Università degli Studi di Modena e Reggio Emilia
Dipartimento di Ingegneria "Enzo Ferrari" di Modena
Corso di laurea in Ingegneria Informatica (L.270/04)

Progettazione e Sviluppo di un'Applicazione Web per la gestione di servizi di Homecare

Candidato: Enrico Bazzani

Relatrice: Sonia Bergamaschi

Anno Accademico 2015/2016

Abstract

E-Health è un termine che identifica il complesso delle risorse, soluzioni e tecnologie informatiche applicate alla salute ed alla sanità.

Le tecnologie sviluppate nell'ambito dell'E-Health trovano una importante applicazione nei servizi di Homecare, ovvero il supporto ed il trattamento del paziente a domicilio attraverso un costante monitoraggio garantito da sensori indossabili e la comunicazione on-demand con professionisti come medici, infermieri e psicologi.

La presente tesi, svolta nell'ambito di uno stage presso l'azienda DataRiver S.R.L., descrive l'intera fase di progetto e di realizzazione di una piattaforma web per la gestione e l'ottimizzazione dei servizi di Homecare, in grado di efficentare i processi informativi e fornire un miglior supporto per Medici, Operatori e Pazienti.

In particolare, la soluzione presentata è stata realizzata come Web Application, e progettata per essere utilizzata da utenti con ruoli e compiti differenti.
Gli Operatori sanitari, attraverso i dispositivi mobili in loro dotazione, possono compilare la scheda del Paziente durante la visita domiciliare.
I Manager, che coordinano le visite degli Operatori, hanno a disposizione un Calendario attraverso il quale visualizzano e modificano le visite in programma.
I Medici possono contattare direttamente i Pazienti attraverso la Video Visita in tempo reale per verificarne lo stato di salute.

La tesi descrive il contesto in cui si colloca l'Applicazione, definisce analiticamente i requisiti formalizzati in fase di progettazione, e descrive la metodologia di lavoro seguita dal team di sviluppo per garantire il rispetto dei requisiti ed il rilascio delle diverse versioni dell'Applicazione nei tempi previsti.

Sono poi presentate le soluzioni architetturali e logiche adottate, ed in particolare il Framework Vaadin, che si occupa dello scambio dei dati tra le componenti client ed il server, ed il Framework Hibernate, che gestisce la corrispondenza tra dati sul Database e classi Java.

Infine, vegnono approfondite l'architettura e la logica dei componenti software principali, realizzati attraverso pannelli ottimizzati per la navigazione da dispositivi mobili.
Il Calendario, attraverso cui i Manager possono pianificare le Visite e filtrare quelle presenti per Programma, Paziente ed Operatore, permette l'inserimento di Attività Ripetute, ovvero la generazione di una serie di visite per un Paziente specificando regole di ripetizione articolate, che offrono livelli di flessibilità paragonabili alle soluzioni consumer più diffuse.
La Video Visita, implementazione di un software conforme allo standard HIPAA per il trattamento dei dati sensibili dei Pazienti, offre una qualità congrua a realizzare brevi colloqui tra Medico e Paziente anche in caso di connessione non ottimale.
La gestione delle Anagrafiche permette di inserire sul Database e modificare i dati di Pazienti, Programmi Assistenziali, Farmacie, Centri Clinici e Operatori Sanitari, e di associare questi ultimi a specifici Pazienti e Programmi.
Le gestione delle Schede permette di generare dinamicamente le schede del Paziente sulla base delle caratteristiche del singolo Programma Assistenziale.

Indice

Abstract 1
Indice 2
1. Introduzione 5
1.1. Tecnologia Mobile nella Medicina 6
1.2. Tutela della Privacy 6
2. Specifica dei Requisiti dell'Applicazione 10
2.1. Scopo dell'Applicazione 10
2.2. Utenti e Ruoli 11
2.3. Requisiti Funzionali 11
2.3.1. Gestione di Utenti e Ruoli 11
2.3.2. Gestione di Anagrafica, Centro Clinico, Farmacia e Paziente 13
2.3.3. Gestione dei Programmi 17
2.3.4. Gestione Visite 20
2.3.5. Scheda Presa in Carico 23
2.3.6. Scheda Visita 26
2.3.7. Video Visita 27
2.3.8. Gestione Trasferte 28
2.4. Requisiti non Funzionali 30
2.4.1. Usabilità 30
2.4.2. Sicurezza 30
2.4.3. Compliance 30
2.4.4. Disaster recovery 31
2.4.5. Backup 31
3. Organizzazione del Lavoro 32
3.1. Metodologia di Sviluppo 32
3.1.1. Agile 32
3.1.2. DevOps 32
3.2. Gantt 33
3.3. Milestones e Demo 34
3.3.1. Milestone 1.1 34
3.3.2. Milestone 1.2 34

3.3.3. Milestone 1.3 35
3.3.4. Versioni successive 35
4. Sistema Informativo 38
4.1. Tecnologie Implementative 38
4.1.1. Vaadin 38
4.1.2. Hibernate 39
4.1.2.1. Entity 39
4.1.2.2. Criteria Query 40
4.1.2.3. HQL Query 41
4.1.2.4. SQL Query 42
4.1.2.5. Prestazioni di Criteria, HQL e SQL Query 42
4.2. Architettura 47
4.2.1. Data Logic 48
4.2.1.1. Database 49
4.2.1.2. Data Access Object 50
4.2.2. Data Integration 51
4.2.2.1. Service 51
4.2.2.2. Data Provider 51
4.2.3. User Interface 51
4.2.3.1. Listener 51
4.2.3.2. Pannelli e Finestre 52
4.2.3.3. Responsiveness 52
4.3. Ruoli e Permessi 53
4.3.1. Role-Based Access Control 53
4.3.1.1. Architettura del Role-Based Access Control 54
4.3.2. Attribute-Based Access Control 54
4.3.2.1. Architettura dell'Attribute-Based Access Control 54
4.3.3. Logica dei Ruoli e Permessi 55
5. Componenti 57
5.1. Calendario 57
5.1.1. Tipologie e Visibilità delle Attività 57
5.1.2. Navigazione del Calendario 58

5.1.3. Filtri del Calendario 59
5.1.4. Attività 59
5.1.4.1. Architettura delle Attività 61
5.1.4.2. Logica delle Attività 62
5.1.4.3. Geolocalizzazione delle Attività 63
5.1.5. Attività Ripetute 64
5.1.5.1. Architettura delle Attività Ripetute 65
5.1.5.2. Logica delle Attività Ripetute 67
5.1.5.3. Materializzazione delle Attività Ripetute 68
5.1.5.4. Esempio di Funzionamento delle Attività Ripetute 70
5.1.5.5. Prestazioni delle Attività Ripetute 72
5.2. Video Visita 75
5.3. Anagrafiche 77
5.3.1. Programma 78
5.3.2. Paziente 80
5.3.3. Farmacia 83
5.3.4. Centro Clinico 85
5.3.5. Layout Indirizzi 87
5.3.5.1. Architettura degli Indirizzi 89
5.3.5.2. Logica degli Indirizzi 90
5.3.6. Layout Numeri di Telefono 91
5.3.6.1. Architettura dei Numeri di Telefono 91
5.4. Scheda Visita 92
5.4.1. Architettura della Scheda Visita 94
5.4.2. Logica della Scheda Visita 98
6. Conclusioni 100
Bibliografia 101

1. Introduzione

La presente tesi è stata svolta nell'ambito di uno stage semestrale presso l'azienda DataRiver S.R.L., nell'ambito della progettazione e della realizzazione di una piattaforma web per la gestione e l'ottimizzazione dei servizi di Homecare.

Il software, la cui architettura funzionale di alto livello è presentata in Figura 1, è stato commissionato da un'azienda che eroga servizi di assistenza domiciliare per pazienti con malattie rare in tutta Italia, con la prospettiva di allargare il proprio bacino di utenza ad altri Paesi europei, a partire dalla Spagna.

Ho personalmente preso parte a tutte le fasi del progetto, a partire dalla specifica dei requisiti insieme al cliente e dalla scelta delle tecnologie implementative migliori per garantire il risultato concordato nei tempi stabiliti.

Rispetto al progetto nel suo complesso, mi sono occupato principalmente dello sviluppo e dell'ottimizzazione dei componenti software relative al Calendario, alla gestione delle Attività e delle loro Ripetizioni, e alle Anagrafiche di Programmi, Pazienti, Farmacie e Centri Clinici. Le parti del progetto relative alle Schede Visita, alla gestione di Utenti e Ruoli e all'implementazione della Video Visita sono state realizzate da dipendenti dell'azienda.

Figura 1.1 - Componenti dell'Applicazione (in blu quelle su cui ho lavorato personalmente)

Il progetto è, al momento della redazione della tesi, in fase di ultimazione, con un prototipo già funzionante a disposizione del cliente per la sperimentazione.

Il corpo del presente elaborato è così strutturato:

- Nel Capitolo 2 è definito l'elenco dettagliato dei requisiti, funzionali e non funzionali, maturati nella fase iniziale del progetto;
- Nel Capitolo 3 è descritta l'organizzazione del lavoro del team di sviluppo, che ci ha permesso di soddisfare gran parte dei requisiti nei miei sei mesi di permanenza in azienda;
- Nel Capitolo 4 sono presentate l'architettura del progetto, ovvero la sua divisone logica in strati, e le tecnologie utilizzate per lo sviluppo;
- Nel Capitolo 5 sono analizzate le componenti principali che compongono il progetto: il Calendario, la Video Visita, le Anagrafiche e la Scheda Visita.

1.1. Tecnologia Mobile nella Medicina

Le risorse, soluzioni e tecnologie informatiche applicate alla salute ed alla sanità rappresentano un'importante opportunità per ririduRre i costi, i tempi e le risorse impiegate per erogare i servizi assistenziali, e rientrano nella definizione generica di "E-Health". In particolare, l'E-Health ha un impatto su:

- Gestione dei Dati del paziente, nella forma della Cartella Clinica Elettronica, che ne permette una comunicazione semplice tra le diverse figure professionali;
- Telemedicina, ovvero tutti i tipi di cure svolti in maniera telematica tra medico e paziente. La Telemedicina comporta un ingente risparmio di tempo e risorse da parte del paziente, e contestualmente permette al medico di estendere il proprio bacino d'utenza ed avere contatti più frequenti con i pazienti.
- Monitoraggio Costante del paziente, attraverso dispositivi indossabili dotati di sensori biometrici, che registrano dati come il battito cardiaco, pressione del sangue, temperatura corporea e numero di passi. Sulla base dei dati raccolti possono essere generati report che aiutano ad identificare precocemente l'insorgere di malattie o disturbi.

Attraverso le diverse declinazioni dell'E-Health è possibile dunque passare da un sistema centralizzato, che ruota attorno ai professionisti ed alle strutture cliniche, ad un sistema che pone il paziente al centro e razionalizza costi e tempi, facendo della prevenzione e della identificazione precoce delle anomalie il proprio cardine.

1.2. Tutela della Privacy

L'HIPAA (Health Insurance Portability and Accountability Act) è una legge statunitense che regolamenta gli standard per la protezione dei dati clinici del paziente, ed è utilizzato come standard anche al di fuori degli Stati Uniti.

Lo standard definisce quattro regole:

- Privacy Rule
- Security Rule
- Enforcement Rule

- Breach Notification Rule

Nella progettazione di una applicazione che gestisce dati clinici (*Protected Health Information, PHI*) è necessario in particolare rispettare la Privacy Rule e la Security Rule.

