- 3) Si verifica la disponibilità dei posti letto
- 4) Si procede con la prenotazione del ricovero, indicando il periodo e il reparto desiderato

Sequenza alternativa: La sequenza alternativa inizia dal punto 3:

- 4) Non vi è disponibilità di posti letto al momento

Caso d'Uso AMMINISTRATORE DELL'UFFICIO DI ACCETTAZIONE: Visualizzazione Ricovero

Questo caso d'uso si verifica quando l'amministratore dell'ufficio di accettazione vuole visualizzare le informazioni relative ad un ricovero.

Pre-condizioni: Il ricovero esiste nel sistema

Post-condizioni: Nessuna

Sequenza degli eventi principali:

- 1) Il caso d'uso inizia quando l'amministratore dell'ufficio di accettazione vuole visualizzare le informazioni relative ad un ricovero
- 2) Il sistema legge le informazioni del ricovero scelto dall'amministratore
- 3) Il sistema visualizza a schermo le informazioni del ricovero (paziente, servizio, data di inizio e di fine)

Sequenza alternativa: Nessuna

Caso d'Uso AMMINISTRATORE DELL'UFFICIO DI ACCETTAZIONE: Modifica Ricovero

Questo caso d'uso si ha quando l'amministratore deve modificare i dati di un ricovero.

Pre-condizioni : Il ricovero esiste nel sistema.

Post-condizioni : Il ricovero esiste a sistema ed i suoi dati risultano essere modificati.

Sequenza degli eventi principali :

- 1) Il caso d'uso inizia quando l'amministratore dell'accettazione deve modificare le informazioni relative ad un ricovero
- 2) Il sistema legge i dati del ricovero scelto dall'amministratore
- 3) Il sistema visualizza a schermo i dati del ricovero

- 4) L'amministratore avvia il processo di modifica
- 5) Il sistema modifica le informazioni del ricovero

Sequenza alternativa : Nessuna

Caso d'Uso AMMINISTRATORE DELL'UFFICIO DI ACCETTAZIONE: Disdetta Prenotazione Ricovero

Questo caso d'uso si verifica qualora l'amministratore dell'ufficio di accettazione debba disdire la prenotazione di un ricovero, sotto richiesta del paziente

Pre-condizioni : La prenotazione del ricovero esiste nel sistema

Post-condizioni : La prenotazione del ricovero non esiste più nel sistema

Sequenza degli eventi principali :

- 1) Il caso d'uso inizia quando l'amministratore deve eliminare dal sistema una prenotazione di un ricovero, richiesta da un paziente
- 2) Il sistema ricerca la prenotazione scelta dall'amministratore
- 3) Il sistema visualizza a schermo le informazioni relative alla prenotazione del ricovero
- 4) L'amministratore avvia la procedura di eliminazione
- 5) Il sistema elimina la prenotazione del ricovero

Sequenza alternativa : Nessuna

Caso d'Uso AMMINISTRATORE DELL'UFFICIO DI ACCETTAZIONE: Ricerca Ricovero

Questo caso d'uso si verifica quando un amministratore vuole cercare un determinato ricovero all'interno della lista dei ricoveri attuali o prenotati.

Pre-condizioni : Il ricovero esiste nel sistema.

Post-condizioni : Nessuna.

Sequenza degli eventi principali :

- 1) Il caso d'uso inizia quando un amministratore vuole cercare un determinato ricovero
- 2) Il sistema legge i dati relativi al ricovero inseriti dall'amministratore
- 3) Il sistema seleziona tutti i ricoveri presenti a sistema con le caratteristiche inserite dall'amministratore
- 4) L'amministratore può visualizzare tutte le informazioni raccolte dal sistema

Sequenza degli eventi alternativa : Nessuna

Caso d'Uso AMMINISTRATORE DEL PRONTO SOCCORSO: Visualizzazione Ricovero di Emergenza

Questo caso d'uso si verifica quando l'amministratore del pronto soccorso vuole visualizzare le informazioni relative ad un ricovero.

Pre-condizioni: Il ricovero esiste nel sistema

Post-condizioni: Nessuna

Sequenza degli eventi principali:

- 1) Il caso d'uso inizia quando l'amministratore del pronto soccorso vuole visualizzare le informazioni relative ad un ricovero
- 2) Il sistema legge le informazioni del ricovero scelto dall'amministratore
- 3) Il sistema visualizza a schermo le informazioni del ricovero (paziente, servizio, data di inizio e di fine)

Sequenza alternativa: Nessuna

Caso AMMINISTRATORE DEL PRONTO SOCCORSO: Ricerca Ricovero di Emergenza

Questo caso d'uso si verifica quando l'amministratore del pronto soccorso vuole cercare un determinato ricovero all'interno della lista dei ricoveri attuali o prenotati.

Pre-condizioni : Il ricovero esiste nel sistema.

Post-condizioni : Nessuna.

Sequenza degli eventi principali :

- 1) Il caso d'uso inizia quando l'amministratore vuole cercare un determinato ricovero
- 2) Il sistema legge i dati relativi al ricovero inseriti dall'amministratore
- 3) Il sistema seleziona tutti i ricoveri presenti a sistema con le caratteristiche inserite dall'amministratore
- 4) L'amministratore può visualizzare tutte le informazioni raccolte dal sistema

Sequenza degli eventi alternativa : Nessuna

Caso d'Uso AMMINISTRATORE DELL'UFFICIO DI ACCETTAZIONE: Visualizzazione Ricoveri Archiviati

Questo caso d'uso si verifica quando l'amministratore dell'ufficio di accettazione vuole visualizzare le informazioni relative ad un ricovero passato.

Pre-condizioni: Il ricovero esiste nel sistema

Post-condizioni: Nessuna

Sequenza degli eventi principali:

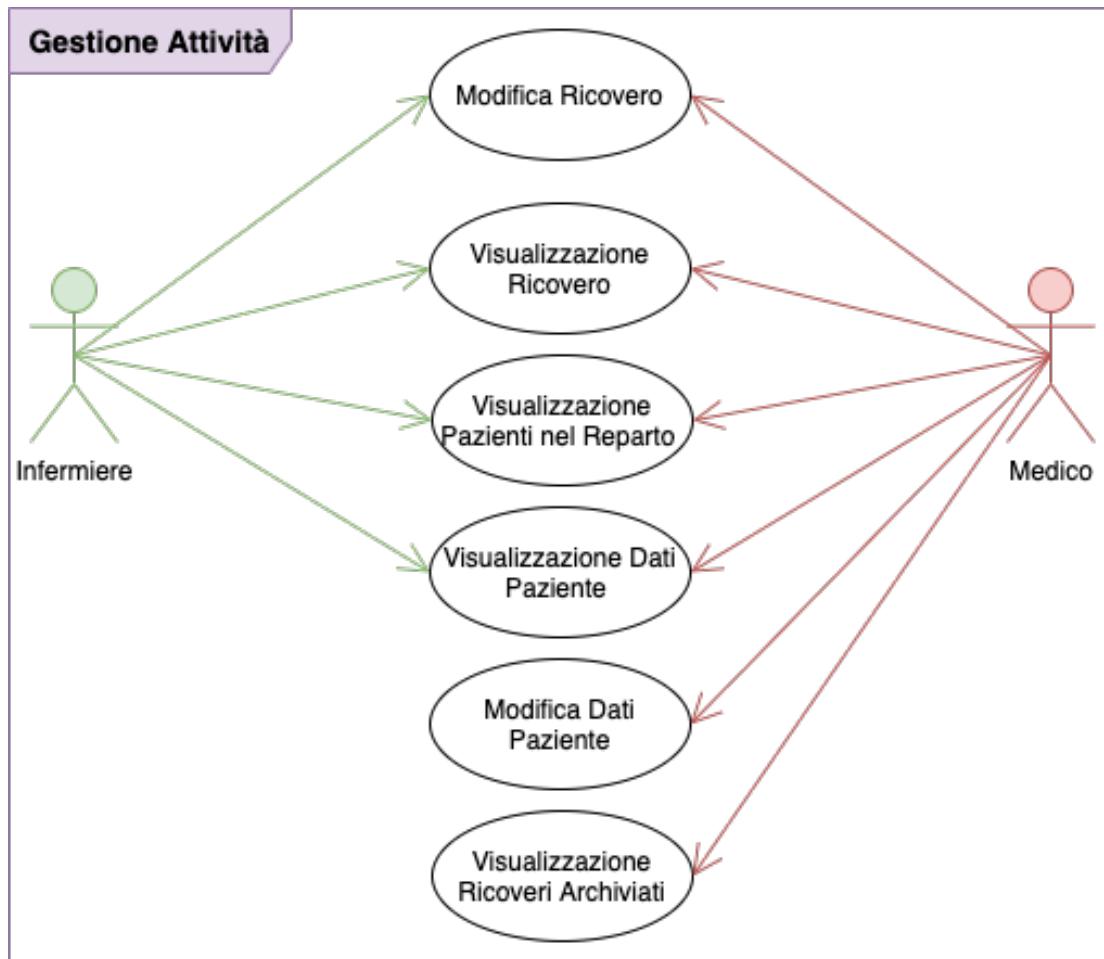
- 1) Il caso d'uso inizia quando l'amministratore dell'ufficio di accettazione vuole visualizzare le informazioni relative ad un ricovero passato
- 2) Il sistema legge le informazioni del ricovero scelto dall'amministratore
- 3) Il sistema visualizza a schermo le informazioni del ricovero (paziente, servizio, data di inizio e di fine)

Sequenza alternativa: Nessuna

DESCRIZIONE DEI CASI D'USO

ATTORI: INFERMIERE – MEDICO

CASI D'USO: Gestione Attività



Caso d'Uso INFERMIERE – MEDICO : Visualizzazione Ricovero

Questo caso d'uso si verifica quando l'infermiere o il medico vogliono visualizzare le informazioni relative ad un ricovero.

Pre-condizioni: Il ricovero esiste nel sistema

Post-condizioni: Nessuna

Sequenza degli eventi principali:

- 1) Il caso d'uso inizia quando l'operatore vuole visualizzare le informazioni relative ad un ricovero
- 2) Il sistema legge le informazioni del ricovero scelto dall'operatore

- 3) Il sistema visualizza a schermo le informazioni del ricovero (paziente, servizio, data di inizio e di fine)

Sequenza alternativa: Nessuna

Caso d'Uso INFERMIERE – MEDICO: Modifica Ricovero

Questo caso d'uso si ha quando un infermiere o un medico deve modificare i dati di un ricovero.

Pre-condizioni : Il ricovero esiste nel sistema.

Post-condizioni : Il ricovero esiste a sistema ed i suoi dati risultano essere modificati.

Sequenza degli eventi principali :

- 1) Il caso d'uso inizia quando l'operatore deve modificare le informazioni relative ad un ricovero
- 2) Il sistema legge i dati del ricovero scelto dall'amministratore
- 3) Il sistema visualizza a schermo i dati del ricovero
- 4) L'operatore avvia il processo di modifica
- 5) Il sistema modifica le informazioni del ricovero

Sequenza alternativa : Nessuna

Caso d'Uso INFERMIERE – MEDICO: Visualizzazione Pazienti nel Reparto

Questo caso d'uso si verifica quando l'infermiere o il medico vuole visualizzare tutti i pazienti presenti in un reparto.

Pre-condizioni : I pazienti esistono nel sistema

Post-condizioni : Nessuna

Sequenza degli eventi principali :

- 1) Il caso d'uso inizia quando un infermiere o un medico vuole visualizzare i pazienti ricoverati in un determinato reparto
- 2) Il sistema legge le informazioni del reparto scelto

- 3) Il sistema visualizza a schermo tutte le informazioni del reparto scelto

Sequenza degli eventi alternativa : Nessuna

Caso d'Uso INFERMIERE - MEDICO: Visualizzazione Dati Paziente

Questo caso d'uso si verifica quando il medico o l'infermiere vuole visualizzare i dati anagrafici e clinici di un determinato paziente.

Pre-condizioni : Il paziente esiste nel sistema

Post-condizioni : Nessuna

Sequenza degli eventi principali :

- 1) Il caso d'uso inizia quando un medico vuole visualizzare i dati di un paziente ricoverato nella struttura
- 2) Il sistema legge le informazioni del paziente selezionato
- 3) Il sistema visualizza a schermo tutte le informazioni del paziente scelto

Sequenza degli eventi alternativa : Nessuna

Caso d'Uso MEDICO: Modifica Dati Paziente

Questo caso d'uso si verifica quando un medico vuole modificare il ricovero di un determinato paziente

Pre-condizioni : il paziente è ricoverato nella struttura.

Post-condizioni : vengono aggiornati i dati di un paziente.

Sequenza degli eventi principali :

- 1) Il caso d'uso inizia quando un medico vuole modificare i dati clinici e/o anagrafici di un paziente
- 2) Il sistema modificherà i dati
- 3) Il sistema visualizza a schermo che l'operazione è andata a buon fine

Sequenza degli eventi alternativa : Nessuna

Caso d'Uso MEDICO: Visualizzazione Ricoveri Archiviati

Questo caso d'uso si verifica quando il medico vuole visualizzare le informazioni relative ad un ricovero passato.

Pre-condizioni: Il ricovero esiste nel sistema

Post-condizioni: Nessuna

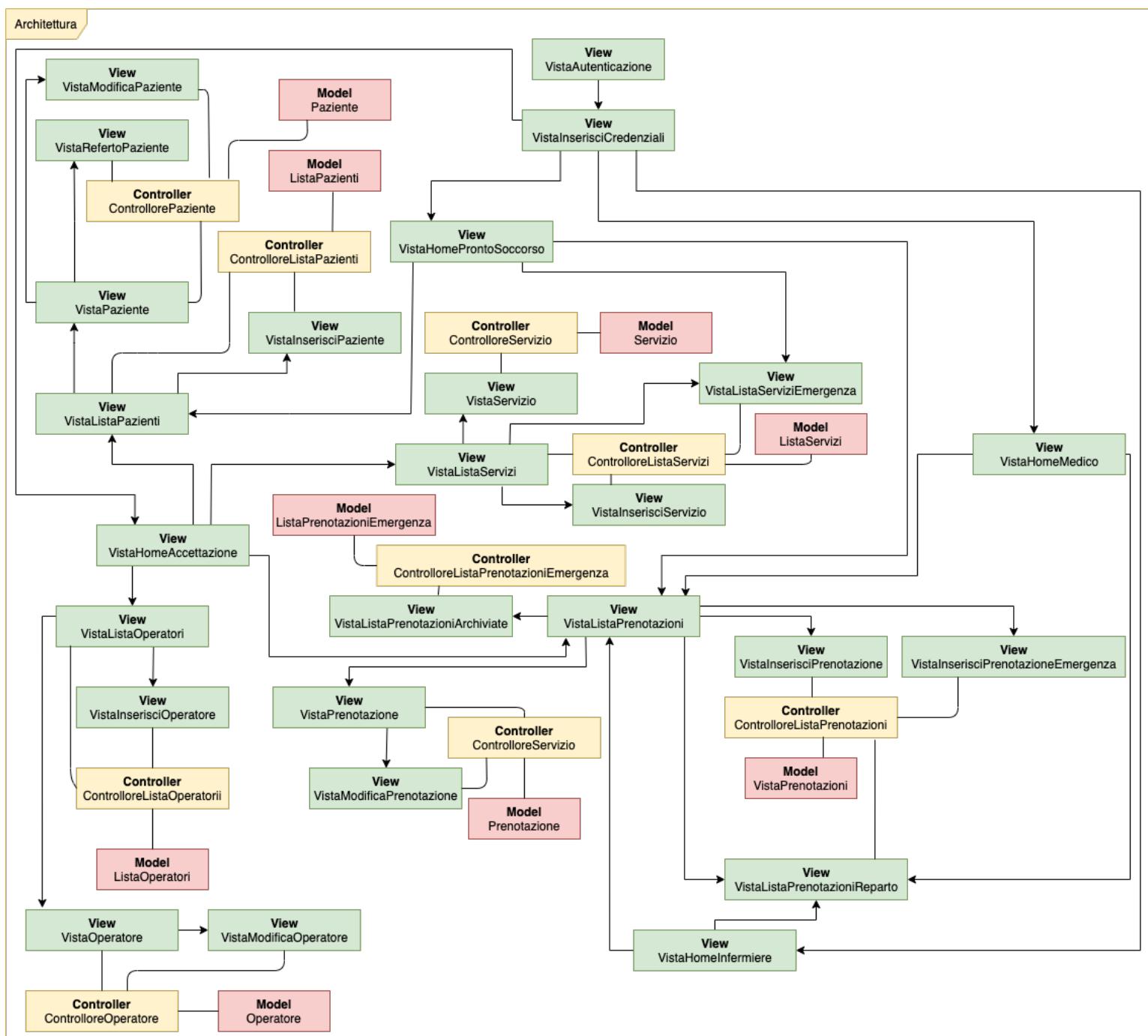
Sequenza degli eventi principali:

- 1) Il caso d'uso inizia quando il medico vuole visualizzare le informazioni relative ad un ricovero passato
- 2) Il sistema legge le informazioni del ricovero scelto dal medico
- 3) Il sistema visualizza a schermo le informazioni del ricovero (paziente, servizio, data di inizio e di fine)

Sequenza alternativa: Nessuna

MAPPA DELL'ARCHITETTURA

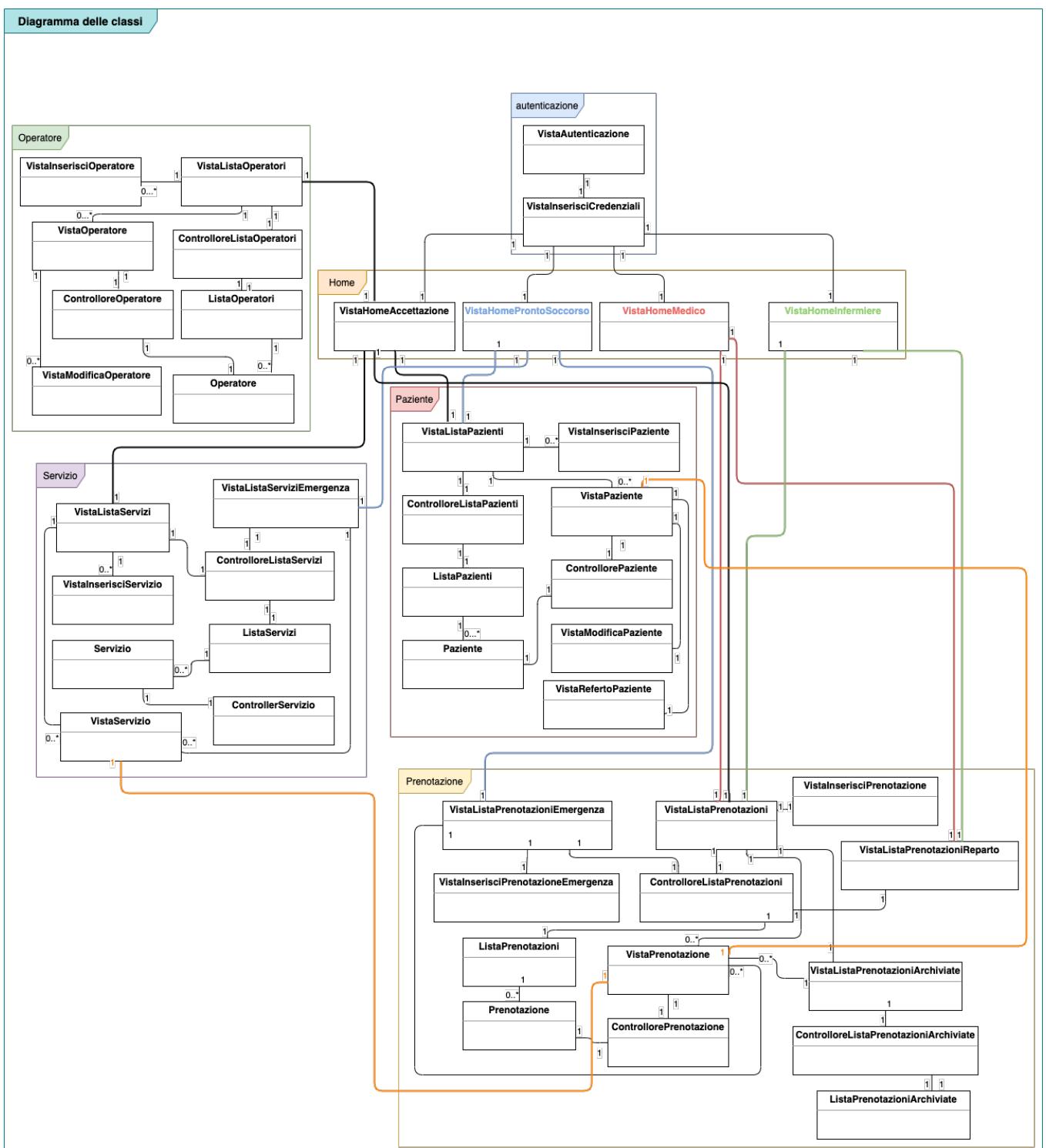
Per lo sviluppo del software è stata utilizzata un'architettura MVC (Model-View-Controller).



DIAGRAMMI DELLE CLASSI

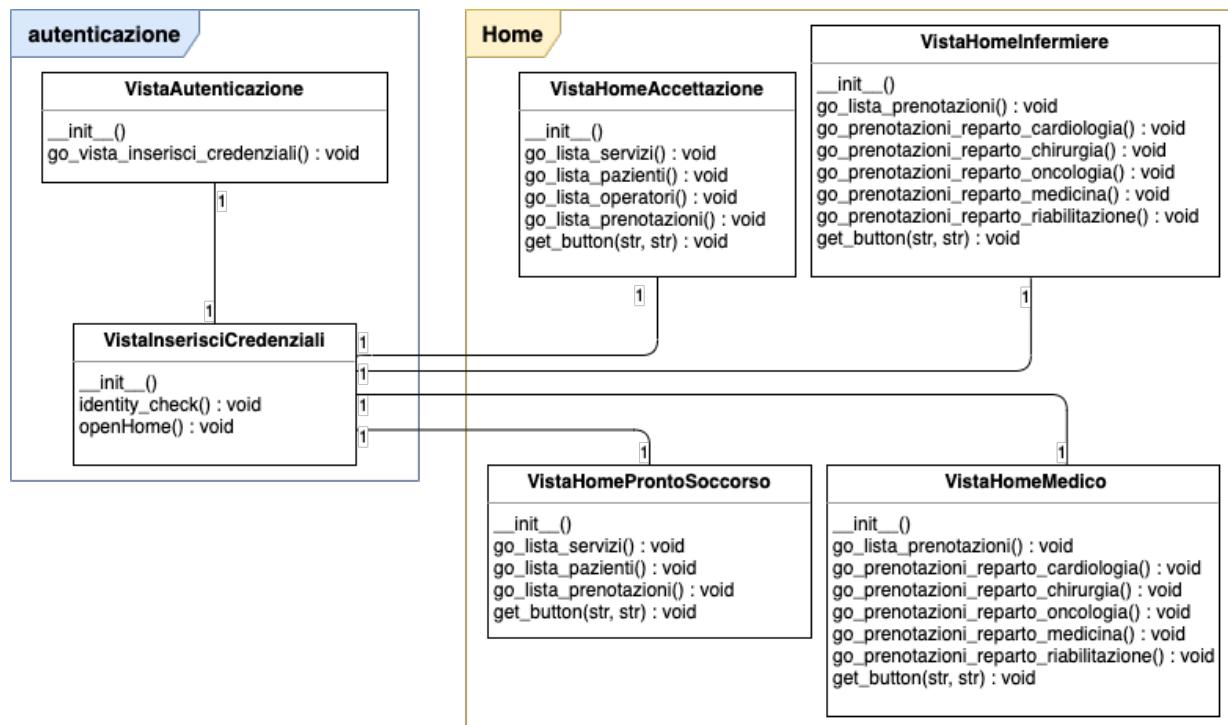
DIAGRAMMA COMPLESSIVO DELLE CLASSI

Questo diagramma ha lo scopo di illustrare le varie relazioni tra le classi che compongono il nostro software. Di seguito verranno illustrati i diagrammi delle classi in maniera dettagliata.

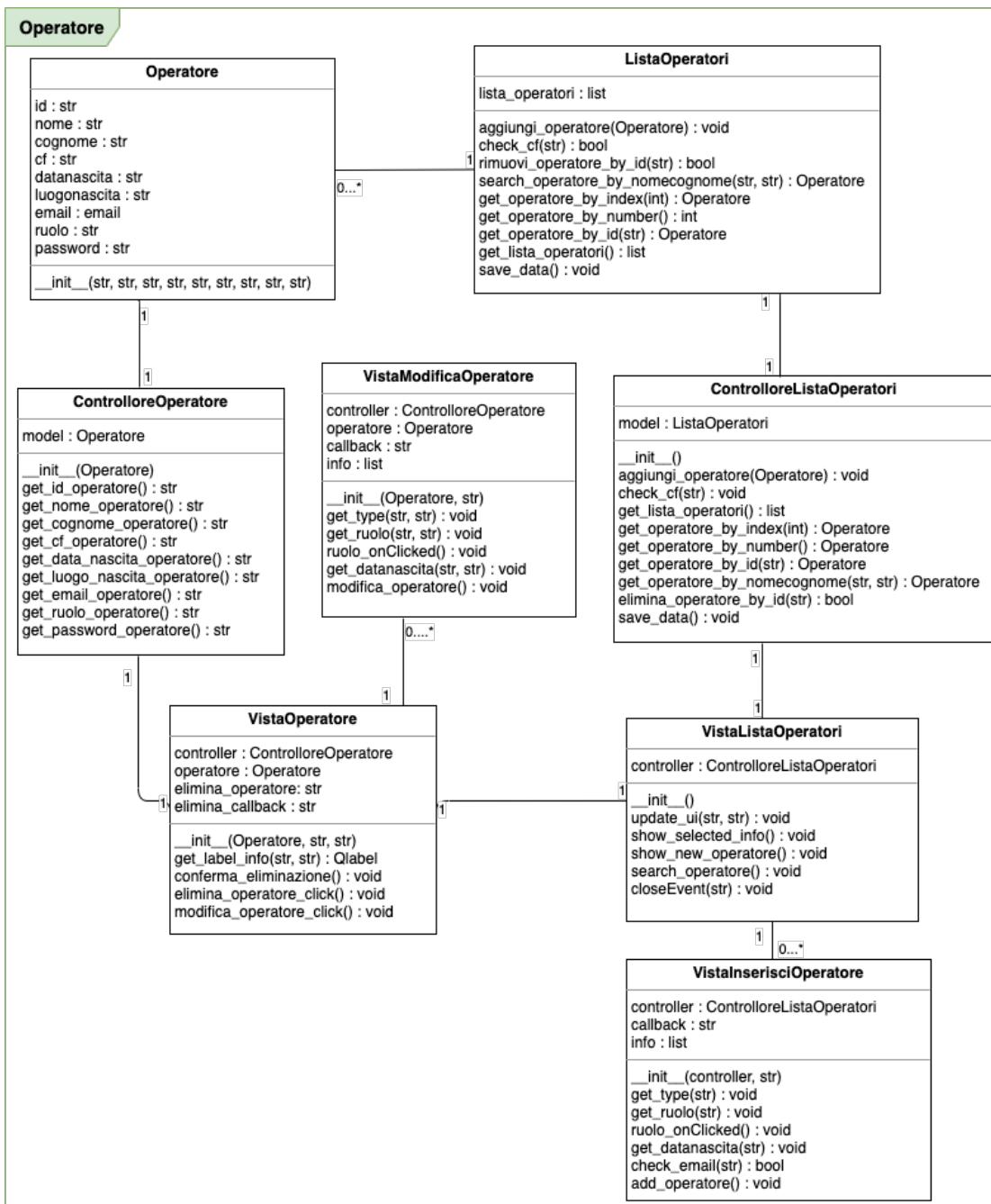


AUTENTICAZIONE - HOME

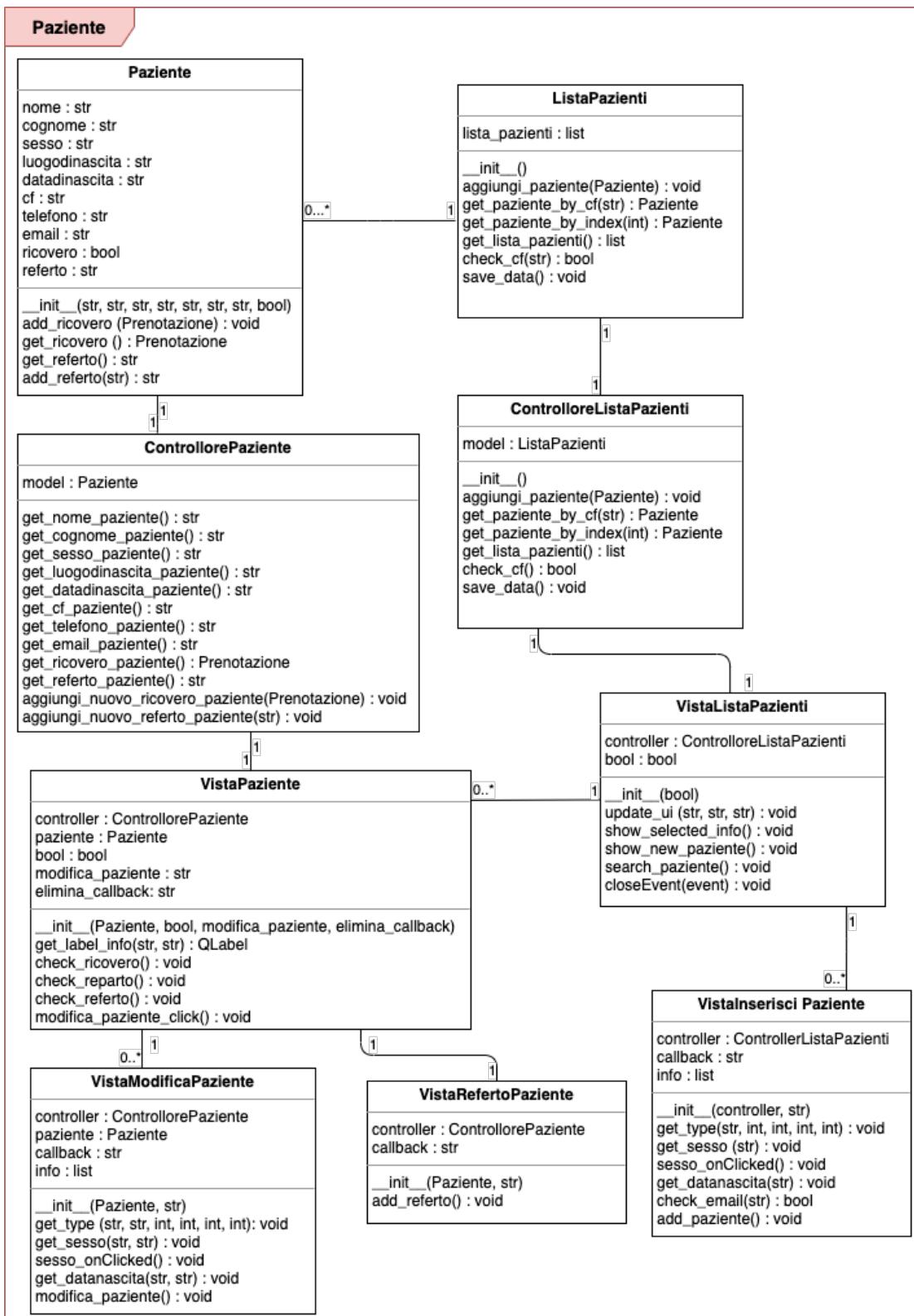
In questo diagramma sono rappresentate le classi che riguardano l'autenticazione dell'utente.