La Privacy Rule impone a chi gestisce i dati clinici di:

- Non permettere nessun tipo di uso non ammesso o divulgazione di PHI;
- Fornire notifica delle violazioni al Possessore dei dati;
- Permettere l'accesso ai dati sia all'individuo che al Possessore dei dati;
- Permettere l'accesso ai dati al Governo, se richiesto;
- Fornire un rendiconto delle divulgazioni;
- Soddisfare le condizioni della Security Rule.

La Security Rule è a sua volta divisa in tre sezioni, che contengono specifiche richieste (*required, R*) e opzionali (*addressable, A*), ovvero da implementare solo nel caso in cui lo si ritenga appropriato:

- Technical Safeguards:
    - Access Control - Unique User Identification (R): Assegnare un identificativo univoco a ciascun utente per permetterne l'identificazione ed il monitoraggio;
    - Access Control - Emergency Access Procedure (R): Stabilire le procedure per ottenere i PHI durante un'emergenza;
    - Access Control - Automatic Logoff (A): Implementare procedure elettroniche che terminano una sessione dopo un tempo predeterminato di inattività;
    - Access Control - Encryption and Decryption (A): Implementare un meccanismo per crittografare e decrittografare PHI;
    - Audit Controls (R): implementare meccanismi procedurali hardware o software che registrano ed esaminare l'attività nei sistemi informativi che utilizzano PHI;
    - Integrity - Mechanism to Authenticate ePHI (A): Implementare meccanismi elettronici per verificare che le PHI non siano state alterate o distrutte in modo non autorizzato;
    - Authentication (R): Implementazione di procedure per verificare se una persona o entità che richiede l'accesso alle PHI corrisponde alle sue credenziali;
    - Transmission Security - Integrity Controls (A): Implementare le misure di sicurezza per garantire che le PHI trasmesse per via elettronica non siano impropriamente modificate;
    - Transmission Security - Encryption (A): Implementare un meccanismo per cifrare le PHI.
- Physical Safeguards:
    - Facility Access Controls - Contingency Operations (A): Stabilire le procedure che consentono l'accesso ed il ripristino dei dati perduti in caso di emergenza;

- Facility Access Controls - Facility Security Plan (A): Stabilire e attuare procedure per salvaguardare la struttura e le attrezzature da accesso fisico non autorizzato, manomissione e furto;
- Facility Access Controls - Access Control and Validation Procedures (A): Attuazione di meccanismi di controllo e di convalida dell'accesso delle persone alle strutture in base al loro ruolo o funzione, controllo dei visitatori, e il controllo di accesso ai software per il test e la revisione;
- Facility Access Controls - Maintenance Records (A): Attuazione di politiche e procedure per documentare le riparazioni e modifiche ai componenti fisici di un impianto che sono legati alla sicurezza (ad esempio hardware, pareti, porte, e serrature);
- Workstation Use (R): Attuare politiche e procedure che specificano le funzioni da svolgere, il modo in cui queste funzioni devono essere eseguite, e gli attrezzi fisici di una postazione di lavoro o di un gruppo di postazioni di lavoro che può accedere alle PHI;
- Workstation Security (R): Implementare misure di sicurezza fisiche per tutte le stazioni di lavoro che accedono alle PHI, per limitare l'accesso agli utenti autorizzati;
- Device and Media Controls - Disposal (R): Attuare le politiche e le procedure per affrontare la distruzione delle PHI, e / o l'hardware o supporti elettronici su cui sono memorizzato;
- Device and Media Controls - Media Re-Use (R): Implementare le procedure per la rimozione di PHI dai media elettronici prima che i media siano resi disponibili per il riutilizzo;
- Device and Media Controls - Accountability (A): Mantenere un registro dei movimenti di hardware e media elettronici e delle persone responsabili;
- Device and Media Controls - Data Backup and Storage (A): Creare una copia esatta e recuperabile delle PHI prima dello spostamento o della modifica delle apparecchiature.

- Administrative Safeguards:
  - Security Management Process - Risk Analysis (R): Eseguire e documentare un'analisi di rischio per verificare dove i PHI vengono utilizzati e conservati al fine di determinare tutte le modalità con cui i dati potrebbero essere violati;
  - Security Management Process - Risk Management (R): Implementare misure sufficienti per ridurre tali rischi ad un livello adeguato;
  - Security Management Process - Sanction Policy (R): Implementare politiche di sanzione per i dipendenti che non rispettano le misure di sicurezza;
  - Security Management Process - Information Systems Activity Reviews (R): Monitorare regolarmente l'attività del sistema, ed i log;

- Assigned Security Responsibility - Officers (R): Designare Ufficiali di Sicurezza e Privacy HIPAA;
- Workforce Security - Employee Oversight (A): Attuare le procedure per autorizzare e supervisionare i dipendenti che lavorano con PHI, e per la concessione e la rimozione dell'accesso alle PHI. Assicurarsi che l'accesso di un dipendente alle PHI si concluda con il suo licenziamento;
- Information Access Management - Multiple Organizations (R): Assicurarsi che le PHI non siano accessibili da partner o subappaltatori non autorizzati all'accesso;
- Information Access Management - ePHI Access (A): Attuare le procedure per la concessione dell'accesso e della documentazione dell'accesso ai dati PHI o a sitemi e servizi che permettono l'accesso ai dati PHI;
- Security Awareness and Training - Security Reminders (A): Inviare periodicamente aggiornamenti e promemoria sulle politiche di sicurezza e privacy ai dipendenti;
- Security Awareness and Training - Protection Against Malware (A): Disporre di procedure per il rilevamento e la segnalazione di software dannoso;
- Security Awareness and Training - Login Monitoring (A): Predisporre un monitoraggio degli accessi ai sistemi e la segnalazione di discrepanze;
- Security Awareness and Training - Password Management (A): Assicurarsi che ci siano procedure per la creazione, la modifica e la protezione delle password;
- Security Incident Procedures - Response and Reporting (R): Identificare, documentare e rispondere agli incidenti di sicurezza;
- Contingency Plan - Contingency Plans (R): Assicurarsi che ci siano backup accessibili dei dati PHI e che ci siano procedure per ripristinare i dati persi;
- Contingency Plan - Contingency Plans Updates and Analysis (A): Disporre di procedure per la verifica periodica e la revisione dei piani di emergenza;
- Contingency Plan - Emergency Mode (R): Stabilire le procedure per consentire la continuazione dei processi aziendali critici per la protezione della sicurezza dei dati PHI durante le emergenze;
- Evaluations (R): Eseguire valutazioni periodiche per verificare se eventuali cambiamenti del business dell'azienda o della legge richiedano modifiche alle procedure di conformità HIPAA;
- Business Associate Agreements (R): Definire contratti speciali con i partner commerciali che hanno accesso alle PHI al fine di garantire che siano conformi. Scegliere i partner che hanno accordi simili con eventuali partner a cui a loro volta estenderanno l'accesso.

2. Specifica dei Requisiti dell'Applicazione

2.1. Scopo dell'Applicazione

È stata progettata e realizzata una piattaforma per la gestione e l'ottimizzazione dei servizi di Homecare. In particolare, è stata realizzata un'Applicazione Web multilingua e utilizzabile da PC e dispositivi mobili, predisposta alla raccolta automatica e alla condivisione di dati clinici, che garantisce il rispetto delle normative vigenti.

Il sistema progettato semplifica il lavoro del personale coinvolto nei programmi assistenziali (medici, infermieri, manager del programma) permettendo l'ottimizzazione dei servizi sanitari e dei percorsi clinici del Paziente da remoto.

La piattaforma software per la gestione dei programmi assistenziali è progettata per automatizzare ed ottimizzare i diversi processi critici dei programmi: l'erogazione delle Attività, il monitoraggio dei Pazienti ed il controllo della qualità del servizio fornito.

La piattaforma è progettata in modo da rendere possibile la raccolta automatica e la gestione di dati fisiologici del Paziente provenienti da dispositivi indossabili, nel rispetto delle normative vigenti in termini legali, di sicurezza, di privacy ed etici.

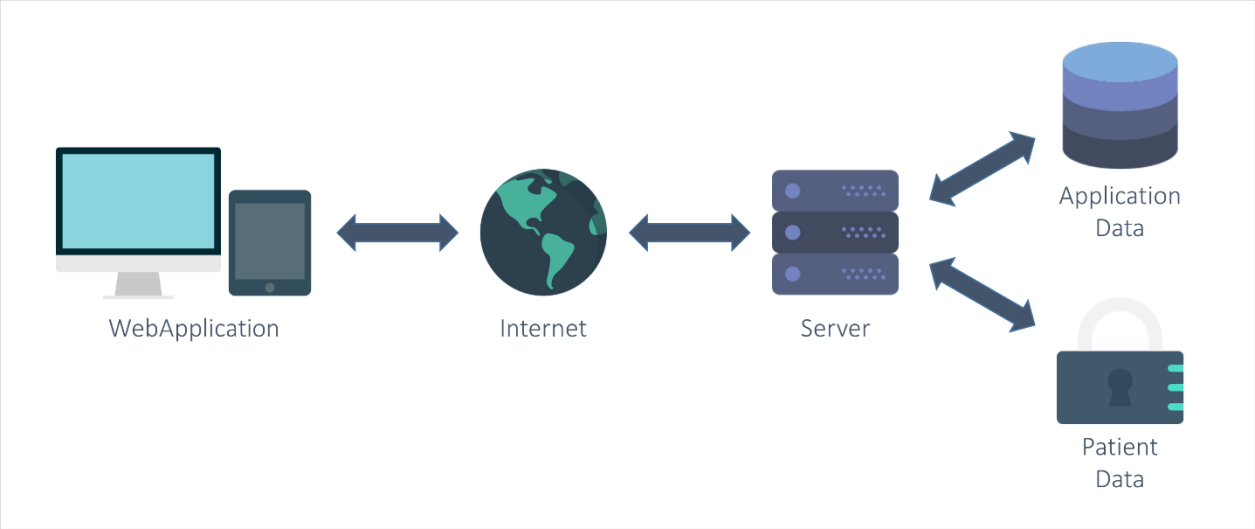


Figura 2.1.1 – Architettura dell'Applicazione e Gestione dei Dati

Le principali tipologie di dati raccolti dall'applicazione Web sono due:

- Dati sensibili dei Pazienti (nome, cognome, data di nascita). Sono memorizzati in un'istanza di Database creata in una porzione di disco opportunatamente criptato;
- Dati clinici dei Pazienti. Sono memorizzati in un'istanza di Database separata da quella dei dati sensibili.

Per offrire un adeguato livello di sicurezza sui dati raccolti, l'applicazione Web è accessibile attraverso il protocollo HTTPS, gli Utenti devono autenticarsi tramite le proprie credenziali e possono vedere e modificare solo la porzione dei dati sui cui hanno l'autorizzazione per garantire il rispetto delle norme vigenti in termini legali, privacy ed etici.

2.2. Utenti e Ruoli

All'interno dell'applicazione sono modellati i seguenti Ruoli:

1. Admin: Ha accesso a tutti i Programmi, i Pazienti e a tutti gli Utenti con qualsiasi Ruolo;
2. Program Manager (PM): Ha accesso a tutte le funzionalità dell'Applicazione. Si occupa anche della creazione dell'Anagrafica dei Centri Clinici, del Medico Specialista e del Pharmacist;
3. Territory Manager (TM): Definsce il Calendario delle visite dei Pazienti del proprio territorio in funzione del PSP. Può decidere quali Operatori assegnare alle visite e riprogrammarle in casi di criticità. Il TM deve poter verificare che la visita sia avvenuta correttamente e sia stata compilata correttamente la relativa Scheda.
4. Operatore (OP): l'Operatore deve poter visualizzare e/o modificare il proprio Calendario, su cui sono presenti le visite che deve effettuare, in funzione del PSP. L'OP effettua la video visita con il Medico Specialista, compila la relativa Scheda e inserisce eventuali note.
5. Contact Center (CC): il Contact Center si occupa dell'inserimento dell'Anagrafica del Paziente nel PSP e gestisce le richieste di adesione dei Pazienti. Deve poter visualizzare e modificare il Calendario dell'Operatore in funzione del PSP. Può visualizzare spese ed eventuali annotazioni dell'Operatore. Il Contact Center si occupa della logistica del trasporto del farmaco.
6. Program Doctor (PD): il Program Doctor deve poter visualizzare tutti i dati relativi ai Pazienti del PSP e visualizzare il Calendario delle visite.
7. Medico Specialista (MS): il Medico Specialista può visualizzare i dati relativi ai suoi Pazienti. Deve poter fissare l'appuntamento per la Video Visita con l'Operatore.
8. Pharmacist (PH): il Pharmacist può visualizzare una porzione dei dati relativi ai Pazienti di interesse.

2.3. Requisiti Funzionali

2.3.1. Gestione di Utenti e Ruoli

INSERIMENTO UTENTI AUTORIZZATI	
INTRODUZIONE	Attori coinvolti Admin

	Descrizione generale della funzione L'Admin registra gli Utenti inserendo tutti i dati necessari e gli assegna il Ruolo.
INPUT	Descrizione generica dei dati Dati Utente: email, nome, cognome, Ruolo, telefono
DESCRIZIONE	Validazione dei dati Sono obbligatori email, nome, cognome, telefono e Ruolo
	Sequenza di operazioni L'Admin accede al sistema con la sua email, attiva la funzione inserimento nuovo Utente, compila i vari campi e salva i dati.
	Risposta ad eventuali anomalie • Messaggio di errore se non vengono inseriti i dati obbligatori, il sistema comunica che i campi sono obbligatori. • Messaggio di errore se l'Utente esiste già (email esistente).
OUTPUT	Se non ci sono anomalie l'Utente viene invitato tramite email ad accedere al sistema.

MODIFICA UTENTI AUTORIZZATI	
INTRODUZIONE	Attori coinvolti Admin
	Descrizione generale della funzione L'Admin accede al sistema per modificare i dati degli Utenti precedentemente inseriti.
INPUT	Descrizione generica dei dati Dati dell'Utente precedentemente inseriti
DESCRIZIONE	Sequenza di operazioni Modifica dati L'Admin accede al sistema con il suo identificativo, sceglie dall'elenco l'Utente, attiva la funzione modifica Utente, modifica e salva i dati.

	Modifica Ruolo L'Admin accede al sistema con il suo identificativo, sceglie dall'elenco l'Utente, attiva la funzione modifica Utente, sceglie e salva il nuovo Ruolo.
OUTPUT	I dati dell'Utente vengono modificati.

CANCELLAZIONE UTENTE	
INTRODUZIONE	Attori coinvolti Admin
	Descrizione generale della funzione L'Admin accede al sistema per cancellare Utenti precedentemente inseriti
INPUT	Descrizione generica dei dati Utente selezionato
DESCRIZIONE	Sequenza di operazioni L'Admin accede al sistema con il suo identificativo, sceglie dall'elenco l'Utente e gli viene chiesta conferma della cancellazione. In caso di risposta affermativa l'Utente viene cancellato altrimenti l'operazione viene annullata.
	Risposta ad eventuali anomalie • Messaggio di errore generico se non si riesce a cancellare l'Utente a causa di problemi di comunicazione con il server o problemi di accesso al database.
OUTPUT	Se non ci sono anomalie l'Utente viene cancellato. Si tratta di una "soft delete": il dato non è più accessibile dagli Utenti, ma lo è dal sistemista autorizzato, che se necessario può recuperare il dato eliminato.

2.3.2. Gestione di Anagrafica, Centro Clinico, Farmacia e Paziente

CREAZIONE ANAGRAFICA CENTRO CLINICO

INTRODUZIONE	Attori coinvolti Program Manager
	Descrizione generale della funzione Program Manager inserisce l'Anagrafica del Centro Clinico
INPUT	Descrizione generica dei dati Dati del Centro Clinico: nome ospedale, indirizzo, nome reparto, numero di telefono, fax, email, associazione con Medico Specialista
DESCRIZIONE	Validazione dei dati Il nome del Centro Clinico e del reparto è un campo obbligatorio
	Sequenza di operazioni Il Program Manager accede all'applicazione e inserisce l'Anagrafica del Centro Clinico
	Risposta ad eventuali anomalie • Messaggio di errore nel caso non vengano inseriti i campi obbligatori
OUTPUT	Se non ci sono anomalie l'Anagrafica del Centro Clinico viene inserito all'interno del software Si attivano delle notifiche per informare altri ruoli direttamente coinvolti nel flusso delle Attività (es. nel fabry@home l'apertura di una nuova Anagrafica di Centro Clinico è comunicata a TM e PD)

CREAZIONE ANAGRAFICA FARMACIA	
INTRODUZIONE	Attori coinvolti Program Manager
	Descrizione generale della funzione Program Manager inserisce l'Anagrafica della Farmacia
INPUT	Descrizione generica dei dati Dati Farmacia: nome, indirizzo, numero di telefono, fax, email, Pharmacist di riferimento, note su modalità di eventuali remind (commento)

DESCRIZIONE	Validazione dei dati Il nome della Farmacia e i contatti sono campi obbligatori
	Sequenza di operazioni Program Manager accede all'applicazione e inserisce l'Anagrafica della Farmacia
	Risposta ad eventuali anomalie • Messaggio di errore nel caso non vengano inseriti i campi obbligatori
OUTPUT	Se non ci sono anomalie l'Anagrafica della Farmacia viene inserita all'interno del software Si attivano delle notifiche per informare altri ruoli direttamente coinvolti nel flusso delle Attività (es. nel fabry@home l'apertura di una nuova Anagrafica di Farmacia è comunicata a CC e TM)

CREAZIONE ANAGRAFICA PAZIENTE	
INTRODUZIONE	Attori coinvolti PD (in via preferenziale) e PM sono gli Utenti autorizzati
	Descrizione generale della funzione Gli Utenti autorizzati inseriscono l'Anagrafica del Paziente
INPUT	Descrizione generica dei dati Dati Paziente: nome, Cognome, Data di nascita, Indirizzo, Numero di telefono, Email, CF
DESCRIZIONE	Validazione dei dati CF, Nome, Cognome e Data di nascita, indirizzo, numero di telefono sono campi obbligatori
	Sequenza di operazioni L'Utente autorizzato accede all'applicazione e inserisce l'Anagrafica del Paziente.

	Risposta ad eventuali anomalie • Messaggio di errore nel caso non vengano inseriti i campi obbligatori • Check per evitare duplicazioni di pazienti (mediante CF)
OUTPUT	Se non ci sono anomalie l'Anagrafica del Paziente viene inserita all'interno del software

VISUALIZZAZIONE DATI PAZIENTE	
INTRODUZIONE	Attori coinvolti Program Manager, Contact Center, Program Doctor, TM, Operatore sono gli Utenti autorizzati Gli attori possono visualizzare i dati in funzione delle permission a loro concesse (es. gli Operatore potranno vedere solo i dati dei pazienti a loro assegnati)
	Descrizione generale della funzione L'Utente autorizzato può visualizzare tutti i dati dei pazienti (schede, pdf, Calendario)
INPUT	Descrizione generica dei dati Paziente di cui visualizzare i dati
DESCRIZIONE	Sequenza di operazioni L'Utente autorizzato accede all'applicazione e seleziona il Paziente di cui vuole visualizzare i dati. Tutti i dati memorizzati nel sistema sono visualizzabili nel formato in cui sono stati inseriti: schede, pdf, Calendario visite, video delle visite.
OUTPUT	Se non ci sono anomalie l'Utente riesce a visualizzare tutti i dati relativi al Paziente

VISUALIZZAZIONE SINTETICA DATI PAZIENTE	
INTRODUZIONE	Attori coinvolti Medico Specialista, Pharmacist

	Descrizione generale della funzione Il Medico Specialista, Pharmacist accede a una porzione dei dati dei propri Pazienti salvati nel sistema
INPUT	Descrizione generica dei dati Paziente di cui visualizzare i dati
DESCRIZIONE	Sequenza di operazioni Il Medico Specialista/Pharmacist accede all'applicazione e seleziona il Paziente di cui vuole visualizzare i dati. Una porzione dei dati memorizzati nel sistema sono visualizzabili nel formato in cui sono stati inseriti: schede con dati clinici, pdf in fasi in cui è coinvolto (es. adesione al programma e presa in carico) e video delle visite.
OUTPUT	Se non ci sono anomalie il Medico Specialista/Pharmacist riesce a visualizzare tutti i dati relativi al Paziente

2.3.3. Gestione dei Programmi

ATTIVAZIONE DEL PSP PER UN CENTRO CLINICO	
INTRODUZIONE	Attori coinvolti Program Manager
	Descrizione generale della funzione Program Manager attiva un PSP per un Centro Clinico (un reparto/medico)
INPUT	Descrizione generica dei dati PSP (già presente nel sistema), Centro Clinico
DESCRIZIONE	Validazione dei dati PSP e Centro Clinico sono due dati obbligatori
	Sequenza di operazioni Program Manager accede all'applicazione e seleziona un Centro Clinico. Successivamente Program Manager sceglie il PSP da attivare per il Centro Clinico e conferma l'operazione.

	Risposta ad eventuali anomalie • Messaggio di errore nel caso non vengano inseriti i campi obbligatori
OUTPUT	Se non ci sono anomalie il PSP viene attivato per il Centro Clinico

AGGIUNTA DEL PAZIENTE A UN PSP	
INTRODUZIONE	Attori coinvolti PD (in via preferenziale) e il PM sono gli Utenti autorizzati
	Descrizione generale della funzione L'Utente autorizzato aggiunge un Paziente al PSP Il PM riceve notifica quando il PD aggiunge i pazienti a un PSP
INPUT	Descrizione generica dei dati Paziente, PSP attivato per il Centro Clinico di riferimento
DESCRIZIONE	Validazione dei dati Paziente e PSP attivato per il Centro Clinico di riferimento sono dati obbligatori
	Sequenza di operazioni L'Utente autorizzato seleziona un Paziente. Successivamente deve scegliere il PSP attivato per il Centro Clinico del Paziente.
	Risposta ad eventuali anomalie • Messaggio di errore nel caso non vengano inseriti i campi obbligatori
OUTPUT	Se non ci sono anomalie il Paziente viene inserito all'interno del PSP

USCITA DEL PAZIENTE DA UN PSP	
INTRODUZIONE	Attori coinvolti Program Manager è l'Utente autorizzato

	Descrizione generale della funzione L'Utente autorizzato all'operazione fa uscire il Paziente dal PSP Il PD, TM, HN, CC, ricevono notifica dell'uscita del Paziente dal PSP
INPUT	Descrizione generica dei dati Paziente, PSP attivato per il Paziente
DESCRIZIONE	Validazione dei dati Paziente e PSP attivato per il Paziente stesso sono dati obbligatori
	Sequenza di operazioni L'Utente autorizzato seleziona un Paziente. Successivamente deve scegliere il PSP attivato per il Paziente.
	Risposta ad eventuali anomalie • Messaggio di errore nel caso non vengano specificati i campi obbligatori
OUTPUT	Se non ci sono anomalie il Paziente non avrà più attivato il PSP di riferimento. Tutti i dati precedentemente inseriti all'uscita del Paziente verranno archiviati nel rispetto della privacy, tuttavia non sarà più possibile inserire nuovi dati.