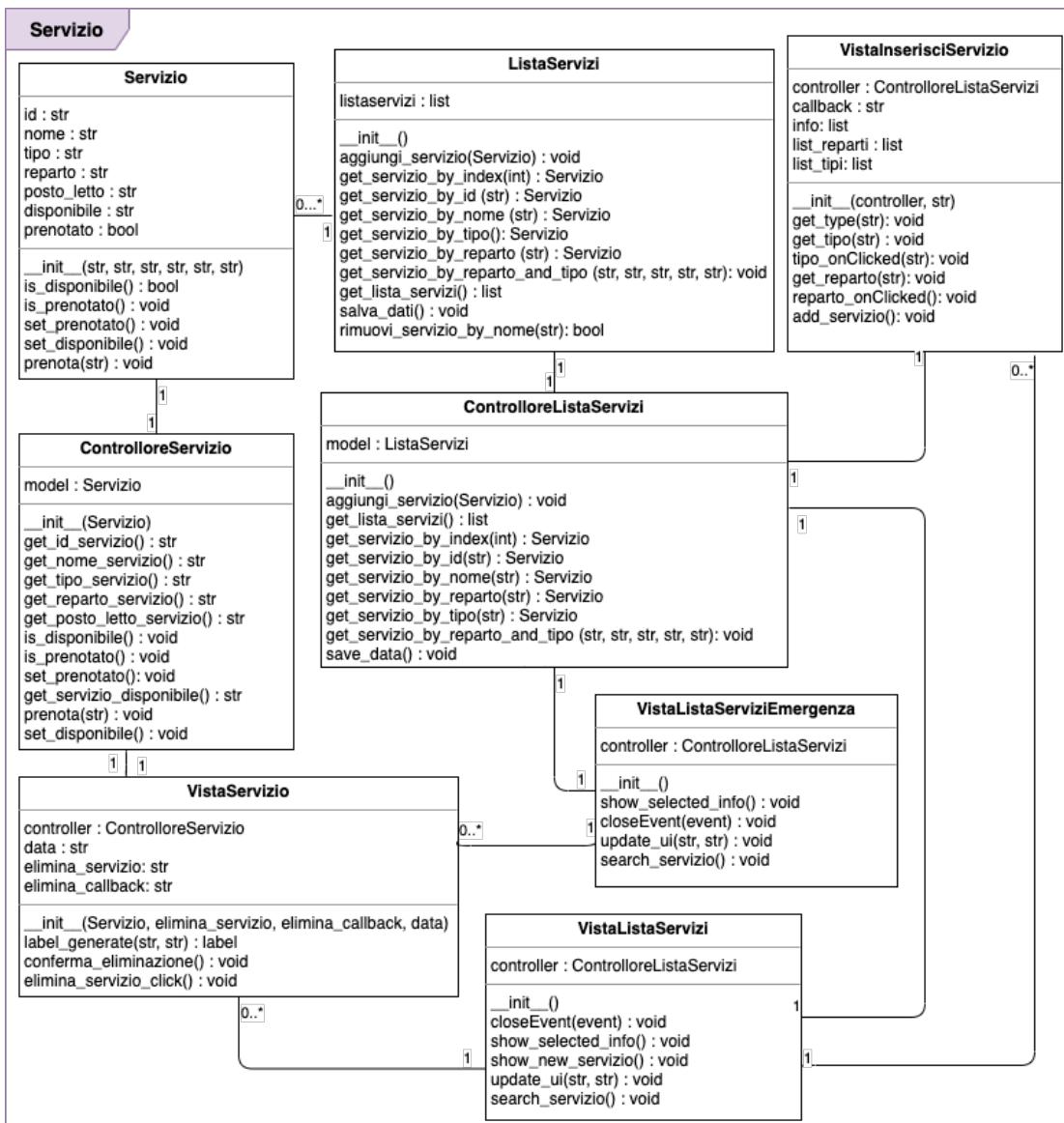
OPERATORE



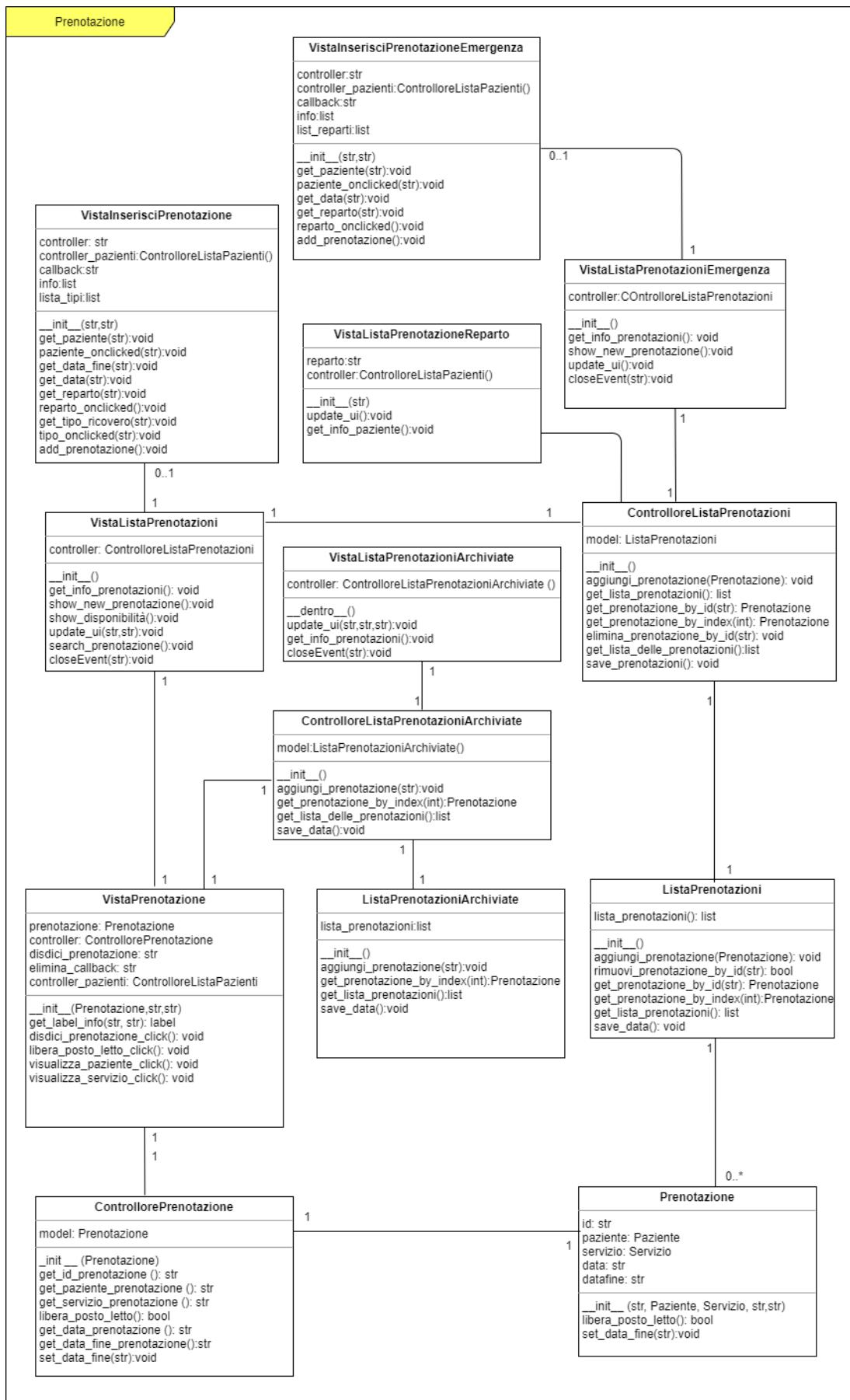
PAZIENTE



SERVIZIO



PRENOTAZIONE

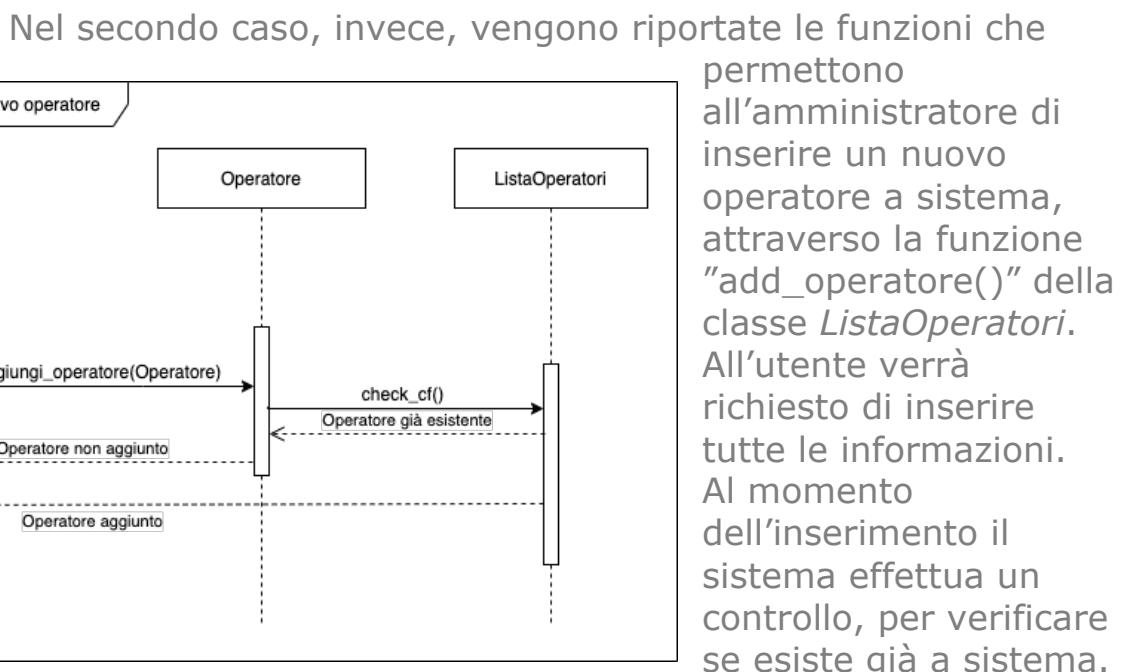
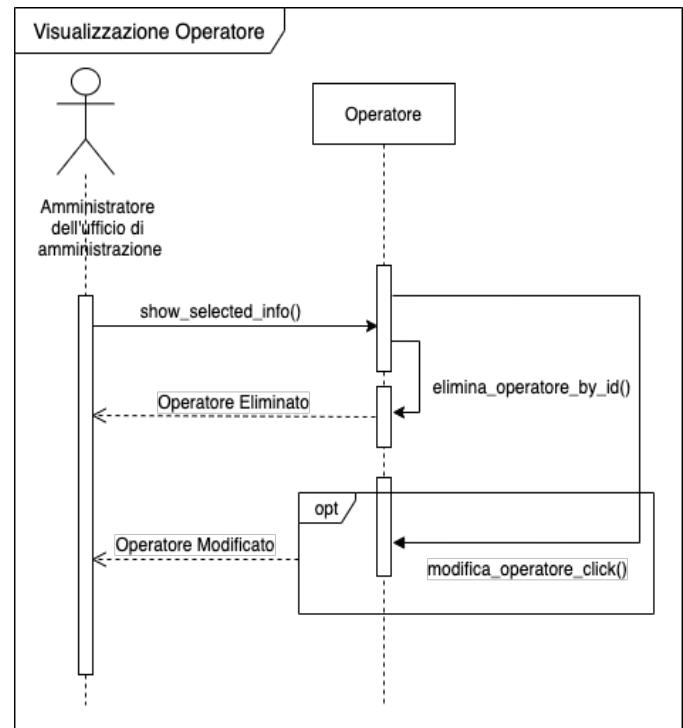


DIAGRAMMI DELLE SEQUENZE:

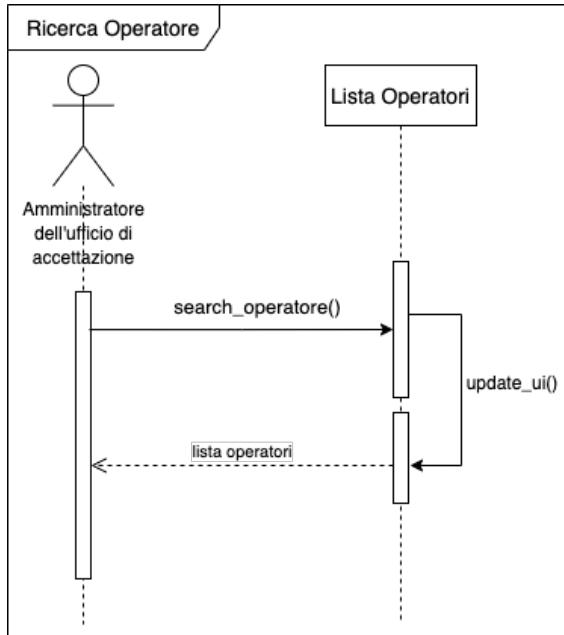
OPERATORI

I seguenti diagrammi delle sequenze illustrano le interazioni tra l'amministratore dell'accettazione e le varie parti del sistema che gestiscono la lista degli operatori.

Nel primo caso vengono riportate le funzioni che permettono la visualizzazione di un operatore, attraverso un'interfaccia grafica. Si può accedere a questa lista anche nel caso in cui si presenti la necessità di modificare i dati di un operatore o di eliminarlo dal sistema. L'utente accede alla lista degli operatori, seleziona uno di essi e attraverso la funzione "show_selected_info()" ne visualizza le informazioni. In questo modo è in grado anche di modificarne i dati e eventualmente di eliminarlo dal sistema.



Nel secondo caso, invece, vengono riportate le funzioni che permettono all'amministratore di inserire un nuovo operatore a sistema, attraverso la funzione "add_operatore()" della classe *ListaOperatori*. All'utente verrà richiesto di inserire tutte le informazioni. Al momento dell'inserimento il sistema effettua un controllo, per verificare se esiste già a sistema.

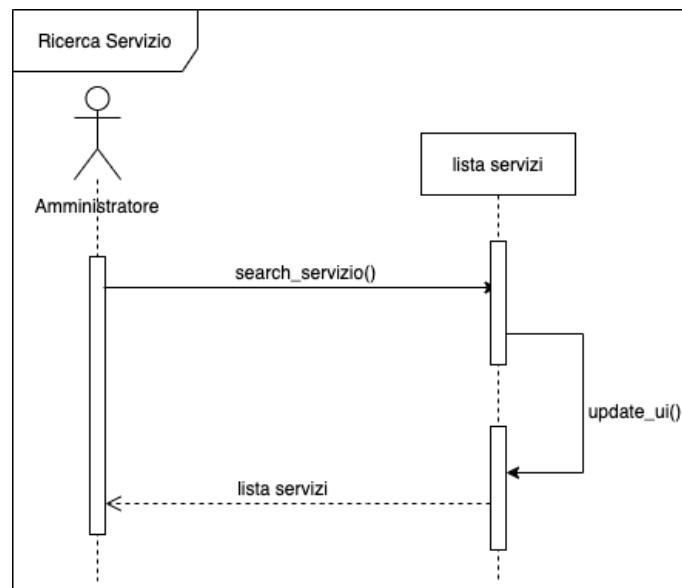


Nel terzo caso vengono riportate le funzioni che servono per ricercare un operatore, attraverso il nome e il cognome di quest'ultimo, (anche parzialmente). Viene utilizzata per fare ciò la funzione “search_operatore()”.

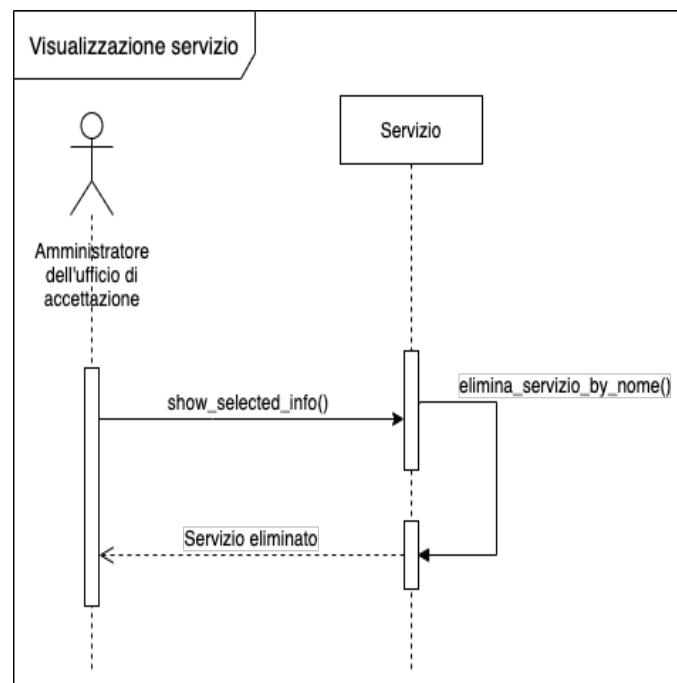
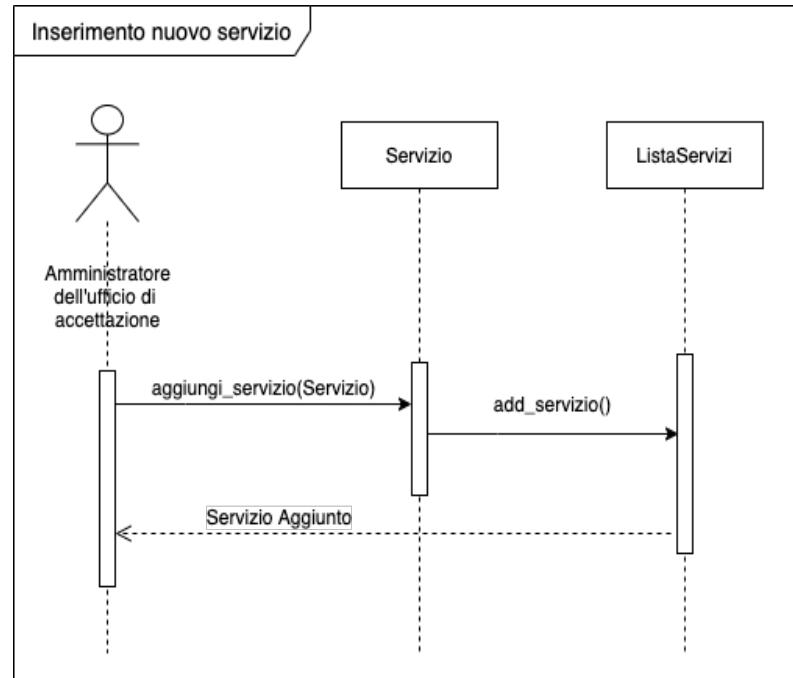
SERVIZI

I seguenti diagrammi delle sequenze illustrano le interazioni tra l'amministratore dell'accettazione e le varie parti del sistema che gestiscono la lista dei servizi.