CARICAMENTO PDF INSERIMENTO PAZIENTE IN PROGRAMMA	
INTRODUZIONE	Attori coinvolti PD è l'Utente autorizzato
	Descrizione generale della funzione L'Utente carica nel sistema il/i PDF per l'adesione e attivazione del programma per un Paziente (es. consenso informato, richiesta attivazione programma, Scheda inserimento Paziente in Fabry@home) PM, TM, CC ricevono notifica quando il caricamento viene effettuato da PD
INPUT	Descrizione generica dei dati PDF per l'adesione e attivazione del programma per un Paziente

DESCRIZIONE	Sequenza di operazioni L'Utente attiva la funzione per caricare i PDF per l'adesione e attivazione del programma per un Paziente; seleziona i file PDF e conferma l'operazione
OUTPUT	Vengono caricati i PDF di interesse nel sistema

ELIMINAZIONE PDF INSERIMENTO PAZIENTE IN PROGRAMMA	
INTRODUZIONE	Attori coinvolti PD è l'Utente autorizzato. La cancellazione deve essere validata/accettata da PM.
	Descrizione generale della funzione L'Utente elimina dal sistema uno dei PDF per l'adesione e attivazione del programma per un Paziente (es. consenso informato, richiesta attivazione programma, Scheda inserimento Paziente in Fabry@home) PM, TM, CC ricevono notifica quando la cancellazione è stata effettuata
INPUT	Descrizione generica dei dati Uno dei pdf già caricati in precedenza
DESCRIZIONE	Sequenza di operazioni L'Utente autorizzato accede al sistema, seleziona uno dei PDF caricati in precedenza e attiva la funzione di eliminazione; alla comparsa della conferma dell'operazione, l'Utente può decidere se eliminare il file o annullare l'operazione.
OUTPUT	Il PDF non viene più visualizzato nel sistema NOTA: dei PDF eliminati viene comunque tenuta traccia

2.3.4. Gestione Visite

INSERIMENTO VISITA PAZIENTE	
INTRODUZIONE	Attori coinvolti

	Program Manager (nel caso della presa in carico), PD (nel caso della prima visita domiciliare) , Operatore (infermiere) (per tutte le infusioni successive alla prima visita domiciliare) sono gli Utenti autorizzati
	<u>Descrizione generale della funzione</u> L'Utente autorizzato inserisce la visita di un Paziente PD e TM ricevono una notifica quando il PM inserisce visita di presa in carico; TM, PM, HN e CC ricevono una notifica quando il PD inserisce la prima visita domiciliare; PD, TM e CC ricevono una notifica quando l'HN modifica visite successive alla prima (che sono proposte dal sistema)
INPUT	<u>Descrizione generica dei dati</u> Paziente, Operatore che farà la visita, Data della visita, Note, Medico Caregiving, CC (per la disponibilità del farmaco)
DESCRIZIONE	<u>Validazione dei dati</u> Paziente, Operatore e data della visita sono cambi obbligatori
	<u>Sequenza di operazioni</u> L'Utente autorizzato accede al sistema, accede al Calendario; vedendo e potendo creare gli eventi per i pazienti a lui associati. Sceglie una data e seleziona Operatore (o manualmente o selezionando un Operatore proposto dal sistema) e Pazienti coinvolti nella visita. Se richiesto sarà visualizzata/notificata la richiesta di da parte del medico di effettuare la video visita con un Medico
	<u>Risposta ad eventuali anomalie</u> • Messaggio di errore nel caso non vengano inseriti i campi obbligatori
OUTPUT	Se non ci sono anomalie viene creata la visita di un Paziente con specificato giorno e Operatore della visita stessa

MODIFICA VISITA PAZIENTE	
INTRODUZIONE	<u>Attori coinvolti</u> Modifica visita: Operatore, PD e TM sono gli Utenti autorizzati a eseguire l'operazione. Chi non l'effettua direttamente è comunque informato. Il

	cambio deve essere validato/accettato da PD e Operatore (sono entrambi necessari). Modifica Operatore: TM è l'Utente autorizzato a eseguire l'operazione. PD, Operatore nuovo e vecchio sono informati/ricevono una notifica
	Descrizione generale della funzione L'Utente autorizzato modifica la visita di un Paziente e può modificare l'Operatore in caso di non disponibilità (sia inserendolo manualmente sia scegliendo tra i suggerimenti del sistema)
INPUT	Descrizione generica dei dati Modifica visita: Operatore, Data della visita, Note, Medico Caregiving
DESCRIZIONE	Validazione dei dati l'Operatore e la data della visita sono cambi obbligatori
	Sequenza di operazioni L'Utente autorizzato accede al sistema, accede al Calendario; sceglie una visita in Calendario di un Paziente e modifica i campi della visita e/o dell'Operatore. Può filtrare le Attività per Programma, Visita e Paziente
	Risposta ad eventuali anomalie • Messaggio di errore nel caso non vengano inseriti i campi obbligatori
OUTPUT	Se non ci sono anomalie vengono modificati i dati della visita

ELIMINAZIONE VISITA PAZIENTE	
INTRODUZIONE	Attori coinvolti PM, PD, sono gli Utenti autorizzati a eseguire l'operazione
	Descrizione generale della funzione L'Utente autorizzato può eliminare: una visita/ un periodo di visite/ visite fino a data da stabilirsi. PM, PD, TM, CC e HN ricevono notifiche quando l'eliminazione viene fatta
INPUT	Descrizione generica dei dati

	Visita presente in Calendario
DESCRIZIONE	<u>Sequenza di operazioni</u> L'Utente autorizzato accede al sistema, accede al Calendario; sceglie una visita in Calendario di un Paziente e richiede l'eliminazione. Compare un messaggio di conferma di eliminazione; se l'Utente conferma l'eliminazione, la visita viene eliminata, altrimenti l'operazione è annullata. E' necessario inserire il motivo dell'annullamento.
	<u>Risposta ad eventuali anomalie</u> • Messaggio di errore nel caso non vengano inseriti i campi obbligatori
OUTPUT	Se non ci sono anomalie la visita viene eliminata

2.3.5. Scheda Presa in Carico

COMPILAZIONE SCHEDA PER LA PRESA IN CARICO	
INTRODUZIONE	<u>Attori coinvolti</u> PD è l'Utente autorizzato
	<u>Descrizione generale della funzione</u> L'Utente autorizzato compila la Scheda per la presa in carico (es. cartella clinica in fabry@home a esclusione dei questionari) PM, TM, HN e CC ricevono una notifica quando PD compila la presa in carico
INPUT	<u>Descrizione generica dei dati</u> Dati relativi alla presa in carico del Paziente: personale coinvolto nella presa in carico, notizie anamnestiche, segni e sintomi alla diagnosi, terapia concomitante, dati clinici attuali, esame obiettivo, informazioni sulla terapia e sul servizio domiciliare previsto
DESCRIZIONE	<u>Sequenza di operazioni</u> L'Utente autorizzato accede all'applicazione, seleziona il Paziente e attiva la funzione per compilare la Scheda per la presa in carico
OUTPUT	I dati compilati dall'Utente vengono inseriti nel sistema

CARICAMENTO PDF SOMMINISTRAZIONE DEL FARMACO	
INTRODUZIONE	Attori coinvolti Admin/Program Manager, Contact Center, Operatore o Territory Manager sono gli Utenti autorizzati
	Descrizione generale della funzione L'Utente carica nel sistema il PDF riguardo la somministrazione del farmaco (es prescrizione domiciliare in fabry@home).
INPUT	Descrizione generica dei dati PDF sulla somministrazione del farmaco: PDF che viene caricato durante la presa in carico del Paziente
DESCRIZIONE	Sequenza di operazioni L'Utente attiva la funzione per caricare i PDF per la presa in carico; seleziona i file PDF di interesse e conferma l'operazione
OUTPUT	Viene caricato il PDF di interesse nel sistema

ELIMINAZIONE PDF SOMMINISTRAZIONE DEL FARMACO	
INTRODUZIONE	Attori coinvolti Admin/Program Manager, Contact Center, Operatore o Territory Manager sono gli Utenti autorizzati
	Descrizione generale della funzione L'Utente elimina dal sistema il PDF riguardo la somministrazione del farmaco (es. prescrizione domiciliare in Fabry@home)
INPUT	Descrizione generica dei dati PDF già caricato in precedenza
DESCRIZIONE	Sequenza di operazioni L'Utente autorizzato accede al sistema, seleziona il PDF caricato in precedenza e attiva la funzione di eliminazione; alla comparsa della

	conferma dell'operazione, l'Utente può decidere se eliminare il file o annullare l'operazione.
OUTPUT	Il PDF non viene più visualizzato nel sistema

CARICAMENTO PDF QUESTIONARI SULLA PRESA IN CARICO	
INTRODUZIONE	Attori coinvolti Admin/Program Manager, Contact Center, Operatore o Territory Manager sono gli Utenti autorizzati
	Descrizione generale della funzione L'Utente carica nel sistema il/i PDF dei questionari sulla presa in carico del Paziente (es. Attività extradomestiche e Abitudini di vita, Questionario sulla Salute, Breve questionario per la valutazione del dolore (BPI) in Fabry@home)
INPUT	Descrizione generica dei dati PDF dei questionari di interesse per la presa in carico del Paziente
DESCRIZIONE	Sequenza di operazioni L'Utente attiva la funzione per caricare i PDF dei questionari riguardanti la presa in carico del Paziente.
OUTPUT	Vengono caricati i PDF di interesse nel sistema

ELIMINAZIONE PDF QUESTIONARI SULLA PRESA IN CARICO	
INTRODUZIONE	Attori coinvolti Admin/Program Manager o Contact Center sono gli Utenti autorizzati
	Descrizione generale della funzione L'Utente elimina dal sistema uno dei PDF dei questionari sulla presa in carico (Attività extradomestiche e Abitudini di vita, Questionario sulla Salute (EQ-5D) con VAS stato salute, Breve questionario per la valutazione del dolore (BPI) in Fabry@home)

INPUT	Descrizione generica dei dati Uno dei pdf già caricati in precedenza
DESCRIZIONE	Sequenza di operazioni L'Utente autorizzato accede al sistema, seleziona uno dei PDF caricati in precedenza e attiva la funzione di eliminazione; alla comparsa della conferma dell'operazione, l'Utente può decidere se eliminare il file o annullare l'operazione.
OUTPUT	Se non ci sono anomalie il PDF non viene più visualizzato nel sistema

2.3.6. Scheda Visita

COMPILAZIONE SCHEDA VISITA	
INTRODUZIONE	Attori coinvolti HN (ed eventualmente per alcuni campi il PD) sono gli Utenti autorizzati
	Descrizione generale della funzione L'Utente autorizzato compila la Scheda della visita (infusione domiciliare in fabry@home) TM, PD, PM e CC ricevono notifica quando l'Operatore compila la Scheda
INPUT	Descrizione generica dei dati Campi presenti nell'infusione domiciliare di fabry@home
DESCRIZIONE	Sequenza di operazioni L'Utente autorizzato accede all'applicazione, seleziona il Paziente e attiva la funzione per compilare l'infusione domiciliare Dovrà essere reso qualche campo obbligatorio (saranno forniti analizzando le specifiche voci)
OUTPUT	Se non ci sono anomalie i dati compilati dall'Utente vengono inseriti nel sistema

COMPILAZIONE SCHEDA CHIAMATA TELEFONICA

INTRODUZIONE	Attori coinvolti HN è l'Utente autorizzato PD, TM ricevono notifica quando l'Operatore compila la Scheda
	Descrizione generale della funzione L'Utente autorizzato compila la Scheda della chiamata telefonica di una visita
INPUT	Descrizione generica dei dati Campi riguardo la chiamata telefonica (Operatori coinvolti, descrizione chiamata, data chiamata)
DESCRIZIONE	Sequenza di operazioni L'Utente autorizzato accede all'applicazione, seleziona una visita e compila i dati riguardanti la chiamata telefonica.
OUTPUT	Se non ci sono anomalie i dati compilati dall'Utente vengono inseriti nel sistema. In caso di mancata conferma dell'infusione deve partire un ciclo di riprogrammazione della somministrazione

2.3.7. Video Visita

AVVIO VIDEO VISITA	
INTRODUZIONE	Attori coinvolti Operatore, Medico Specialista, eventualmente PD
	Descrizione generale della funzione L'Operatore avvia la video visita con il Medico Specialista (eventualmente con PD)
DESCRIZIONE	Sequenza di operazioni L'Operatore accede all'applicazione e avvia la video visita con il MS (eventualmente con il PD).

	Attende che il MS risponda alla videochiamata, una volta che il MS risponde, il MS può interagire con il Paziente (nel caso del PD potrà supervisionare l'Operatore durante la somministrazione del Paziente).
OUTPUT	L'Operatore effettua una video visita con il MS

REGISTRAZIONE VIDEO VISITA	
INTRODUZIONE	Attori coinvolti Operatore
	Descrizione generale della funzione L'Operatore registra la video visita
DESCRIZIONE	Sequenza di operazioni Una volta avviata la video visita, l'Operatore preme il pulsante di registrazione, alla fine della video visita verrà salvata automatica sul dispositivo dell'Operatore.
OUTPUT	La video visita viene salvata sul dispositivo dell'Operatore

2.3.8. Gestione Trasferte

INSERIMENTO DATI TRASFERTA OPERATORE	
INTRODUZIONE	Attori coinvolti Operatore
	Descrizione generale della funzione L'Operatore può inserire dati relativi alla trasferta di una visita
INPUT	Descrizione generica dei dati Dati della spesa: ammontare spesa, nota spesa, Informazioni aggiuntive nel caso di spesa dovuta al trasporto (Durata, Mezzo utilizzato, Km percorsi)

DESCRIZIONE	Validazione dei dati E' obbligatorio selezionare una visita e inserire i dati della spesa
	Sequenza di operazioni L'Operatore accede all'applicazione e dopo aver selezionato una visita attiva la funzione per aggiungere la spesa relativa alla trasferta
	Risposta ad eventuali anomalie • Messaggio di errore nel caso non vengano inseriti i campi obbligatori
OUTPUT	Se non ci sono anomalie vengono inseriti i dati relativi alle spese del Paziente

INSERIMENTO DATI TRASPORTO FARMACO	
INTRODUZIONE	Attori coinvolti Admin/Program Manager, Contact Center, Operatore sono gli Utenti autorizzati
	Descrizione generale della funzione L'Utente inserisce i dati relativi al trasporto del farmaco
INPUT	Descrizione generica dei dati Dati sul trasporto: km, mezzo utilizzato, tipo di trasporto (es. vettore logistico, Operatore o Paziente), Data
DESCRIZIONE	Sequenza di operazioni L'Utente autorizzato accede all'applicazione e inserisce i dati relativi al trasporto del farmaco
OUTPUT	Se non ci sono anomalie i dati relativi al trasporto del farmaco vengono salvati.

CONTROLLO TEMPERATURA FARMACO CON DATALOGGER	
INTRODUZIONE	Attori coinvolti Operatore

	Descrizione generale della funzione L'Operatore importa nella piattaforma i dati raccolti con i Datalogger
INPUT	Descrizione generica dei dati Dati raccolti con Datalogger
DESCRIZIONE	Sequenza di operazioni L'Operatore scarica dal Datalogger i dati riguardo la temperatura del farmaco e li carica nel sistema
OUTPUT	Se non ci sono anomalie i dati riguardo la temperatura del farmaco sono inseriti nel sistema

2.4. Requisiti non Funzionali

2.4.1. Usabilità

L'interfaccia grafica del software deve essere progettata con lo scopo di garantire la migliore esperienza Utente (user experience) per l'utilizzo dell'applicazione tramite PC e tablet. L'elevata usabilità deve permettere un facile utilizzo delle funzioni fornite dal sistema, riducendo la necessità di supporto e addestramento da parte degli Utenti.

La piattaforma deve essere testata e validata sul campo per verificarne il grado di usabilità da parte degli Utenti.

2.4.2. Sicurezza

L'applicazione deve essere accessibile tramite il protocollo Internet HTTPS (Hypertext Transfer Protocol Secure), ovvero un canale di comunicazione crittografato tra il server su cui vengono memorizzati i dati e il dispositivo utilizzato dagli Utenti.

L'utilizzo della tecnologia SSL, rispetto a una connessione Web non crittografata, offre un maggiore livello di sicurezza e privacy riducendo il rischio che alcune informazioni salvate nel sistema vengano intercettate utilizzate in modo illecito da terzi.

2.4.3. Compliance

Il sistema deve essere progettato seguendo le linea guida Good Clinical Practice (GCP) adottate dall'Unione Europea e dalla legislazione italiana, tenendo in considerazione la normativa US FDA CFR 21 Part 11 in materia di raccolta di dati clinici.

In particolare, la Video Visita deve essere HIPAA compliant (Health Insurance Portability and Accountability Act, rispettando gli standard per la protezione dei dati sensibili del Paziente.

2.4.4. Disaster recovery

Devono essere applicate adeguate procedure per fare in modo che il sistema possa essere ripristinato in caso si presentino criticità, quali:

- Indisponibilità del server;
- Problema irreparabile al database;
- Errore dell'applicazione.

2.4.5. Backup

Devono essere implementate adeguate procedure di Backup per garantire in qualsiasi momento la possibilità di rispristino dei dati e dell'Applicazione alla versione più recente, in caso di criticità.

Devono essere inoltre implementate procedure di backup automatico a cadenza giornaliera e procedure di backup manuali per le operazioni di manutenzione ordinaria e straordinaria.

3. Organizzazione del Lavoro

3.1. Metodologia di Sviluppo

3.1.1. Agile

Agile è un'insieme di metodologie di sviluppo del software emerse a partire dai primi anni 2000 e fondate su insieme di principi comuni, direttamente o indirettamente derivati dai principi del "Manifesto for Agile Software Development" pubblicato nel 2001:

I metodi agili si contrappongono al modello a cascata e altri processi software tradizionali, proponendo un approccio meno strutturato e focalizzato sull'obiettivo di consegnare al cliente, in tempi brevi e frequentemente, software funzionante e di qualità.

3.1.2. DevOps

DevOps (dalla contrazione inglese di "development" e "operations") è una metodologia di sviluppo del software che punta alla comunicazione, collaborazione e integrazione tra sviluppatori e addetti alle *operations* dell'information technology (IT) nelle fasi di design, sviluppo e produzione.

L'implementazione del DevOps si realizza attraverso i suoi "*five pillars*":

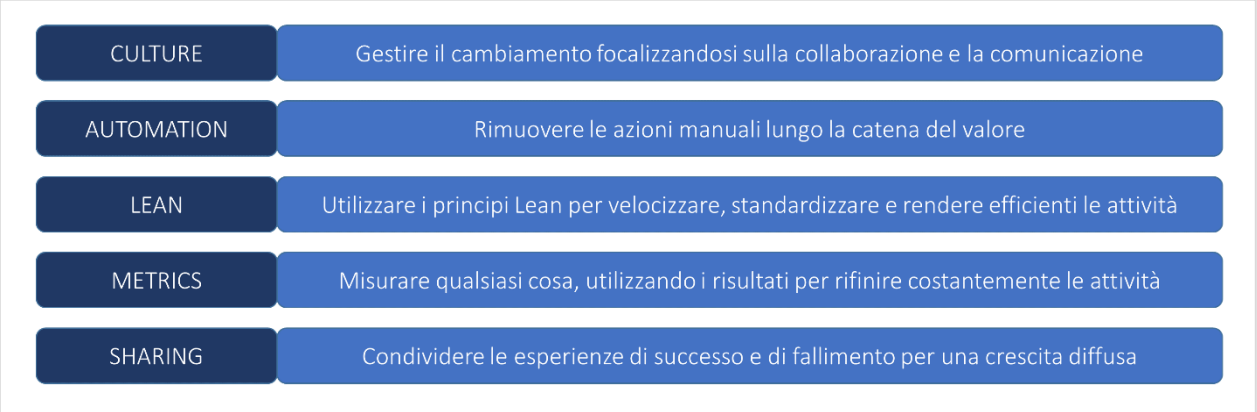


Figura 3.1.2.1 – I cinque pilastri della metodologia DevOps

Questi valori sono stati rispecchiati nella produzione del software attraverso una costante valutazione del lavoro compiuto e una comunicazione costante, favorita dalle ridotte dimensioni del team di sviluppo. La comunicazione è stata costante anche con il cliente: durante la produzione del software si sonoinfatti definite due demo per una valutazione preliminare, e in seguito si è provveduto a rilasciare regolarmente aggiornamenti alla versione online di demo dell'Applicazione.