Nel primo caso sono riportate le funzioni che permettono la ricerca di un servizio, tramite il reparto o il posto letto associato, attraverso la funzione “search_servizio()”.



Nel secondo caso d'uso sono invece riportati i metodi che permettono di inserire le informazioni relative ad un nuovo servizio. Bisogna perciò specificare il tipo di reparto e il posto letto. Fatto ciò, il servizio viene aggiunto, attraverso il metodo "add_servizio()", alla lista dei servizi.

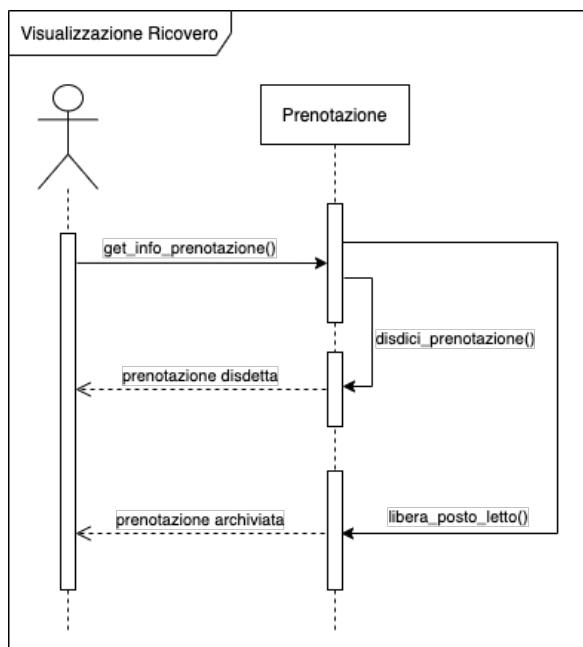
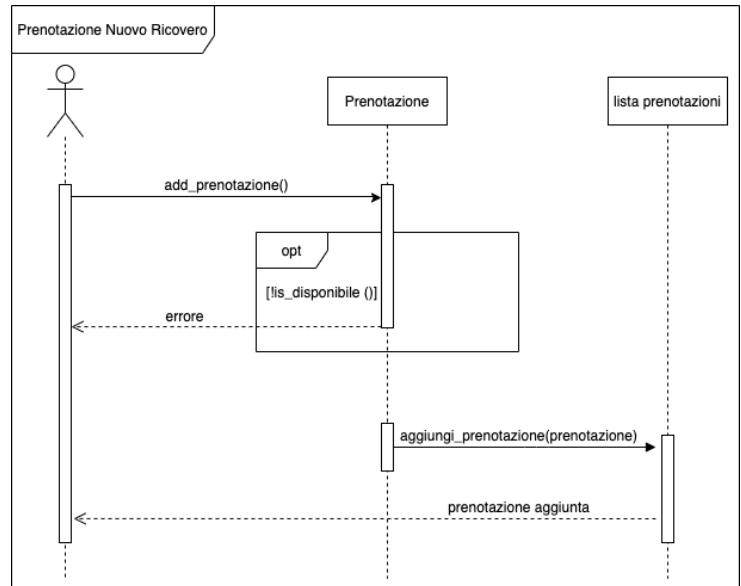


Nel terzo diagramma possiamo osservare le funzioni che permettono la visualizzazione di un servizio, attraverso la funzione "show_selected_info()" e se necessaria, la conseguente eliminazione di esso attraverso il suo nome univoco.

RICOVERI:

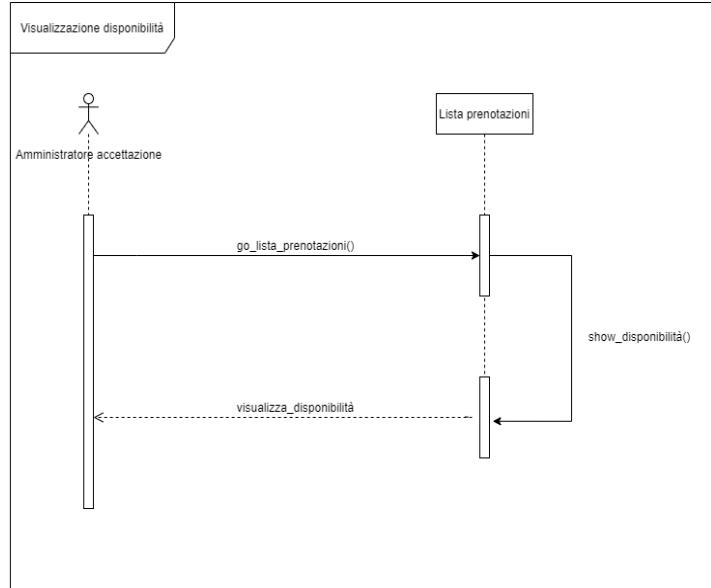
I seguenti diagrammi delle sequenze illustrano le interazioni tra l'amministratore dell'accettazione e le varie parti che gestiscono le prenotazioni dei ricoveri; la stessa cosa vale per l'amministratore del pronto soccorso con le prenotazioni dei ricoveri di emergenza.

Nel primo caso sono riportate le funzioni che permettono di prenotare un nuovo ricovero, nel solo caso in cui ci siano dei posti letto disponibili. L'aggiunta della prenotazione avviene attraverso la funzione "add_prenotazione()" della classe *VistaInserisciPrenotazione*.

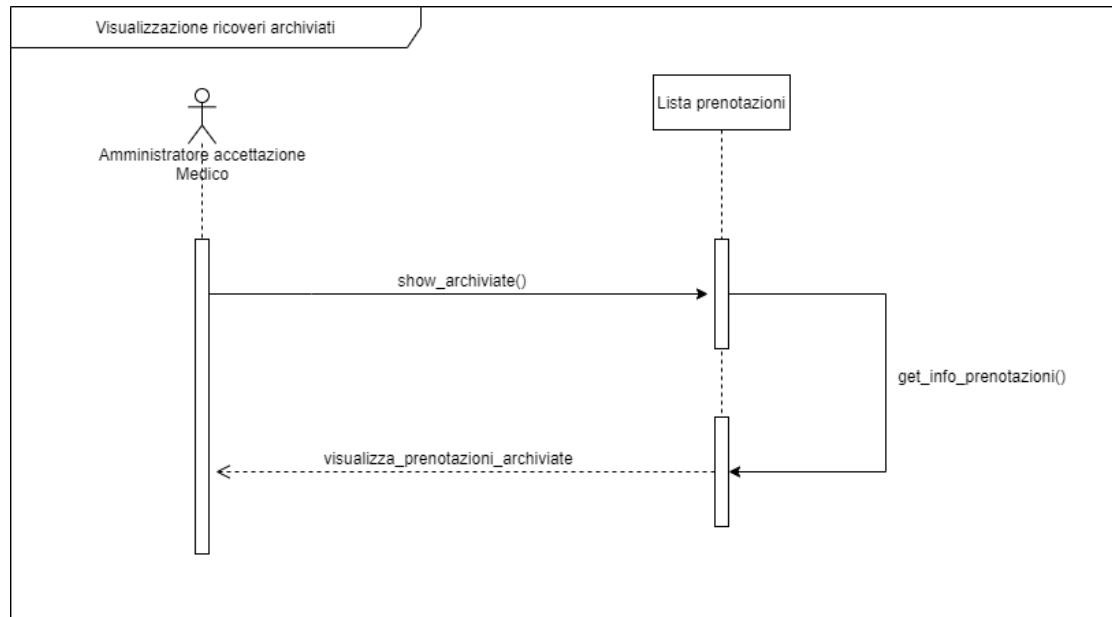


Nel secondo diagramma possiamo osservare le funzioni che permettono a tutti gli utenti di visualizzare un ricovero attraverso la funzione "get_info_prenotazioni()"; di conseguenza, se necessario, è possibile disdire una prenotazione o liberare un posto letto. L'amministratore dell'accettazione può visualizzare tutti i tipi di ricovero, mentre l'amministratore del pronto soccorso può visualizzare solo quelli di emergenza.

Nel terzo diagramma possiamo osservare le funzioni che permettono all'amministratore dell'accettazione di visualizzare i posti disponibili in ogni reparto e quelli occupati, attraverso la funzione "show_disponibilità()".

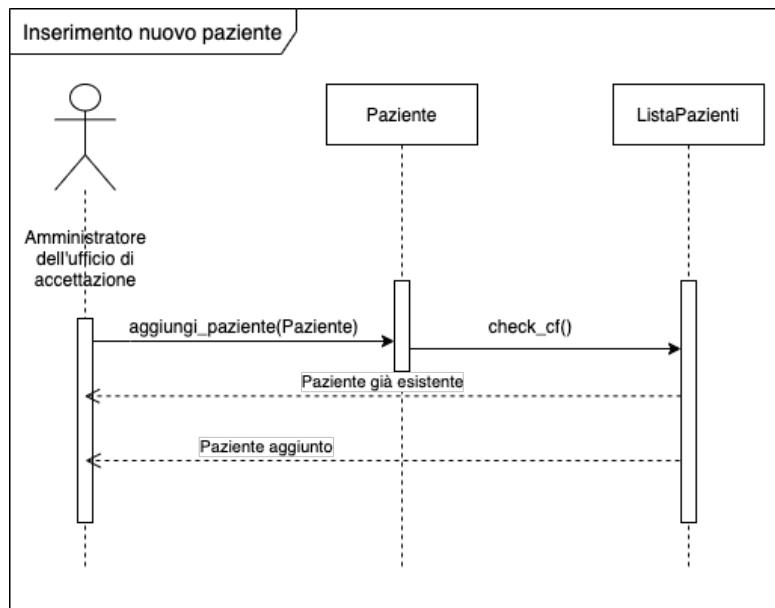


Nel diagramma sottostante vengono rappresentate le funzioni che consentono di visualizzare la lista dei ricoveri passati, che sono stati salvati nel sistema. A questa funzionalità possono accedere sia l'amministratore dell'ufficio di accettazione che il medico.



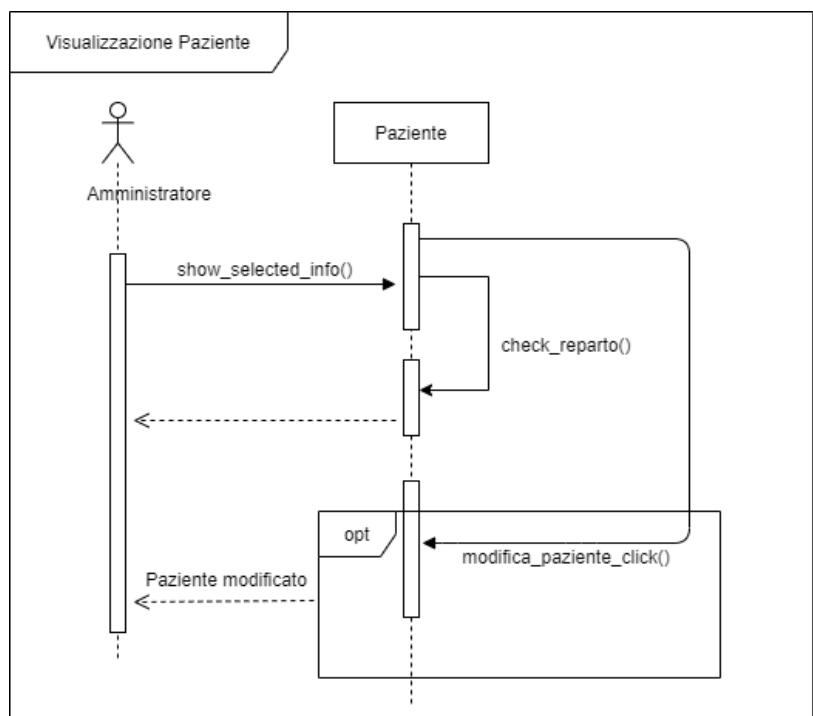
PAZIENTI:

I seguenti diagrammi delle sequenze mostrano le interazioni tra gli amministratori con le varie parti del sistema che gestiscono i pazienti.

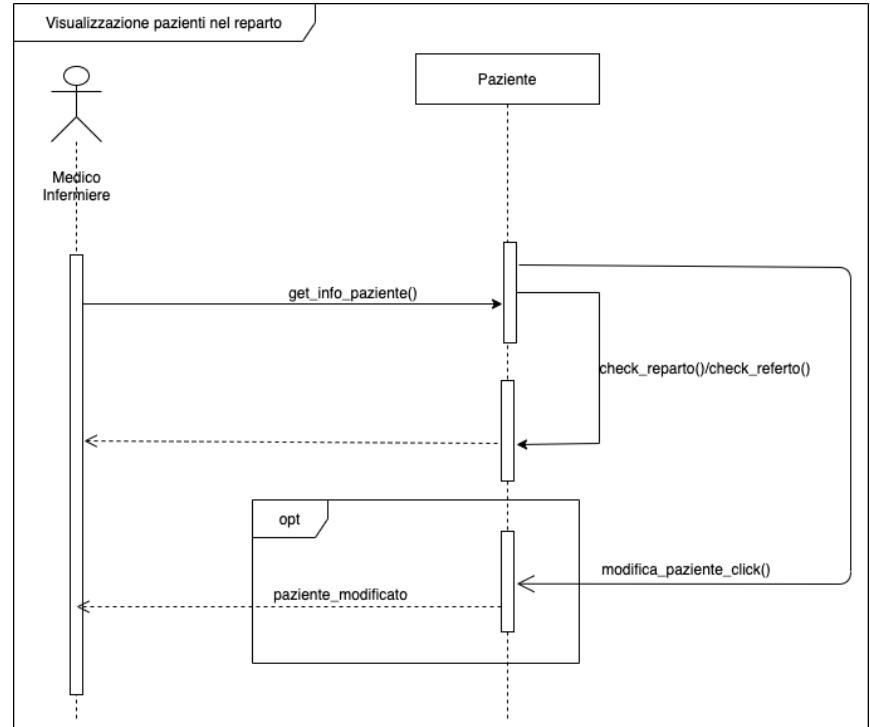


Nel primo caso sono riportate le sequenze necessarie all'inserimento di un nuovo paziente all'interno dell'archivio, attraverso la funzione "aggiungi_paziente". Ovviamente il software controllerà che il paziente non sia già presente a sistema, attraverso il controllo del codice fiscale.

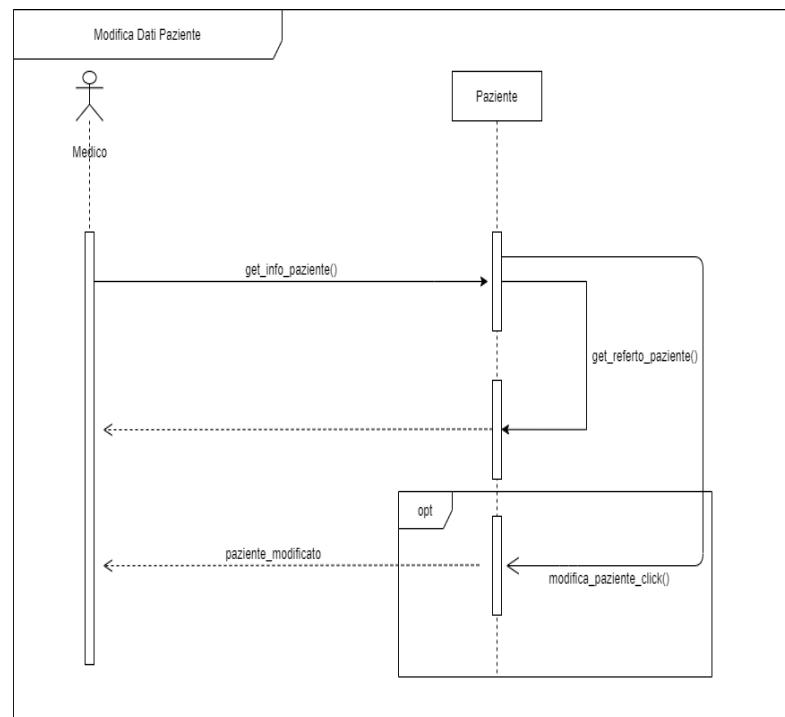
Nel secondo caso, invece, sono illustrate le funzioni che permettono la visualizzazione dell'anagrafe di un determinato paziente. Se necessario è possibile modificare i dati e/o visualizzare se il paziente è ricoverato nella struttura.