3.2. Gantt

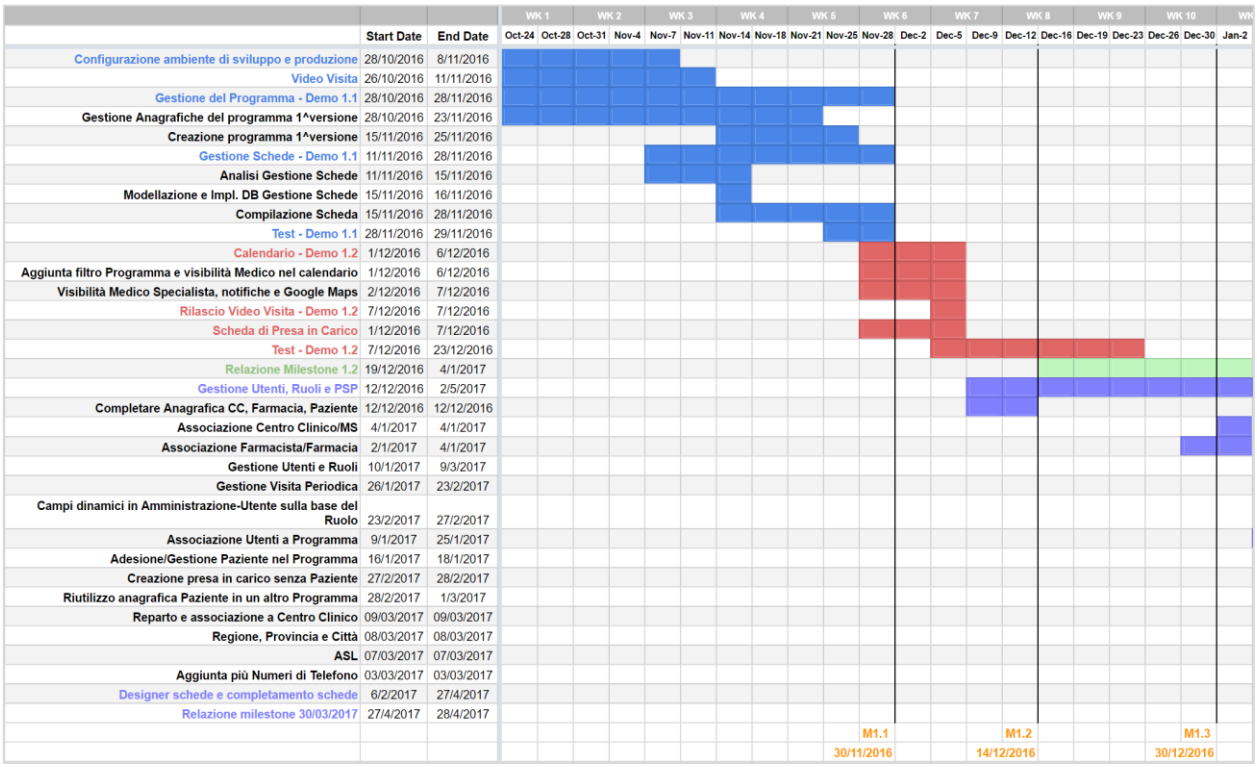


Figura 3.2.1 - Grafico Gantt relativo alle Milestones 1 e 2

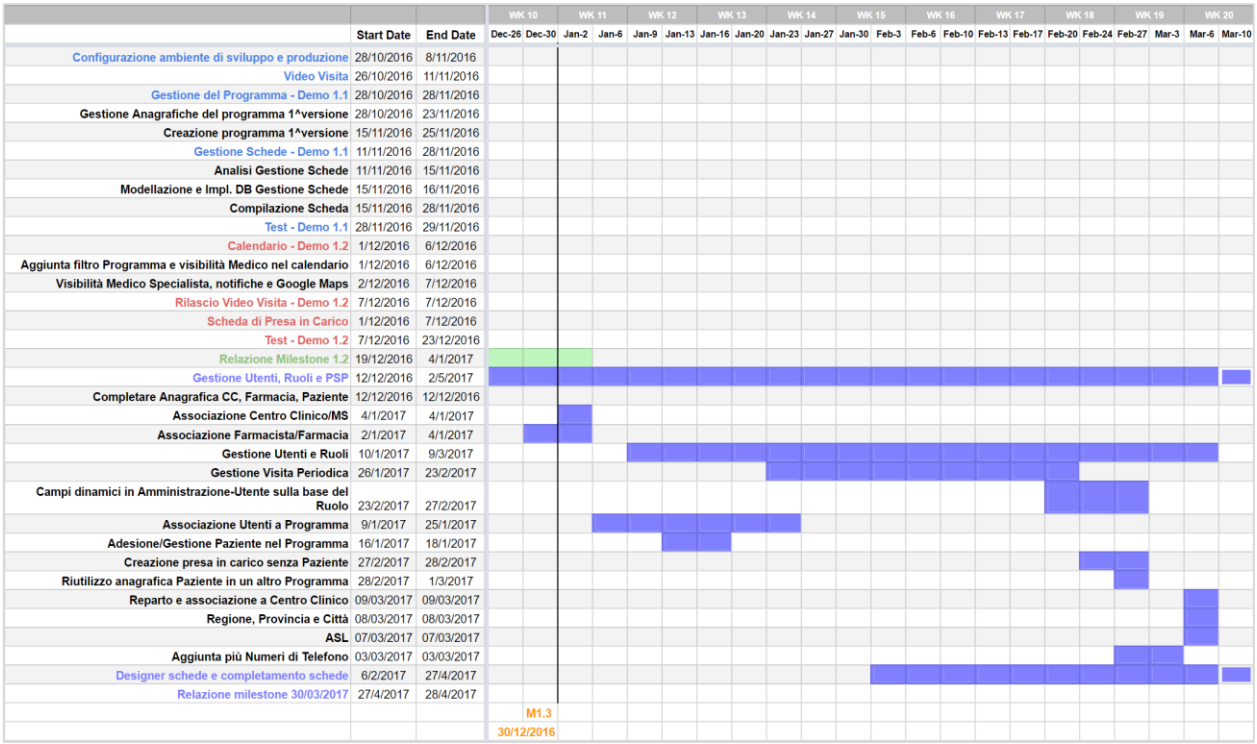


Figura 3.2.2 - Grafico Gantt relativo alle Milestones 3

3.3. Milestones e Demo

Lo sviluppo del progetto, specialmente nella fase iniziale, è stato scandito da due demo, presentate rispettivamente a fine Novembre e a metà Dicembre, sulla base delle quali è stato valutata l'impostazione del lavoro del team di sviluppo e sono state avanzate alcune richieste ulteriori e modifiche rispetto ai requisiti definiti nella fase preliminare all'inizio del progetto.

3.3.1. Milestone 1.1

Nella prima versione dell'Applicazione, il lavoro si è concentrato nel produrre la maggiore quantità possibile di elementi grafici e logiche orientate all'usabilità del software, poichè le valutazioni previste per questo rilascio erano principalmente legate all'impatto che l'interfaccia dell'Applicazione avrebbe avuto sui futuri utilizzatori.

È pertanto stato sviluppato un prototipo del Calendario, senza filtri e corredato da una logica delle Attività particolarmente semplice, che ne permetteva solo l'inserimento e l'eliminazione. Contestualmente al Calendario, è stata realizzata la Video Visita integrando il software HIPAA compliant Vsee.

La dimostrazione della Video Visita, in una prova dal vivo, ne ha dimostrato l'usabilità come strumento utile al Medico per intervistare il Paziente riguardo le sue condizioni, ed eventualmente per verificarne sommariamente le condizioni fisiche.

È stata infine sviluppata la gestione dinamica delle Schede Visita in una versione stabile, che però non includeva il modulo per il design delle Schede da parte degli Utenti.

La dimostrazione dell'uso delle Schede, avvenuta attraverso un *tablet*, per riprodurre le condizioni in cui sarà effettivamente utilizzata, ne ha dimostrato la praticità, ricercata attraverso la riduzione al minimo degli input da tastiera da parte dell'Operatore.

3.3.2. Milestone 1.2

Nel breve periodo tra la prima e la seconda Milestone, il lavoro si è concentrato nell'implementazione della logica per gestire diversi Programmi Assistenziali e sulle conseguenti modifiche alla visibilità delle Attività nel Calendario sulla base dei Ruoli e del Programma scelti.

In particolare, sono stati introdotti i filtri nel Calendario, in modo da permettere ai Manager di visualizzare solo Attività di un particolare Tipo, Programma e/o Paziente. È stata poi introdotta una prima differenziazione nella visibilità delle Attività sulla base del Ruolo: il Medico Specialista non poteva infatti visualizzare le Attività di Tipo Indisponibilità o Ferie degli Operatori, in quanto non di suo interesse. È inoltre stata estesa la finestra dell'Attività attraverso una tab che mostra la posizione geografica del Paziente attraverso le API di Google Maps.

È stata infine aggiornata la gestione delle Schede, introducendo una Scheda specifica per la presa in carico del Paziente, ovvero la prima Visita che viene effettuata.

Durante la dimostrazione è stato mostrato il funzionamento additivo dei filtri del Calendario, e nuovamente la Video Visita, che in virtù delle modifiche effettuate permetteva al solo Medico Specialista di richiederla ed al solo Operatore di effettuarla, e solo in corrispondenza dell'orario stabilito.

3.3.3. Milestone 1.3

Nella terza Milestone, terminata alla fine di Dicembre, non sono state introdotte novità sostanziali dal punto di vista delle funzionalità, ma sono state corrette numerose falle generatesi per via del lavoro intenso dovuto ai due rilasci precedenti, ed è stata prodotta la documentazione che descriveva i requisiti ed il lavoro svolto dall'inizio del progetto.

Per via della sua natura, alla terza Milestone non è seguita una dimostrazione, e dalla ripresa del lavoro a Gennaio si è scelto di procedere con rilasci frequenti sulla versione online di demo dell'Applicazione per permettere al cliente una valutazione più autonoma ed una comunicazione più rapida in caso di rilevamento di malfunzionamenti o comportamenti indesiderati.

3.3.4. Versioni successive

Da Gennaio a Marzo ho lavorato su numerose Attività, molte delle quali hanno implicato la modifica e l'estensione della struttura del Database, in quanto sono stati definiti legami più complessi tra le tabelle, con numerose tabelle di raccordo per realizzare relazioni 1-N.

Ho inoltre completato la struttura delle Attività del Calendario, introducendo la logica delle Attività Ripetute:

ATTIVITÀ	ORE STIMATE	ORE EFFETTIVE	DATA INIZIO	DATA FINE
Associazione Farmacista/Farmacia	**12**	**17**	**2/1/2017**	**4/1/2017**
Dao/Service Associazione Farmacista/Farmacia	3	9	2/1/2017	3/1/2017
Pannello Associazione Farmacista/Farmacia	7	7	2/1/2017	3/1/2017
Test Associazione Farmacista/Farmacia	2	1	4/1/2017	4/1/2017
Associazione Centro Clinico/MS	**12**	**7**	**4/1/2017**	**4/1/2017**
Dao/Service Associazione Centro Clinico/MS	3	3	4/1/2017	4/1/2017
Pannello Associazione Centro Clinico/MS	7	3	4/1/2017	4/1/2017
Test Associazione Centro Clinico/MS	2	1	4/1/2017	4/1/2017
Adesione/Gestione Paziente nel Programma	**18**	**17**	**16/1/2017**	**18/1/2017**
Aggiornamento Anagrafica Paziente con Programma	4	9	17/1/2017	18/1/2017
Dao/Service adesione Paziente al Programma	4	1	18/1/2017	18/1/2017