Il seguente diagramma riguarda l'interazione tra il medico e/o l'infermiere e le liste dei pazienti ricoverati nel rispettivo reparto. Al medico è infatti concesso di visualizzare i dati anagrafici e clinici di ogni paziente, e se necessario modificarli.



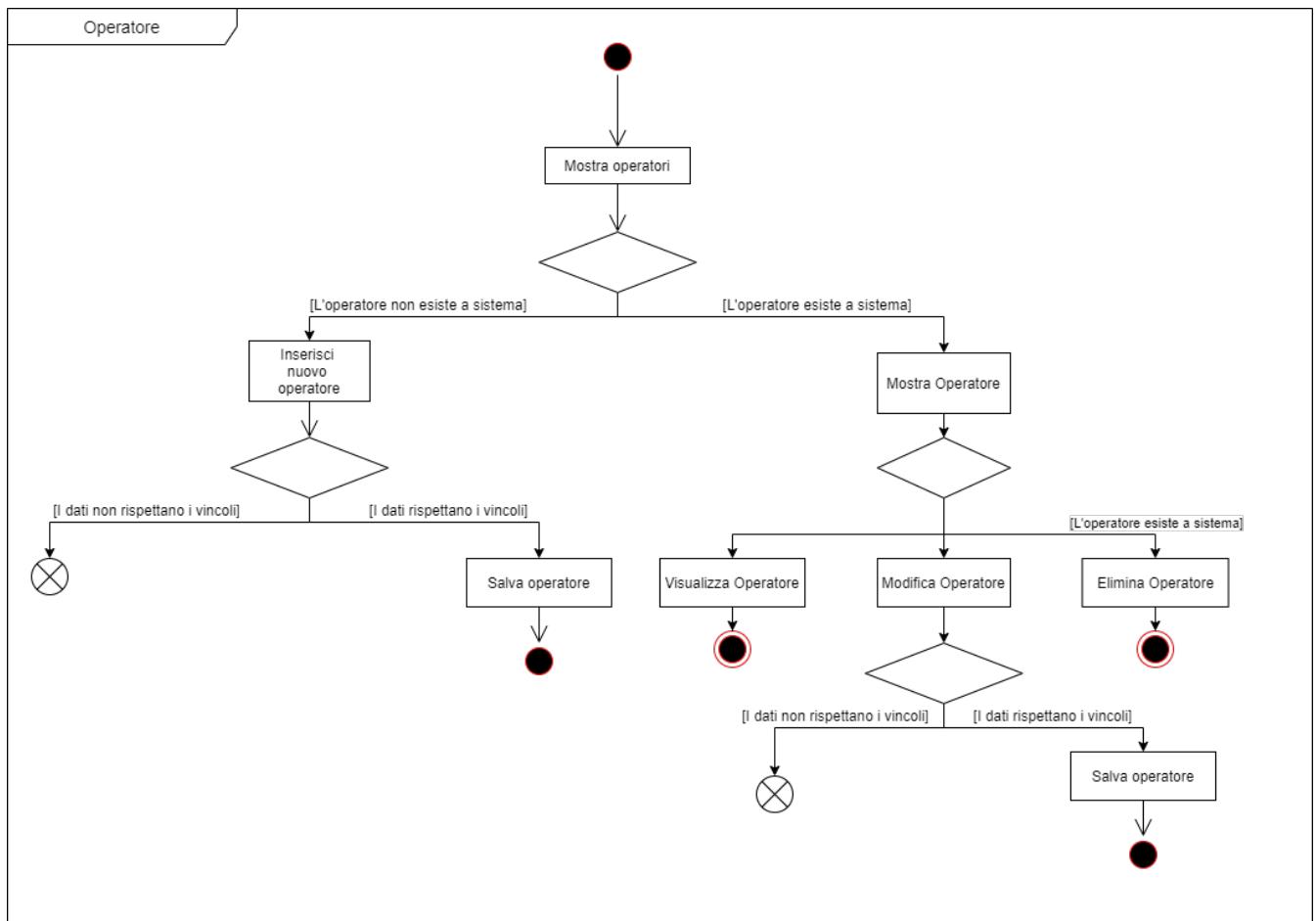
Questo diagramma riguarda sempre il medico, e mostra le funzioni che gli permettono di modificare i dati anagrafici di un paziente e quelli clinici, aggiungendo un nuovo referto.



DIAGRAMMI DELLE ATTIVITA'

OPERATORE

I seguenti diagrammi descrivono quelle che sono le attività relative alla gestione degli operatori. L'amministratore dell'accettazione è in grado di visualizzare un operatore (dati anagrafici e credenziali di accesso), solo se quest'ultimo è presente nel sistema, dunque è attualmente impiegato. Di conseguenza, può scegliere di modificare le informazioni di un dipendente (dati anagrafici), e se necessario (il dipendente non è più un impiegato della struttura), può eliminare il dipendente dal sistema. Se viene assunto un nuovo operatore è possibile inserirlo solo se vengono rispettati tutti i vincoli (l'amministratore deve inserire correttamente tutti i dati).

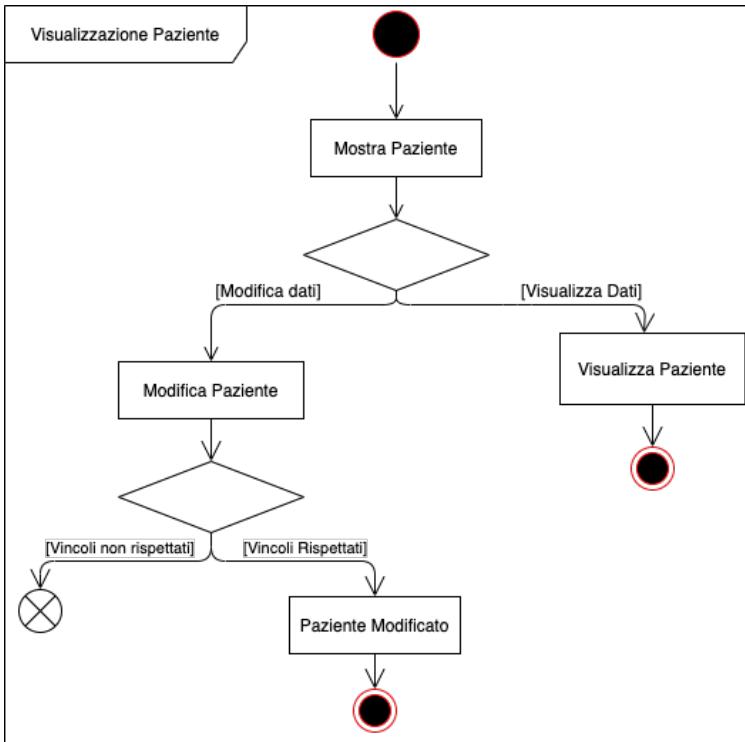


PAZIENTE

I seguenti diagrammi descrivono le attività che consentono agli amministratori e agli operatori (medici e infermieri) di poter

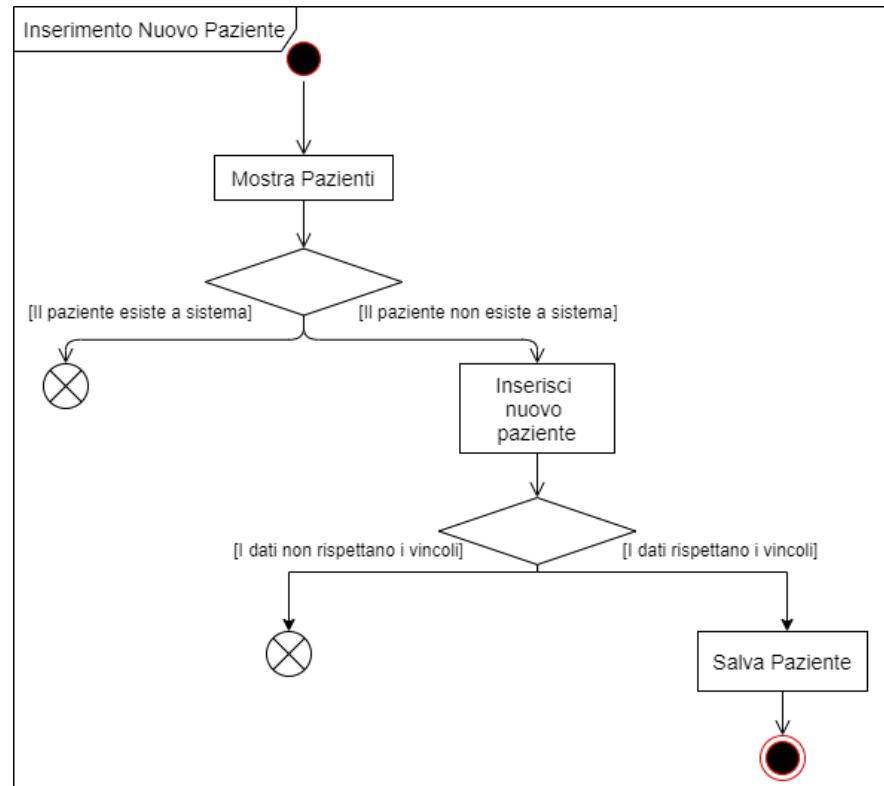
visualizzare i dati di un paziente, solo se quest'ultimo è presente a sistema. Gli utenti possono scegliere di modificare i dati anagrafici del paziente se non sono stati inseriti correttamente.

La modifica può avvenire solo se tutti i vincoli sono rispettati (ad esempio il codice fiscale deve essere univoco e i devono essere tutti inseriti).



Gli operatori della struttura sono in grado di inserire un nuovo paziente se non è già presente a sistema, e dunque salvarlo nella lista dei pazienti della struttura.

L'inserimento avviene solo se vengono rispettati tutti i vincoli.



SERVIZIO

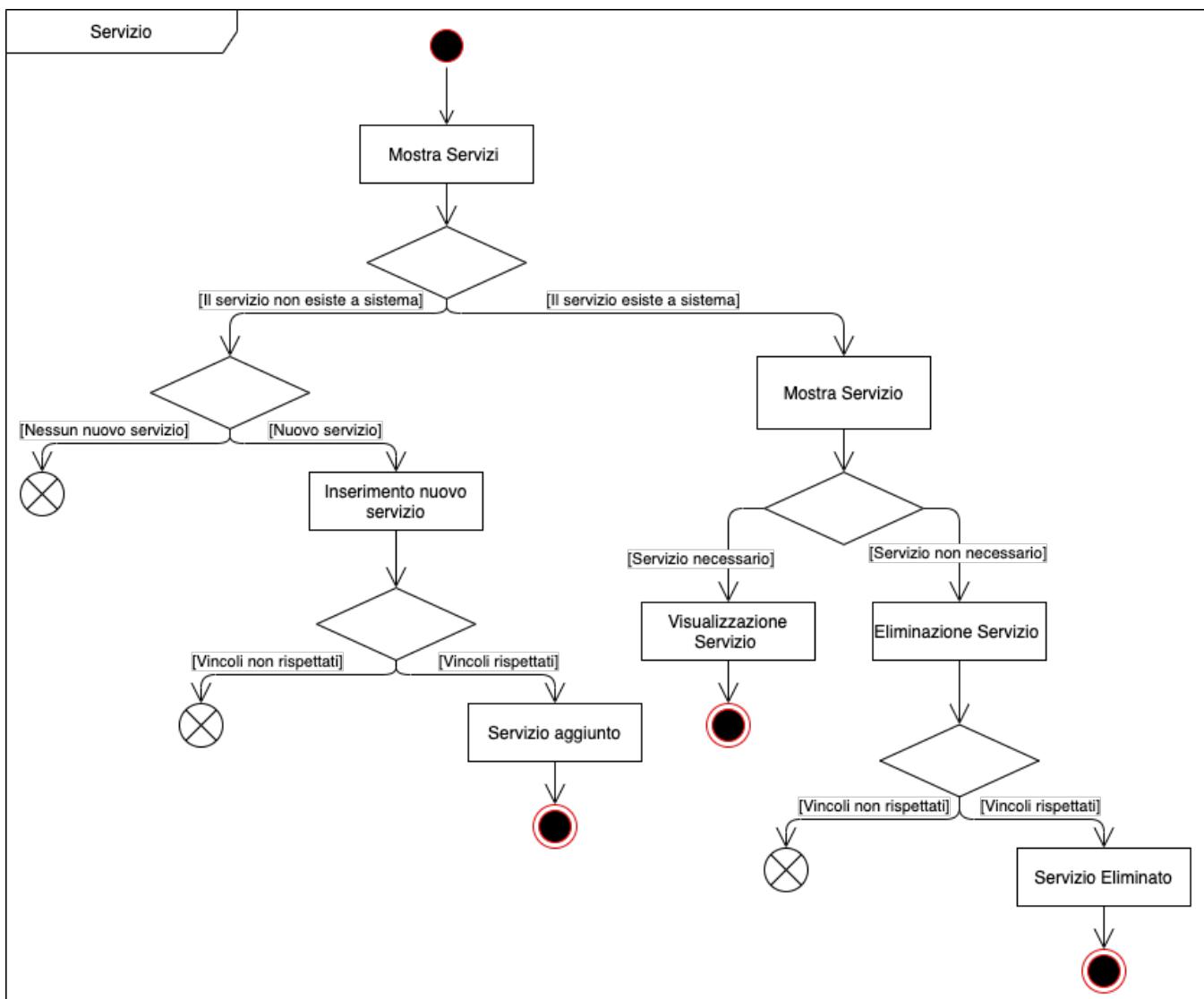
Il diagramma sottostante descrive le attività legate ai servizi della struttura ospedaliera.

L'amministratore dell'ufficio dell'accettazione, se necessario, è in grado di inserire nuovi servizi se non esistono già a sistema, e se tutti i vincoli vengono rispettati (deve scrivere correttamente tutti i dati).

Se il servizio esiste già a sistema è possibile visualizzarlo: quindi verranno mostrati il reparto, il posto letto e il suo stato (disponibile o non disponibile).

A questo punto, l'utente ha la possibilità di eliminare il servizio (qual'ora non fosse più necessario all'interno della struttura ospedaliera) o semplicemente visualizzarlo.

L'eliminazione può avvenire solo se tutti i vincoli vengono rispettati: il servizio è attualmente disponibile e non è prenotato.

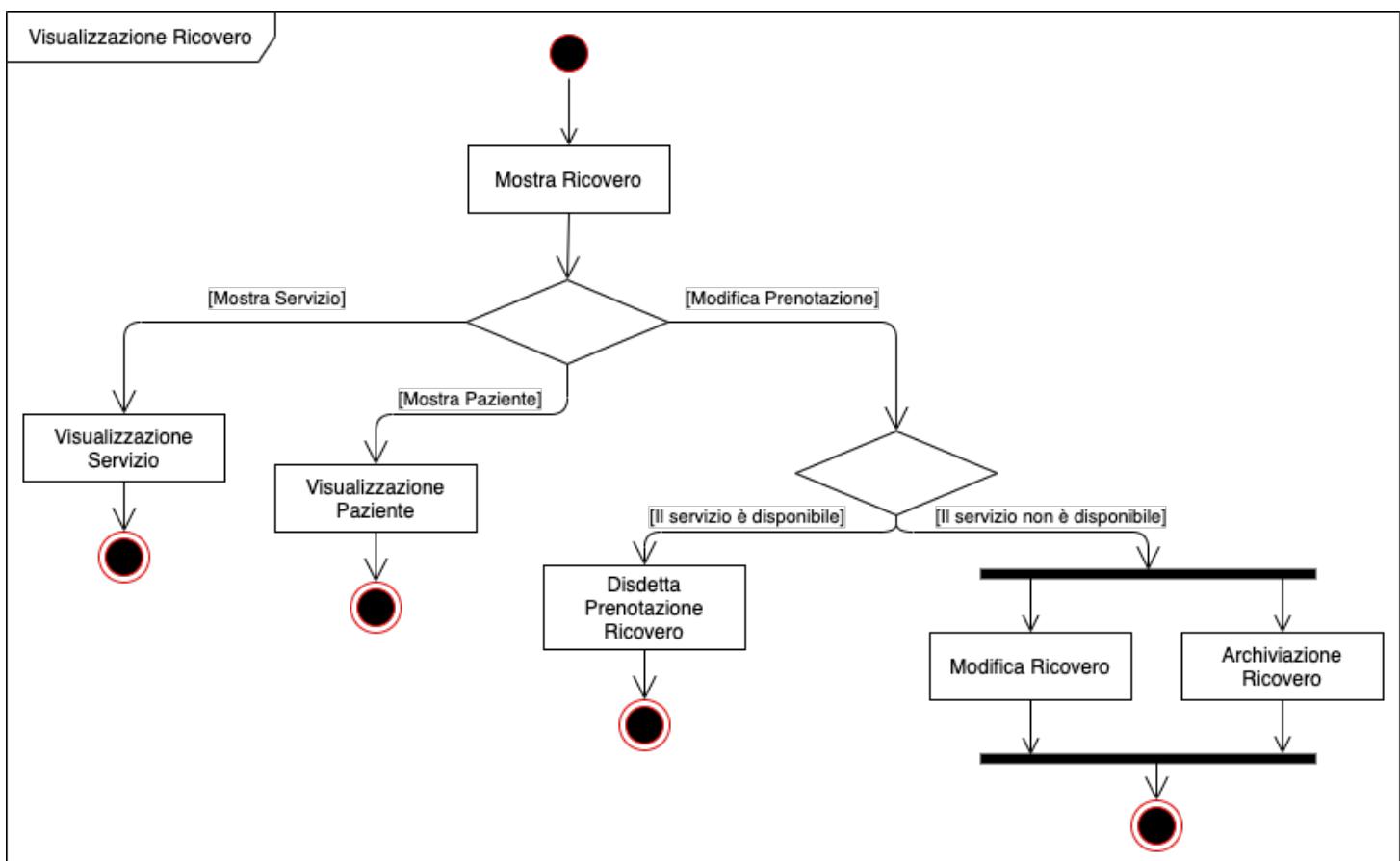


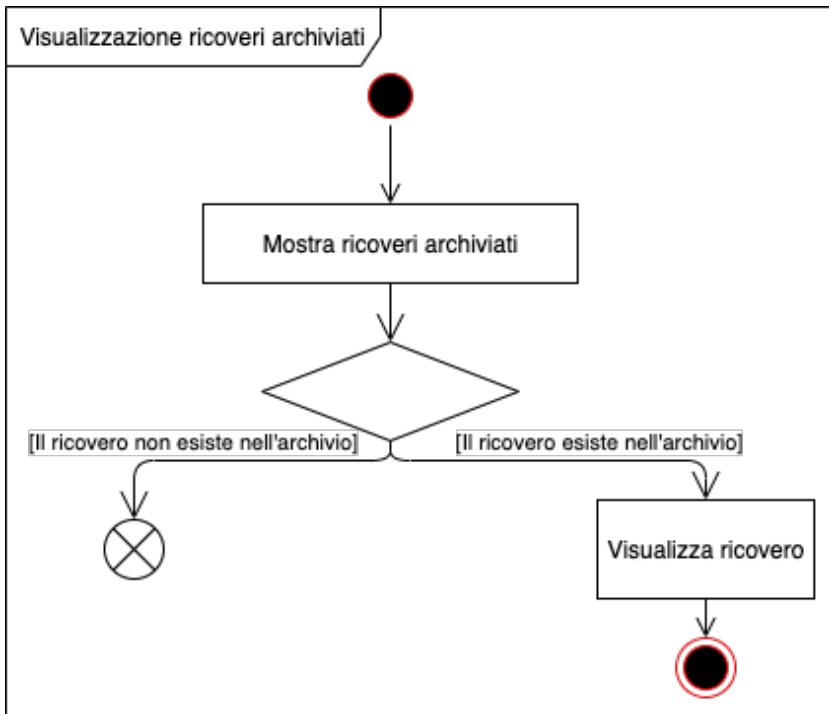
PRENOTAZIONE

Il seguenti diagrammi mostrano le attività legate alle prenotazioni.

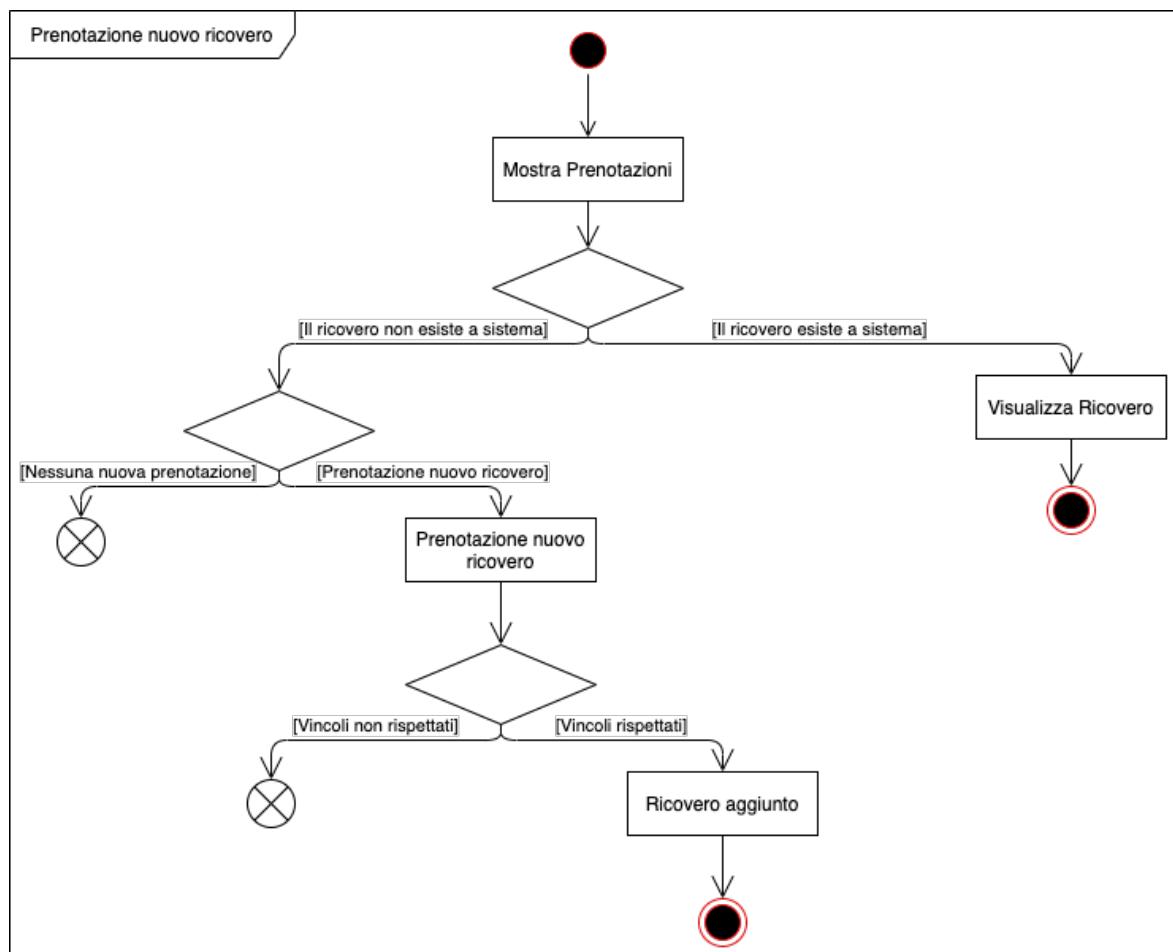
Gli operatori sono in grado di poter visualizzare le informazioni dei ricoveri presenti nel sistema: quest'ultimo mostrerà a schermo i dati relativi al paziente o al servizio, in base alla scelta dell'utente.

L'utente è inoltre in grado di modificare una prenotazione: liberando un posto letto (dunque archiviando la prenotazione nella lista delle prenotazioni passate) oppure disdicendo una prenotazione (solo se il paziente non è ancora ricoverato, quindi il servizio è disponibile).

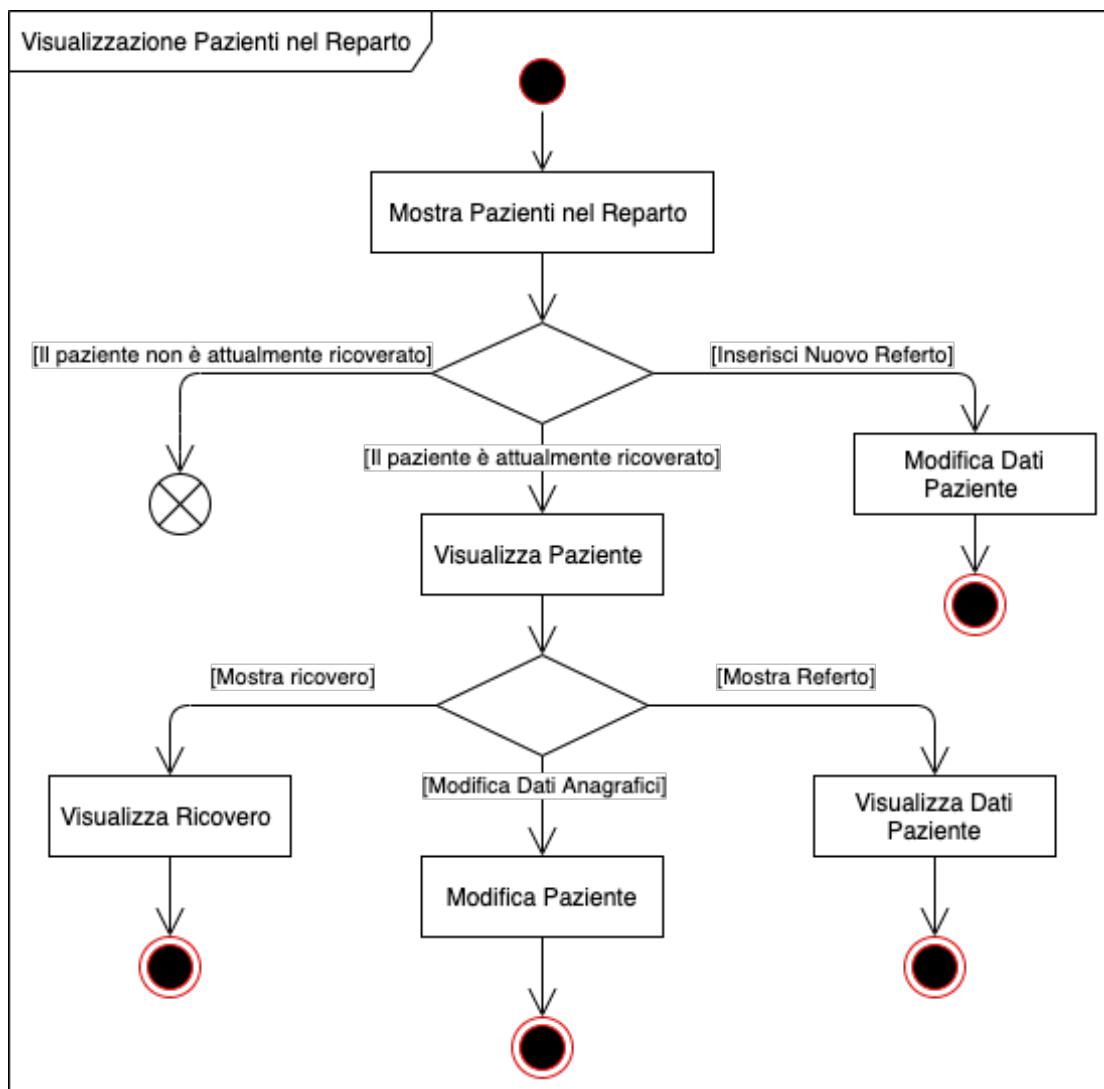




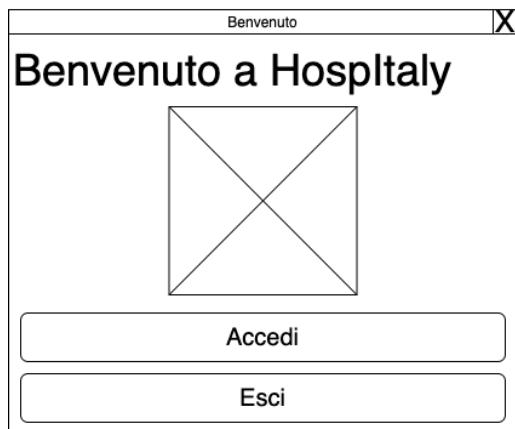
Il diagramma sottostante mostra l'attività che consente la prenotazione di un nuovo ricovero. Ciò si verifica solo se il ricovero non esiste già a sistema, e se tutti i vincoli delle prenotazioni vengono rispettati (ad esempio i servizi disponibili, la prenotazione viene effettuata una settimana prima dell'inizio del ricovero, il paziente non ha altre prenotazioni in altri reparti in quel periodo).



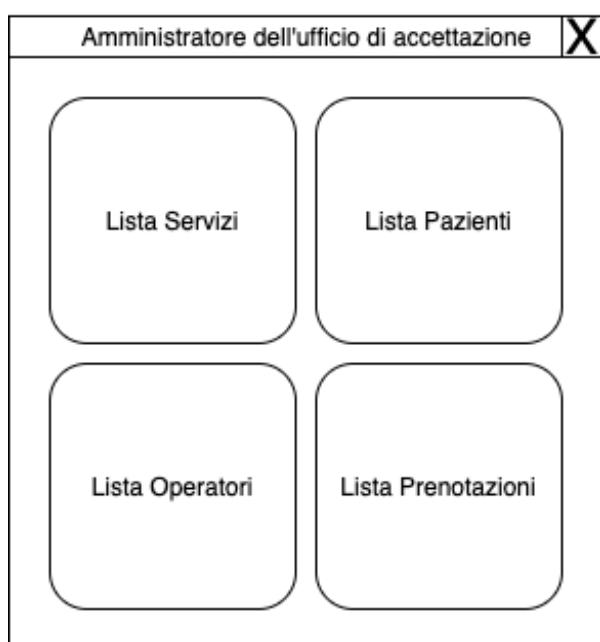
Gli utenti medico e infermiere sono in grado di visualizzare i pazienti attualmente ricoverati, in ogni reparto, dunque il sistema è in grado di mostrare il ricovero (la prenotazione), il referto (cartella clinica) del paziente stesso o permette all'utente di modificare i dati anagrafici del paziente.
Al medico è inoltre concesso di inserire un nuovo referto alla cartella clinica del sistema.



MOCKUP DI LIVELLO ZERO



Il software al momento dell'avvio richiede l'autenticazione da parte del client, in modo tale da verificare che egli sia registrato all'interno del sistema, ovvero che sia effettivamente un impiegato della struttura ospedaliera.

A wireframe-style mockup of a credential input screen titled "HospItaly". It has fields for "Username" and "Password", each with an associated input box. Below the password field is a "Accedi" (Log In) button.

In base alle credenziali inserite, il sistema riconosce l'utente e apre interfacce differenti.

Queste forniscono le informazioni sulle possibili attività che può effettuare un determinato utente.

Selezionando una di queste funzionalità, l'utente può proseguire con l'attività desiderata.

Lista Operatori	
Mauro Bartolini Maria Ferrari Giuseppe Romano Mario Rossi Isabel Sorrentino	<input type="button" value="Apri"/> <input type="button" value="Nuovo"/> Nome <input type="text"/> Cognome <input type="text"/> <input type="button" value="Cerca"/>

Alcune delle funzionalità più comuni:

- *Inserimento di nuove informazioni (pazienti, operatori, servizi, prenotazioni) all'interno dell'archivio del sistema della struttura ospedaliera.*

Nuovo Servizio	
Id <input type="text"/>	
Tipo <input type="text" value="ricovero"/> ▼	
Reparto <input type="radio"/> Oncologia <input type="radio"/> Chirurgia <input type="radio"/> Cardiologia <input type="radio"/> Medicina <input type="radio"/> Riabilitazione	
Posto letto <input type="text"/>	<input type="button" value="OK"/>

Ludovica Ferrari
Ludovica Ferrari Sesso: Femmina Luogo di nascita: Firenze Data di nascita: 09/02/1999 Codice Fiscale: FRRLVC99B49D612K Telefono: 3345709765 Email: ludo.ferrari@gmail.com <input type="button" value="Ricovero"/> <input type="button" value="Modifica"/>

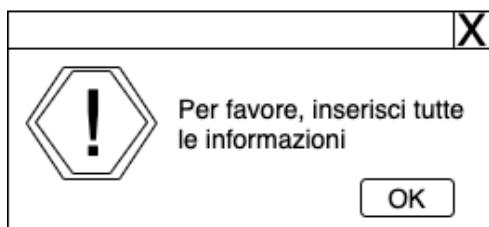
- *Visualizzazione di informazioni (pazienti, operatori, servizi, prenotazioni) presenti all'interno dell'archivio dell'ospedale.*

- Modifica di informazioni (pazienti, operatori, prenotazioni) presenti all'interno dell'archivio.