Pannello adesione Paziente al Programma	4	2	18/1/2017	18/1/2017
Modifica Visibilità Calendario	2	2	18/1/2017	18/1/2017
Test adesione Paziente al Programma	4	3	18/1/2017	18/1/2017
Gestione Visita Periodica	**36**	**138**	**26/1/2017**	**23/2/2017**
Dao/Service Gestione Visita Periodica	4	17	26/1/2017	30/1/2017
Logica Visita Periodica	12	92	30/1/2017	22/2/2017
Pannello Gestione Visita Periodica	8	12	16/11/2016	30/1/2017
Test Visita Periodica	12	17	30/1/2017	23/2/2017
Campi dinamici in Amministrazione-Utente sulla base del Ruolo	**10**	**17**	**23/2/2017**	**27/2/2017**
Associazione Utenti a Programma	**18**	**35**	**9/1/2017**	**25/1/2017**
Aggiornamento UserAccountRole	4	4	9/1/2017	9/1/2017
Dao/Service associazione Utenti a Programma	4	8	9/1/2017	10/1/2017
Pannello associazione Utenti a Programma	4	6	10/1/2017	11/1/2017
Selezione dell'Operatore subordinata a Programma in Calendario	2	1	11/1/2017	11/1/2017
Test Associazione Utenti a Programma	4	16	11/1/2017	25/1/2017
Mapping Programma-ActivityType dinamico	**8**	**15**	**12/1/2017**	**16/1/2017**
Creazione presa in carico senza Paziente	**9**	**4**	**27/2/2017**	**28/2/2017**
Permettere l'inserimento di un'Attività senza Paziente	4	2	27/2/2017	27/2/2017
Disabilitare Scheda se Paziente non presente	1	1	27/2/2017	27/2/2017
Togliere obbligatorierà del campo Paziente	2	1	27/2/2017	27/2/2017
Test creazione presa in carico senza Paziente	2	0	27/2/2017	28/2/2017
Riutilizzo Anagrafica Paziente in un altro Programma	**6**	**6**	**28/2/2017**	**1/3/2017**
Dao/Service associa Anagrafica a un altro Programma	3	2	28/2/2017	28/2/2017
Riutilizzo pannello in modalità associazione a un altro Programma	1	2	28/2/2017	28/2/2017
Riutilizzo Anagrafica Paziente in un altro Programma	2	2	28/2/2017	1/3/2017
Reparto e associazione a Centro Clinico	**10**	**6**	**09/03/2017**	**09/03/2017**
~~Dao/Service Reparto~~	~~0~~	~~0~~		

~~Pannello per gestione Reparti~~	~~0~~	~~0~~		
Associazione Reparti a Centro Clinico	6	4	09/03/2017	09/03/2017
Aggiunta ComboBox CentroClinicoReparto in Anagrafica Paziente	4	2	09/03/2017	09/03/2017
Regione, Provincia e Città	**18**	**8**	**08/03/2017**	**08/03/2017**
Importazione Regioni, Province e Città esistenti da sorgente esterna	3	3	08/03/2017	08/03/2017
~~Aggiunta FK a tabella Adress~~	~~1~~	~~0~~		
~~Dao/Service Adress~~	~~6~~	~~0~~		
Selezione dei campi tramite ComboBox	8	5	08/03/2017	08/03/2017
ASL	**6**	**8**	**07/03/2017**	**07/03/2017**
~~Dao/Service ASL~~	~~0~~	~~0~~		
Aggiunta ASL da repository ASL	3	3	07/03/2017	07/03/2017
~~Pannello per gestione ASL~~	~~0~~	~~0~~		
Aggiunta ComboBox ASL in Anagrafica Paziente	3	5	07/03/2017	07/03/2017
Aggiunta più Numeri di Telefono	**8**	**16**	**03/03/2017**	**03/03/2017**
Dao/Service Numeri di Telefono dinamici	3	8	03/03/2017	06/03/2017
Componente grafico per aggiunta dinamica di Numeri di Telefono	5	8	03/03/2017	06/03/2017

4. Sistema Informativo

4.1. Tecnologie Implementative

4.1.1. Vaadin

Vaadin è un framework che facilita la creazione di interfacce Utenti (UI) per applicazioni Web nascondendo completamente allo sviluppatore la parte di comunicazione tra Server e Client, e rendendo di fatto la creazione di UI per applicazioni Web simile alla creazione di UI per applicazioni locali.

Consiste in un *framework server-side* ed un *engine client-side* che viene eseguito nel browser come programma JavaScript e si occupa unicamente di fare il *rendering* dell'interfaccia Utente e di contattare il server in occasione delle interazioni dell'Utente con l'UI. L'Applicazione viene eseguita come una sessione *Servlet Java* in un server Java.

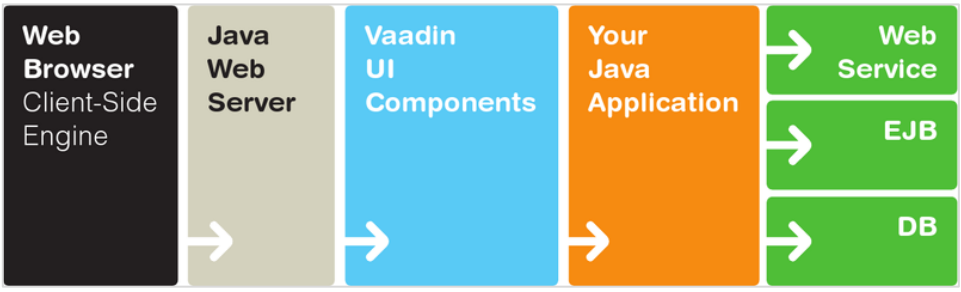


Figura 4.1.1.1 - Architettura base di Vaadin

Vaadin fa uso di due tecnologie fondamentali:

- GWT (Google Web Toolkit), un set di strumenti open-source che permette la creazione ed il mantenimento di complesse applicazioni front-end JavaScript scritte in Java, e che comprende:
  - Un compilatore da Java a JavaScript;
  - Una serie di interfacce personalizzabili e classi per creare widget.
- AJAX (Asynchronous JavaScript and XML), una tecnologia che permette uno scambio di dati in background fra web browser e server, e che consente l'aggiornamento dinamico di una pagina web senza un esplicito ricaricamento da parte dell'utente.

La logica dell'Applicazione viene eseguita *server-side*, a differenza delle implementazioni più tradizionali di GWT, in cui è eseguita *client-side*. Questo comporta una serie una serie di vantaggi e svantaggi:

- Viene fatto ricorso a richieste al Server ad ogni iterazione con i componenti grafici, interrompendo l'esperienza utente in caso di banda insufficiente;
- La logica dell'Applicazione viene eseguita sul Server e non è accessibile all'utente;

- Non deve essere fatta una distizione tra componente Client e Server del codice, e questo permette di risparmiare una grande quantità di tempo e ridurre la complessità dell'Applicazione;
- Non deve essere definita la logica di comunicazione Client-Server;
- È possibile associare sorgenti dati, quali ad esempio un Database, ai widget;
- È possibile utilizzare uno qualunque dei linguaggi della JVM (Java Virtual Machine), mentre le implementazioni più tradizionali di GWT sono legate al solo Java.

L'esecuzione del *client-side engine* come JavaScript non richiede l'installazione di alcun *plugin* nel browser. Questo rende Vaadin molto vantaggioso rispetto ad altri framework basati su *plugin* come *Flash*. Inoltre, l'implementazione del GWT assicura il supporto costante per un ampio numero di browser.

4.1.2. Hibernate

Hibernate è un framework che si occupa del mapping tra entità di un Database SQL e oggetti Java, utilizzando i driver JDBC. L'obiettivo di Hibernate, come di altri strumenti di Object/Relational Mapping, è quello di garantire la persistenza dei dati, ovvero la possibilità di salvare lo stato dei dati oltre il livello della JVM, utilizzando quindi un Database.

Vi è però una differenza sostanziale tra il modello dei RDBMS (Relational Database Management System), che rappresenta i dati in forma tabellare, e il modello degli OOL (Object Oriented Languages), come Java, che rappresentano i dati sottoforma di un grafo interconnesso di oggetti, sulla base dell'ereditarietà tra classi.
Per via della differente rappresentazione dei dati nei due approcci, sono state definite le JPA (Java Persistence API), ovvero uno standard per gestire dati relazionali in applicazioni Java.

Hibernate, fornendo un'implementazione delle JPA, permette di interfacciarsi un con RDBMS seguendo gli idiomi tipici dell'OOL.

4.1.2.1. Entity

Le tabelle del Database vengono rappresentate attaverso le classi Entity, che sono POJO (Plain Old Data Object), ovvero oggetti Java che non estendono nè implementano altre classi.

Le Entity sono correlate alle rispettive tabelle del Database da una serie di annotazioni basate sulle specifiche delle JPA, in particolare:

- @Entity: dichiara la classe come una Entity
- @Table: permette di associare una Tabella, un Catalogo ed uno Schema del Database all'Entity
- @Id: dichiara la proprietà identificativa di questa Entity, che corrisponde alla chiave primaria del Database
- @GeneratedValue: specifica la modalità con cui viene generato l'identificativo dell'Entity

- @ManyToOne: definisce il grado di associatività di una particolare proprietà all'Entity
- @JoinColumn: specifica a quale colonna del Database di riferisce una proprietà.

```
@Entity
@Table(name = "activity", schema = "public")
public class Activity implements java.io.Serializable {

    private Long idActivity;
    private UserAccount operator;
    private ProgramPatient programPatient;
    private ActivityType activityType;
    private Date startDate;
    private Date endDate;
    private String comments;
    private Boolean videoVisit;
    private String videoVisitUrl;
    private Boolean deleted;

    public Activity() {
    }

    @Id
    @GeneratedValue(strategy = IDENTITY)
    @Column(name = "id_activity", unique = true, nullable = false)
    public Long getIdActivity() {
        return this.idActivity;
    }

    public void setIdActivity(Long idActivity) {
        this.idActivity = idActivity;
    }

    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "id_operator")
    public UserAccount getOperator() {
        return this.operator;
    }

    public void setOperator(UserAccount operator) {
        this.operator = operator;
    }
```

Figura 4.1.2.1.1 - Esempio di Entity (i getter ed i setter non sono stati riportati integralmente)

Chiamando le funzioni *save()* e *update()* di Hibernate, i valori delle proprietà dell'Entity vengono quindi riportate nel Database. Questo fa sì che il programmatore non debba incorporare codice SQL per realizzare le operazioni CRUD sulle Entity.

4.1.2.2. Criteria Query

Le Criteria Query sono query di alto livello implementate utilizzando oggetti Hibernate chiamati Criterion e Restriction.
I Criterion rappresentano rappresenta la query, e definisce la classe Entity sulla quale la query

viene eseguita, mentre le Restriction sono condizioni logiche sui valori delle Entity, come ad esempio:

- Restrictions.or()
- Restrictions.and()
- Restrictions.like("name", "Fritz"))
- Restrictions.eq("age", new Integer(0))
- Restrictions.isNull("age")
- Restrictions.between("weight", minWeight, maxWeight)

Le Restriction possono essere combinate tra di loro ed aggiunte dinamicamente al Criterion.

Le Criteria Query rendono possibile l'accesso ad Entità in relazione logica tramite la semantica OOP, e quindi sfruttare l'ereditarietà tra classi e il polimorfismo per accedere alle proprietà di Entità logicamente correlate.
Quando il Criterion è definito, il metodo *list()* restituisce i risultati in una lista del tipo definito dal Criterion.