In questo caso il sistema permette all'utente di visualizzare a schermo tutti i dati registrati in precedenza. L'operatore, dunque, è in grado di modificare tutte le informazioni desiderate, senza dover necessariamente eliminare e reinserire una nuova entità nell'archivio.

Modifica Operatore	
Nome	Cognome
Mauro	Bartolini
Codice Fiscale	Email
BRTMRA83M12A271F	maurobartolini@gmail.com
Data di nascita	Luogo di nascita
12/08/1983	Ancona
Ruolo	
<input checked="" type="radio"/> Amministratore dell'ufficio di accettazione	<input type="radio"/> Infermiere
<input type="radio"/> Amministratore del pronto soccorso	<input type="radio"/> Medico
Password	
mauro12	<input type="button" value="Modifica"/>

In alcune situazioni si possono presentare dei messaggi di errore, dovuti a delle mancanze dell'utente.



I messaggi possono essere di vario tipo, a seconda della situazione in cui si trova l'utente, alcuni esempi possono essere:

- "Per favore, inserisci tutte le informazioni" : messaggio di errore nel caso in cui l'utente nell'inserimento di una nuova identità dimentica alcune informazioni
- "Bisogna prenotare almeno una settimana prima" : messaggio di errore restituito dal sistema nel caso in cui l'amministratore non rispetta i vincoli delle prenotazioni

- “*Selezionare almeno un paziente*” : messaggio di errore restituito nel caso in cui non venga selezionato alcun elemento dalla lista delle entità.
- “*Non ci sono posti disponibili*” : messaggio di errore che ottengono gli amministratori nel momento in cui i posti letto della struttura ospedaliera sono saturi.

Il nostro software è in grado di visualizzare a schermo alcuni dati relativi alla disponibilità della struttura, attraverso una tabella che si aggiorna continuamente:

Reparto	Posti disponibili	Posti occupati	Statistiche posti disponibili
Oncologia	3	1	75%
Chirurgia	4	0	100%
Cardiologia	2	2	50%
Riabilitazione	4	0	100%
Medicina	1	3	25%

Nel momento in cui si vogliono visualizzare determinati dati, relativi ad alcuni pazienti (ad esempio l’utente necessita di

X	Il paziente verrà ricoverato nel reparto di ONCOLOGIA il giorno 02/06/2021	X conoscere informazioni relative al ricovero, o alla cartella clinica di un paziente), verrà visualizzata una finestra con quanto desiderato.
----------	---	---

IMPLEMENTAZIONE DEL SOFTWARE

Il progetto è stato implementato utilizzando Python, un linguaggio di programmazione di alto livello, orientato agli oggetti.

Per lo sviluppo si è fatto riferimento al pattern architetturale Model-View-Controller(MVC). Ogni directory del progetto è stata suddivisa in tre componenti: *model* (responsabile della gestione dei dati e delle funzioni ad essi associate), *controller* (responsabile della gestione dell'interazione degli utenti e dei componenti) e *view* (gestisce e definisce il modo in cui i dati sono presentati all'utente) . In alcuni è stata aggiunta la directory *data* (contiene i file che hanno funzione di database).

Il progetto è dotato di interfacce grafiche, dunque è stato utilizzato PyQt5, un toolkit di Python per la creazione di GUI.

Nel main è stato creato un oggetto applicazione ed è stata richiamata la classe *VistaAutenticazione*, che permette all'utente di identificarsi. Successivamente il programma entra nel mainloop dell'applicazione con il metodo app.exec()

```
import sys
from PyQt5.QtWidgets import QApplication, QWidget

from autenticazione.views.VistaAutenticazione import VistaAutenticazione

if __name__ == '__main__':
    app = QApplication(sys.argv)
    vista_autenticazione = VistaAutenticazione()
    vista_autenticazione.show()
    sys.exit(app.exec())
```

In seguito, dunque sono state create varie interfacce grafiche che permettono all'utente di visualizzare diverse funzionalità, attraverso l'utilizzo di librerie differenti.

Le *view* (viste) definite nel programma estendono tutte dalla classe *QWidget*, classe base di tutti gli oggetti che si possono definire all'interno di un'interfaccia.

La classe *QWidget*, infatti, ha permesso di importare gli oggetti necessari da aggiungere alle interfacce, alcuni di questi sono:

- *QGridLayout* : dispone i widget in una griglia
- *QHBoxLayout* : dispone i widget orizzontalmente
- *QVBoxLayout* : dispone i widget verticalmente
- *QSpacerItem* : definisce spazi bianchi in un layout
- *QSizePolicy* : attributo del layout che desfinisce il ridimensionamento orizzontale e verticale
- *QPushButton* : pulsante di comando
- *QLabel* : definisce un testo o un'immagine
- *QLineEdit* : linea di testo in cui è possibile scrivere
- *QDateEdit* : fornisce un widget per modificare le date in base al widget *QDateTimeEdit* che a sua volta permette di modificare date e orari
- *QRadioButton* : widget che definisce una classe di oggetti selezionabili con un bottone e una linea di testo
- *QComboBox* : widget caratterizzato da una lista popup
- *QMessageBox* : finestra che mostra un messaggio e (facoltativamente) chiede all'utente di rispondere facendo click su uno dei pulsanti messi a disposizione. Ne esistono di diversi tipi, nel progetto sono stati utilizzati:
 - *Critical* : permette di visualizzare un messaggio di errore stabilito
 - *About* : permette di visualizzare un messaggio definito
- *QListView* : crea una lista all'interno di un layout
- *QFont* : permette di definire un font di una scritta
- *QTableWidget* : permette di visualizzare un oggetto in formato tabulare con un formato di default.
- *QPalette*: classe che contiene gruppi di colori per ogni stato del widget

Le interfacce create possono variare in base all'operatore che effettua l'autenticazione al programma.

Il programma infatti è in grado di verificare se un utente è autenticato, altrimenti non permette l'accesso.



Un esempio di interfaccia che l'utente visualizza immediatamente dopo aver effettuato l'autenticazione può essere la seguente.

In questo caso è stata presa in esame la *view* visualizzata dall'amministratore dell'ufficio di accettazione.



A livello implementativo, è stato creato un layout di tipo "griglia" (*QGridLayout*), al cui interno sono stati definiti quattro bottoni (*QPushButton*), i quali rappresentano le funzionalità dell'utente "amministratore dell'ufficio di accettazione" in macro-gruppi, attraverso l'utilizzo del metodo *addWidget()*.

```
def __init__(self, parent=None):
    super(VistaHomeAccettazione, self).__init__(parent)
    grid_layout = QGridLayout()

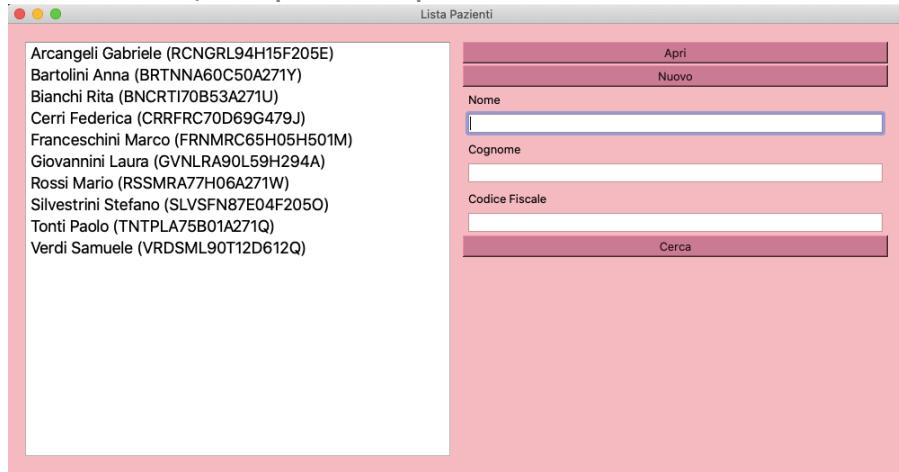
    grid_layout.addWidget(self.get_button("Lista Servizi",
                                         self.go_lista_servizi,
                                         "background-color: plum"), 0, 0)
    grid_layout.addWidget(self.get_button("Lista Pazienti",
                                         self.go_lista_pazienti,
                                         "background-color: lightpink"), 0, 1)
    grid_layout.addWidget(self.get_button("Lista Operatori",
                                         self.go_lista_operatori,
                                         "background-color: lightgreen"), 1, 0)
    grid_layout.addWidget(self.get_button("Lista Prenotazioni",
                                         self.go_lista_prenotazioni,
                                         "background-color: #ffffac"), 1, 1)

    self.setLayout(grid_layout)
    self.resize(400, 400)
    self.setWindowTitle("Amministratore dell'ufficio di accettazione")
    self.setStyleSheet("background-color: lightblue")
```

```
def get_button(self, titolo, on_click, colore):
    palette = QPalette()
    palette.setColor(QPalette.ButtonText, Qt.black)
    bottone = QPushButton(titolo)
    bottone.setPalette(palette)
    bottone.setSizePolicy(QSizePolicy.Expanding, QSizePolicy.Expanding)
    font = bottone.font()
    font.setPointSize(15)
    font.setBold(True)
    bottone.setFont(font)
    bottone.clicked.connect(on_click)
    bottone.setStyleSheet(colore)
    return bottone
```

La funzione "get_button()" permette di caratterizzare ciascun bottone con caratteristiche differenti: titolo, compito da eseguire al momento del click e colore di sfondo.

Ciascun bottone, dunque, risulta essere collegato ad un'ulteriore *view*, di questo tipo:



La seguente funzione permette di aggiornare ogni volta i dati (in questo caso si fa riferimento ai pazienti) presenti all'interno della lista. Ogni elemento (Item) della lista, risulta essere univoco.

```
def update_ui(self, nome_search="", cognome_search="", cf_search=""):
    self.listview_model = QStandardItemModel(self.list_view)
    self.controller.get_lista_pazienti().sort(key=lambda x: x.cognome.lower(), reverse=False)
    for paziente in self.controller.get_lista_pazienti():
        if nome_search == "" or ((paziente.nome.lower()).find(nome_search.lower())) == 0:
            if cognome_search == "" or ((paziente.cognome.lower()).find(cognome_search.lower())) == 0:
                if cf_search == "" or paziente.cf.lower() == cf_search.lower():
                    item = QStandardItem()
                    item.setText(paziente.cognome + " " + paziente.nome + " (" + paziente.cf.upper() + ")")
                    item.setEditable(False)
                    font = item.font()
                    font.setPointSize(18)
                    item.setFont(font)
                    self.listview_model.appendRow(item)
    self.list_view.setModel(self.listview_model)
```

Essa infatti viene richiamata in funzioni che:

- Riguardano la visualizzazione dei dati nella lista

```
def show_selected_info(self):
    if (len(self.list_view.selectedIndexes()) > 0):
        selected = self.list_view.selectedIndexes()[0].data()
        stringa = selected.split()
        paziente_selezionato = self.controller.get_paziente_by_cf(
            stringa[len(stringa)-1].replace('(', '').replace(')', ''))
        self.vista_paziente = VistaPaziente(paziente_selezionato,
                                              self.bool, self.controller.archivia_paziente_by_cf,
                                              self.update_ui)
        self.vista_paziente.show()
    else:
        QMessageBox.critical(self, 'Errore', "Selezionare un paziente", QMessageBox.Ok, QMessageBox.Ok)
```

- Riguardano l'inserimento di nuovi dati nella lista

```
def show_new_paziente(self):
    self.vista_inserisci_paziente = VistaInserisciPaziente(self.controller, self.update_ui)
    self.vista_inserisci_paziente.show()
```

- Riguardano la ricerca di dati all'interno della lista

```
def search_paziente_(self):
    self.update_ui(self.lineEdit_nome.text(), self.lineEdit_cognome.text(), self.lineEdit_cf.text())
```

Ovviamente tutte queste chimate effettuate dall'utente vengono gestite dalla classe *controller* ad essa associata.

Nel progetto, la classe *controller* contiene i metodi *get()* che restituiscono i valori dei dati del *model*, o funzioni che richiamano altre funzionalità del *model* stesso.

In questo caso, all'interno del costruttore della classe *model* è stata definita la lista con cui è in grado di interagire l'utente.

```
def __init__(self):
    super(ListaPazienti, self).__init__()
    self.lista_pazienti = []
    if os.path.isfile('listapazienti/data/lista_pazienti_salvata.pickle'):
        with open('listapazienti/data/lista_pazienti_salvata.pickle', 'rb') as f:
            self.lista_pazienti = pickle.load(f)
```

È stato utilizzato il modulo *pickle* di Python, il quale contiene funzioni in grado di leggere e scrivere valori Python in un formato binario. In particolar modo il metodo *pickle.load()* è in grado di leggere i valori di un file già esistente, mentre il metodo *pickle.dump()* è in grado di scrivere dei valori all'interno di un file.

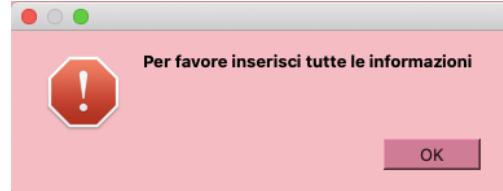
Il primo è stato utilizzato per visualizzare i dati presenti nel sistema della struttura ospedaliera, mentre il secondo per aggiungerne di nuovi.

```
def save_data(self):
    with open('listaoperatori/data/list_operatori_salvata.pickle', 'wb') as handle:
        pickle.dump(self.lista_operatori, handle, pickle.HIGHEST_PROTOCOL)
```

I metodi che permettono l'inserimento di nuovi dati all'interno della lista, solitamente effettuano dei controlli, a causa di alcuni vincoli imposti nel sistema, che vengono verificati tramite delle if e/o elif.

```
def add_paziente(self):
    nome = self.info["Nome"].text()
    cognome = self.info["Cognome"].text()
    sesso = self.info["Sesso"].text()
    luogodinascita = self.info["Luogo di nascita"].text()
    datadinascita = self.info["Data di nascita"].text()
    cf = self.info["Codice fiscale"].text()
    telefono = self.info["Telefono"].text()
    email = self.info["Email"].text()
    if nome == "" or cognome == "" or sesso == "" or \
        luogodinascita == "" or datadinascita == "" or \
        cf == "" or telefono == "" or email == "":
        QMessageBox.critical(self, 'Errore', 'Per favore inserisci tutte le informazioni',
                             QMessageBox.Ok, QMessageBox.Ok)
    elif self.controller.check_cf(cf):
        QMessageBox.critical(self, 'Errore', "Il paziente è già stato inserito nella lista",
                             QMessageBox.Ok, QMessageBox.Ok)
    elif self.check_email(email):
        QMessageBox.critical(self, 'Errore', "Per favore, inserisci correttamente l'email",
                             QMessageBox.Ok, QMessageBox.Ok)
    elif len(telefono) < 8 or len(telefono) > 10:
        QMessageBox.critical(self, 'Errore', "Numero di telefono sbagliato", QMessageBox.Ok, QMessageBox.Ok)
    else:
        self.controller.aggiungi_paziente(Paziente(nome, cognome, sesso, luogodinascita,
                                                    datadinascita, cf, telefono, email))
    self.callback()
    self.close()
```

Se alcuni di essi non vengono rispettati, si visualizza a schermo un messaggio di errore, utilizzando la classe *QMessageBox.critical* che obbliga l'utente ad effettuare alcune attività.



I dati all'interno delle liste vengono inseriti utilizzando il metodo *append()*, il quale inserisce un elemento, passatogli

```
def aggiungi_servizio(self, servizio1):
    self.listaservizi.append(servizio1)
```

come argomento, in coda alla lista stessa.

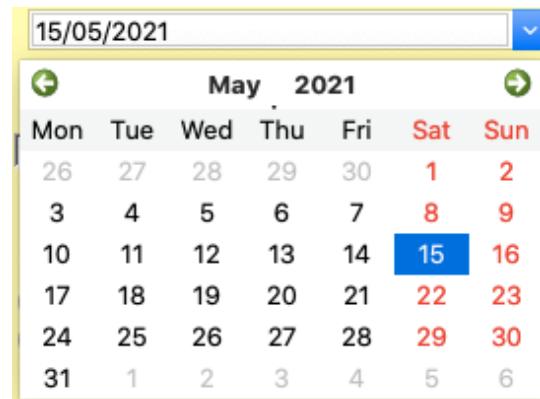
La rimozione dei dati dalle liste, invece, è stata gestita utilizzando il metodo *remove()*, il quale rimuove un elemento specifico, passatogli come argomento.

```
def rimuovi_operatore_by_id_(self, id):
    for operatore in self.lista_operatori:
        if operatore.id == id:
            self.lista_operatori.remove(operatore)
            return True
    return False
```

Nel progetto è stata posta una particolare cura alla gestione delle date. Il software, infatti, ha lo scopo di gestire i ricoveri di una struttura ospedaliera, dunque le prenotazioni caratterizzate da una data di inizio ed una data di fine.

Per l'inserimento delle date (da parte dell'utente), è stata utilizzata la libreria *QDateEdit*: fornisce un widget per la modifica della data (come definito precedentemente). L'utente è in grado di visualizzare un calendario che permette di selezionare la data desiderata.

```
def get_datanascita(self, tipo):
    lbl = QLabel(tipo)
    font_lbl = lbl.font()
    font_lbl.setPointSize(17)
    font_lbl.setBold(True)
    lbl.setFont(font_lbl)
    self.grid_layout.addWidget(lbl, 4, 0)
    dateedit = QDateEdit(calendarPopup=True)
    dateedit.setDate(QDateTime.currentDateTime())
    dateedit.setDateFormat("dd/MM/yyyy")
    self.grid_layout.addWidget(dateedit, 5, 0)
    self.info[tipo] = dateedit
```



A livello implementativo, il calendario restituisce una data in formato 'dd/MM/yyyy' (altri formati non sono accettati dal sistema, il quale restituirà un messaggio di errore).

Per la gestione del formato delle date è stato importato il modulo *datetime*, che consente la manipolazione e la formattazione delle date. In particolar modo sono state utilizzate le funzioni:

- *timedelta(int)* : rappresenta una durata ed è stata richiamata nella classe *VistaInserisciPrenotazione* come vincolo per nuove prenotazioni (obbligo di prenotare almeno una settimana prima della data di inizio).
- *strptime(str, "%d/%m/%Y")*: crea un oggetto *datetime*, nel formato desiderato a partire da una stringa passatogli come argomento. È stata utilizzata in varie classi per effettuare confronti con le date.

Altri widget presenti nel progetto sono i *QRadioButton*, caratterizzati da una serie di oggetti selezionalizzabili (come definito precedentemente). Sono un esempio quelli utilizzati in *VistaInserisciServizio*.

Reparto

- Oncologia
- Chirurgia
- Cardiologia
- Medicina
- Riabilitazione

```

def get_reparto(self, tipo):
    lbl = QLabel(tipo)
    font_lbl = lbl.font()
    font_lbl.setPointSize(17)
    font_lbl.setBold(True)
    lbl.setFont(font_lbl)
    self.v_layout.addWidget(lbl)
    for reparto in self.list_reparti:
        rbtn = QRadioButton(reparto)
        self.v_layout.addWidget(rbtn)
        rbtn.toggled.connect(self.reparto_onClicked)
    self.v_layout.addWidget(self.lbl_reparto)
    self.info[tipo] = self.lbl_reparto

def reparto_onClicked(self):
    rbtn = self.sender()
    if rbtn.isChecked() == True:
        self.lbl_reparto.setText(rbtn.text())

```

Altri widget utilizzati nel sistema sono i *QComboBox()*. Questi ultimi, sono dei menù a tendina che offrono all’utente alcune scelte. Un esempio è riportato nella classe *VistaInserisciPrenotazione*, in cui l’utente deve selezionare un determinato paziente registrato nella struttura.

```

def get_paziente(self, titolo):
    lbl = QLabel(titolo)
    font_lbl = lbl.font()
    font_lbl.setPointSize(17)
    font_lbl.setBold(True)
    lbl.setFont(font_lbl)
    self.grid_layout.addWidget(lbl, 2, 0)
    combo_pazienti = QComboBox()
    combo_pazienti.setStyleSheet("background-color: white")
    for paziente in self.controller_pazienti.get_lista_pazienti():
        combo_pazienti.addItem(paziente.cf)
    self.grid_layout.addWidget(combo_pazienti, 3, 0)
    combo_pazienti.activated[str].connect(self.paziente_onClicked)
    self.grid_layout.addWidget(self.label_paziente, 4, 0)
    self.info[titolo] = self.label_paziente

def paziente_onClicked(self, text):
    self.label_paziente.setText(text)

```

metodo *activated[str].connect()* viene richiamata la funzione *paziente_onClicked()* che restituisce una *QLabel* con il paziente selezionato. A tale paziente verrà assegnata la prenotazione.

A livello implementativo sono stati definiti tutti gli elementi all’interno di una lista, *list_reparti*, e per ognuno di essi è stato generato un *QRadioButton*. Per ciascun elemento, attraverso il metodo *toggled.connect()* viene richiamato il metodo *reparto_onClicked()*. Quest’ultimo verifica se il bottone è stato selezionato attraverso il metodo booleano *isChecked()*.



A livello implementativo è stata definita una *QComboBox* contenente tutti gli elementi della lista dei pazienti della struttura ospedaliera. Attraverso il

Il sistema in aggiunta è in grado di poter visualizzare alcune informazioni relativamente alla disponibilità della struttura ospedaliera. Con lo scopo di visualizzare in maniera ordinata tali informazioni, è stata implementata una funzione in grado di generare una tabella con i dati costantemente aggiornati.

1 Reparto	2 Posti Disponibili	3 Posti Occupati	4 Statistiche sui posti disponibili
2 Oncologia	3	1	75.0%
3 Chirurgia	3	1	75.0%
4 Cardiologia	3	1	75.0%
5 Medicina	4	0	100.0%
6 Riabilitazione	4	0	100.0%

Dapprima sono state definite alcune funzioni per effettuare alcuni calcoli: numero di posti disponibili nella struttura, numero di posti occupati e/o numero di posti totali. Inoltre è stata aggiunta una funzione in grado di riportare la percentuale dei posti disponibili per ogni reparto.

```
def posti_disponibili(self, reparto):
    contatore_posti_disponibili = 0
    controller_servizi = ControlloreListaServizi()
    for servizio in controller_servizi.get_lista_servizi():
        if servizio.reparto.lower() == reparto.lower():
            if (servizio.is_disponibile()):
                contatore_posti_disponibili += 1
    return contatore_posti_disponibili

def posti_occupati(self, reparto):
    contatore_posti_occupati = 0
    controller_servizi = ControlloreListaServizi()
    for servizio in controller_servizi.get_lista_servizi():
        if servizio.reparto.lower() == reparto.lower():
            if servizio.is_disponibile() == False:
                contatore_posti_occupati += 1
    return contatore_posti_occupati
```

```
def posti_totali(self, reparto):
    contatore_posti_totali = 0
    controller_servizi = ControlloreListaServizi()
    for servizio in controller_servizi.get_lista_servizi():
        if servizio.reparto.lower() == reparto.lower():
            contatore_posti_totali += 1
    return contatore_posti_totali

def statistiche(self, reparto):
    numero = (self.posti_disponibili(reparto)/self.posti_totali(reparto))*100
    statistica = str(numero) + '%'
    return statistica
```

Per implementare tale tabella è stata utilizzata la classe *QTableWidget*, la quale permette di visualizzare un oggetto in formato tabulare con un formato di default.

Per ogni reparto, dunque sono stati effettuati gli opportuni calcoli, i quali sono stati poi inseriti nell'apposita cella della tabella.

Quest'ultima viene visualizzata a schermo tramite il metodo *show()*, utilizzato anche per le altre view del programma.

```

def show_disponibilita(self):
    self.tableWidget = QTableWidget()
    self.tableWidget.setRowCount(6)
    self.tableWidget.setColumnCount(4)
    self.tableWidget.setItem(0, 0, QTableWidgetItem("Reparto"))
    self.tableWidget.setItem(0, 1, QTableWidgetItem("Posti Disponibili"))
    self.tableWidget.setItem(0, 2, QTableWidgetItem("Posti Occupati"))
    self.tableWidget.setItem(0, 3, QTableWidgetItem("Statistiche sui posti disponibili"))
    self.tableWidget.setItem(1, 0, QTableWidgetItem("Oncologia"))
    self.tableWidget.setItem(1, 1, QTableWidgetItem(str(self.posti_disponibili("oncologia"))))
    self.tableWidget.setItem(1, 2, QTableWidgetItem(str(self.posti_occupati("oncologia"))))
    self.tableWidget.setItem(1, 3, QTableWidgetItem(self.statistiche("oncologia")))
    self.tableWidget.setItem(2, 0, QTableWidgetItem("Chirurgia"))
    self.tableWidget.setItem(2, 1, QTableWidgetItem(str(self.posti_disponibili("chirurgia"))))
    self.tableWidget.setItem(2, 2, QTableWidgetItem(str(self.posti_occupati("chirurgia"))))
    self.tableWidget.setItem(2, 3, QTableWidgetItem(self.statistiche("chirurgia")))
    self.tableWidget.setItem(3, 0, QTableWidgetItem("Cardiologia"))
    self.tableWidget.setItem(3, 1, QTableWidgetItem(str(self.posti_disponibili("cardiologia"))))
    self.tableWidget.setItem(3, 2, QTableWidgetItem(str(self.posti_occupati("cardiologia"))))
    self.tableWidget.setItem(3, 3, QTableWidgetItem(self.statistiche("cardiologia")))
    self.tableWidget.setItem(4, 0, QTableWidgetItem("Medicina"))
    self.tableWidget.setItem(4, 1, QTableWidgetItem(str(self.posti_disponibili("medicina"))))
    self.tableWidget.setItem(4, 2, QTableWidgetItem(str(self.posti_occupati("medicina"))))
    self.tableWidget.setItem(4, 3, QTableWidgetItem(self.statistiche("medicina")))
    self.tableWidget.setItem(5, 0, QTableWidgetItem("Riabilitazione"))
    self.tableWidget.setItem(5, 1, QTableWidgetItem(str(self.posti_disponibili("riabilitazione"))))
    self.tableWidget.setItem(5, 2, QTableWidgetItem(str(self.posti_occupati("riabilitazione"))))
    self.tableWidget.setItem(5, 3, QTableWidgetItem(self.statistiche("riabilitazione")))
    self.tableWidget.horizontalHeader().setStretchLastSection(True)
    self.tableWidget.horizontalHeader().setSectionResizeMode(QHeaderView.Stretch)
    self.tableWidget.verticalHeader().setStretchLastSection(True)
    self.tableWidget.verticalHeader().setSectionResizeMode(QHeaderView.Stretch)
    self.tableWidget.setWindowTitle("Posti disponibili in ogni reparto")
    self.tableWidget.setGeometry(500, 500, 500, 500)
    self.tableWidget.move(self.pos())
    self.tableWidget.setStyleSheet("background-color: #ffffcc")
    self.tableWidget.show()

```

TESTING DEL SISTEMA

Il nostro software è stato testato tramite l'utilizzo dell'applicativo PyUnit. Quest'ultimo consente l'esecuzione di test automatici sui singoli componenti del sistema per verificare la correttezza del codice.

Sono stati isolati alcuni frammenti di codice, in modo tale da testare le funzioni in modo più rigoroso: in questo modo eventuali errori possono essere subito identificati e corretti.

Sono stati inseriti alcuni dati finti e sono state testate alcune delle funzioni più significative del sistema, in particolare quelle del Controller. Infatti, l'architettura scelta, MVC, è caratterizzata da una classe Controller che gestisce le interazioni tra la classe View e Model, dunque le sue funzioni sono di grande importanza.

Di seguito sono mostrati e commentati alcuni dei test effettuati:

- La funzione testata *aggiungi_nuovo_referto_paziente()* è una funzione appartenente alla classe *ControllorePaziente*. Ha lo scopo di inserire un referto ad un paziente. Il test verifica che l'attributo *referto* dell'entità *Paziente* non sia una stringa vuota.

```
# Test della funzione aggiungi_nuovo_referto_paziente()
def test_aggiungi_nuovo_referto_paziente(self):
    self.paziente = Paziente(nome="Mario", cognome="Rossi", sesso="Maschio",
                             luogodinascita="Ancona", datadinascita="12/07/1980",
                             cf="RSSMRA80L12A271V", telefono="3345609712",
                             email="mariorossi@gmail.com")
    self.assertEqual(self.paziente.referto, "")
    self.paziente.add_referto(self.paziente.referto == "Il paziente accusa dolori")
    self.assertNotEqual(self.paziente.referto, "")
```

- La funzione testata *aggiungi_prenotazione()* è una funzione appartenente alla classe *ControlloreListaPrenotazioni*. Ha lo scopo di inserire una nuova prenotazione all'interno del sistema della struttura ospedaliera. Il test verifica che, una volta inserita una nuova prenotazione nella lista, quest'ultima non risulti vuota.

```

# Test della funzione aggiungi_prenotazione()
def test_aggiungi_prenotazione(self):
    self.lista_prenotazioni = ListaPrenotazioni()
    self.paziente = Paziente(nome="Mario", cognome="Rossi", sesso="Maschio",
                            luogodinascita="Ancona", datadinascita="12/07/1980",
                            cf="RSSMRA80L12A271V", telefono="3345609712",
                            email="mariorossi@gmail.com")
    self.servizio = Servizio(id="a1", nome="ricovero in Oncologia (a1)", tipo="ricovero",
                            reparto="Oncologia", posto letto="a1", disponibile=None)
    self.prenotazione = Prenotazione(id="id", paziente=self.paziente, servizio=self.servizio,
                                    data="28/05/2021", datafine="02/06/2021")
    self.assertListEqual(self.lista_prenotazioni.get_lista_prenotazioni(), [])
    self.lista_prenotazioni.aggiungi_prenotazione(self.prenotazione)
    self.assertIsNotNone(self.lista_prenotazioni.get_lista_prenotazioni())

```

- La funzione testata *libera_posto letto()* è una funzione appartenente alla classe *ControllorePrenotazione*. Ha lo scopo di rendere disponibile un servizio della struttura, modificando dunque la fine della prenotazione. Il test verifica che il servizio della struttura risulti essere disponibile.

```

# Test della funzione libera_posto letto()
def test_libera_posto letto(self):
    self.paziente = Paziente(nome="Mario", cognome="Rossi", sesso="Maschio",
                            luogodinascita="Ancona", datadinascita="12/07/1980",
                            cf="RSSMRA80L12A271V", telefono="3345609712",
                            email="mariorossi@gmail.com")
    self.servizio = Servizio(id="a1", nome="ricovero in Oncologia (a1)", tipo="ricovero",
                            reparto="Oncologia", posto letto="a1", disponibile=False)
    self.prenotazione = Prenotazione(id="id", paziente=self.paziente, servizio=self.servizio,
                                    data="28/05/2021", datafine="02/06/2021")
    self.prenotazione.libera_posto letto()
    self.assertTrue(self.servizio.disponibile)

```

- La funzione testata *aggiungi_servizio()* è una funzione appartenente alla classe *ControlloreListaServizi*. Ha lo scopo di inserire un nuovo servizio, all'interno della lista dei servizi della struttura. Il test verifica che, una volta inserito il nuovo servizio, la lista non risulti essere vuota.

```

# Test della funzione aggiungi_servizio()
def test_aggiungi_servizio(self):
    self.lista_servizi = ListaServizi()
    self.servizio = Servizio(id="a1", nome="ricovero in Oncologia (a1)", tipo="ricovero",
                            reparto="Oncologia", posto letto="a1", disponibile=True)
    self.assertListEqual(self.lista_servizi.get_lista_servizi(), [])
    self.lista_servizi.aggiungi_servizio(self.servizio)
    self.assertIsNotNone(self.lista_servizi.get_lista_servizi())

```

- La funzione testata `elimina_operatore_by_id()` è una funzione appartenente alla classe `ControlloreOperatore`. Ha lo scopo di eliminare un operatore, presente all'interno della lista degli operatori registrati nella struttura ospedaliera. Il test verifica che, una volta eliminato l'operatore selezionato, la lista risulti essere vuota.

```
# Test della funzione elimina_operatore_by_id()
def test_elimina_operatore_by_id(self):
    self.lista_operatori = ListaOperatori()
    self.operatore = Operatore(id="mariaferrari", nome="Maria", cognome="Ferrari",
                                cf="FRRMRA80C62F205M", datanascita="22/03/1980",
                                luogonascita="Milano", email="mariaferrari@gmail.com",
                                ruolo="Infermiere", password="mary22")
    self.assertListEqual(self.lista_operatori.get_lista_operatori(), [])
    self.lista_operatori.aggiungi_operatore(self.operatore)
    self.lista_operatori.rimuovi_operatore_by_id("mariaferrari")
    self.assertListEqual(self.lista_operatori.get_lista_operatori(), [])
```

CONCLUSIONE

Il sistema è stato progettato con lo scopo di poter gestire in maniera efficiente i ricoveri all'interno di una struttura ospedaliera.

In aggiunta il sistema potrà interfacciarsi con altri sistemi che ne faciliteranno le funzionalità.

NOTE

Le *credenziali di accesso* per il nostro software sono:

Amministratore dell'ufficio di accettazione:

username: ammacc

password: 1

Amministratore del pronto soccorso:

username: ammps

password: 1

Infermiere:

username: infermiere

password: 1

Medico:

username: medico

password: 1