A diffrenza degli altri tipi di query, le Criteria Query non permettono di specificare i parametri della select, quindi si otterrà sempre in output una lista di Entity.

```
List<UserAccountRole> userRoles = (List<UserAccountRole>) getSessionFactory().getCurrentSession().
        createCriteria(UserAccountRole.class).add(
                Restrictions.and(
                        Restrictions.eq("userAccount.idUserAccount", idUserAccount),
                        Restrictions.eq("userRole", roleCurrent),
                        Restrictions.eq("deleted", false)
                        )
                ).list();
UserAccountRole userAccountRole = userRoles.get(0);
```

Figura 4.1.2.2.1 - Criteria Query

4.1.2.3. HQL Query

Le HQL Query sono query che presentano una sintassi simile a quella SQL, ma a differenza delle query SQL sono completamente orientate agli oggetti e rendono possibile sfruttare nozioni dell'OOP quali ereditarietà e polimorfismo.

```
StringBuilder hqlQuery = new StringBuilder(
        "SELECT igv "
                + "FROM FormItemGroupValue igv "
                + "WHERE igv.formItemGroup in :formItemGroup "
                + "AND igv.activityId=:activityId "
                + "AND igv.deleted=false "
                + "ORDER BY igv.rowOrder,colOrder"
        );

Query query = getSessionFactory().getCurrentSession().createQuery(hqlQuery.toString());
query.setParameterList("formItemGroup", formItemGroupList);
query.setParameter("activityId", idActivity);
@SuppressWarnings("unchecked")
List<FormItemGroupValue> result = (List<FormItemGroupValue>)query.list();
```

Figura 4.1.2.3.1 - HQL Query

4.1.2.4. SQL Query

Attraverso Hibernate è possibile esguire query in linguaggio SQL nativo. I risultati delle SQL Query vengono restituiti in una lista di Object (Object[]), per cui è necessario effettuare un cast di tutti i risultati e mapparli manualmente nell'Entity.

```
StringBuilder sqlQueryListActivity = new StringBuilder(
        "SELECT a.*, at.type_color, reg.name as patient_name, reg.surname as patient_surname, pa.id_patient as id_patient, "
                +"pr.program_id as id_program, pr.program_name as program_name "
        +"FROM activity a JOIN activity_type at on a.id_activity_type = at.id_activity_type "
        +"JOIN program_patient pp on a.id_program_patient = pp.id_program_patient "
        +"JOIN program pr on pp.id_program = pr.program_id "
        +"JOIN patient pa on pp.id_patient = pa.id_patient "
        +"JOIN dblink('dbname= "
        + DbCredentials.getSchemaReg()
        +" user= "
        + DbCredentials.getUserReg()
        +" password= "
        + DbCredentials.getPasswordReg()
        +"', 'select name, surname, id_registry from registry') AS reg(name varchar(105), surname varchar(105), id_registry integer) "
        +"ON pa.id_registry = reg.id_registry "
        +"WHERE a.deleted = false "
        +"AND a.id_activity = " + idActivity + " ");
Query queryListActivity= getSessionFactory().getCurrentSession()
        .createSQLQuery(sqlQueryListActivity.toString());
queryListActivity.setResultTransformer(AliasToEntityMapResultTransformer.INSTANCE);
@SuppressWarnings("unchecked")
List<Map<String, Object>> listMap = queryListActivity.list();
Iterator<?> iter = listMap.iterator();
while(iter.hasNext()) {
    ActivityBean ab = new ActivityBean();

    @SuppressWarnings("rawtypes")
    Map record = (Map) iter.next();

    if (record!=null){

        if(record.get("id_activity") != null)
            ab.setIdActivity(Long.valueOf(record.get("id_activity").toString()));
        if(record.get("start_date")!=null){
            Date start = ((Date)record.get("start_date"));
            ab.setStartDate(start);
        }
```

Figura 4.1.2.4.1 - SQL Query (la parte di setting delle proprietà non è stata riportata integralmente)

4.1.2.5. Prestazioni di Criteria, HQL e SQL Query

In fase di progettazione del software è stato eseguito un test di prestazioni tra le tre tipologie di query che Hibernate mette a disposizione, su un dataset creato automaticamente che comprende due tabelle, caratterizzate dalle stesse colonne:

TestTable	
NOME COLONNA	TIPO DI DATO

id	Serial
name	Varchar (80)
deleted	Boolean
column_1	Varchar (80)
column_2	Varchar (80)
column_3	Varchar (80)
column_4	Varchar (80)
column_5	Varchar (80)
column_6	Varchar (80)
column_7	Varchar (80)

Il test è stato suddiviso in due parti:

- Nella prima parte si sono utilizzati i tre tipi di query per elencare tutte le entità TestTable con dimensione del Database di:
    - 1000 record
    - 10000 record
    - 100000 record
- Nella seconda parte i tre tipi di query sono stati utilizzati realizzando un join tra la TestTable1 e la TestTable2 sul campo “name”, con dimensione del Database di:
    - 1000 record
    - 10000 record
    - 32500 record

Ogni misurazione è stata ripetuta dieci volte per ogni caso, per valutare l’impatto di eventuali valori outlier:

Timestamp: 22/09/2016 16:06:03		
DB Size: 1000		
HQL Query	SQL Query	Criteria Query
795.684546	78.577527	76.931715
79.414318	15.944928	40.259587
32.674123	18.079775	36.681789
37.44432	18.714311	34.638713
35.593838	20.583508	33.320735
35.077636	18.17094	31.460594
36.979435	16.178578	36.215093
31.924271	17.943328	29.173605
35.696475	62.48468	28.727436
36.318334	30.044207	39.60573
Avg: 115.68072959999999	Avg: 29.672178199999998	Avg: 38.70149970000001
Timestamp: 22/09/2016 16:06:07		
DB Size: 10000		
HQL Query	SQL Query	Criteria Query
157.161694	96.42667	106.140337
183.04911	91.563497	115.890833
142.205702	102.374774	84.110253
158.635438	98.858558	54.515235
141.088167	90.028171	45.364861
137.440335	73.153473	66.298543
168.250091	71.412271	59.922986
106.760384	68.246831	67.86949
99.85655	67.902696	51.512806
104.483659	69.240596	49.799979
Avg: 139.893113	Avg: 82.9207537	Avg: 70.1425323
Timestamp: 22/09/2016 16:06:16		
DB Size: 100000		
HQL Query	SQL Query	Criteria Query
1451.172052	405.141258	742.802444
496.362431	331.100279	610.672658
541.515982	360.453196	720.878016
492.662073	309.762692	882.369775
526.796657	329.365113	592.511377
1070.716151	283.778073	704.409639
466.53678	365.629713	538.905385
478.311033	289.121827	551.647441
525.47868	492.763502	526.485124
459.645022	454.125577	595.668365
Avg: 650.9196861	Avg: 362.124123	Avg: 646.6350224

Figura 4.1.2.5.1 - Performance Test

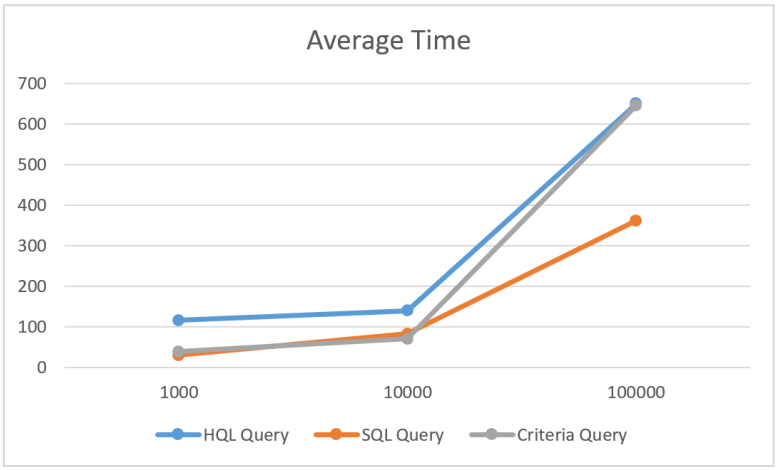


Figura 4.1.2.5.2 - Grafico - Asse X: dimensione DB - Asse Y: tempo (ms)

Timestamp: 23/09/2016 14:21:31		
DB Size: 1000		
HQL Query	SQL Query	Criteria Query
353.507099	26.721793	104.992616
37.344702	8.216378	90.93379
27.719181	6.614639	95.939446
16.686328	7.384416	66.486911
16.12726	9.812682	56.983952
16.668217	10.974288	79.507899
18.37561	6.525889	89.560871
17.439804	6.756521	84.697095
16.429133	6.719692	70.856824
16.29329	6.85312	88.743399
Avg: 53.659062399999996	Avg: 9.657941800000001	Avg: 82.87028029999999
Timestamp: 23/09/2016 14:21:35		
DB Size: 10000		
HQL Query	SQL Query	Criteria Query
168.430611	72.072165	3197.382641
129.104415	59.48527	2975.784961
97.829775	44.087939	2988.514943
81.845603	35.143444	3003.589873
112.67649	34.424383	3027.225921
121.501443	49.252382	3051.661932
73.1577	54.847899	3060.78091
75.844369	46.29946	2969.53136
60.610653	35.17665	3010.922971
69.474246	36.280298	2964.547438
Avg: 99.0475305	Avg: 46.706989	Avg: 3024.994295
Timestamp: 23/09/2016 14:22:18		
DB Size: 32500		
HQL Query	SQL Query	Criteria Query
274423.6612	157.793212	30969.06527
418.772029	140.431896	28821.90342
578.162149	134.661292	28905.60795
255.783584	155.656556	28887.08865
264.693062	148.797399	32659.03753
253.300981	182.233449	30282.50658
262.433844	142.192419	30928.35891
223.740979	114.172572	30384.15807
268.70133	117.022857	30184.34354
264.48477	128.398032	28898.86532
Avg: 27721.373389099997	Avg: 142.1359684	Avg: 30092.093525099997

Figura 4.1.2.5.3 - Performance Test – Nested

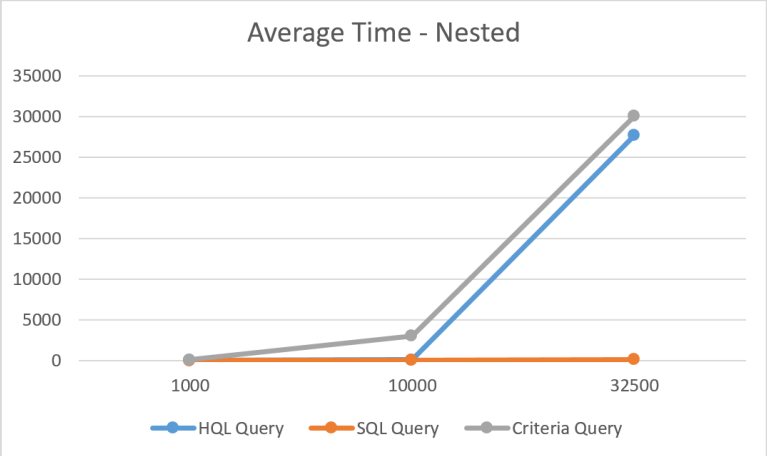


Figura 4.1.2.5.4 - Grafico Nested - Asse X: dimensione DB - Asse Y: tempo (ms)

In entrabi i casi è possibile notare come, a prescindere dal tipo di query, all'aumentare della dimensione del set dei risultati, ed in particolare dopo le 10000 unità, le prestazioni della Criteria e dell'HQL Query degradano molto più rapidamente che quelle dell'SQL Query.

In particolare, inoltre, se la Criteria Query si è dimostrata più veloce nel caso di semplice lettura di una tabella dal Database, l'HQL Query è risultata più rapida nel caso di una semplice join su un campo.

Visti e considerati questi risultati, si è scelto di limitare l'utilizzo di Criteria Query ed HQL Query alle semplici operazioni CRUD sui singoli dati, mentre nei casi in cui la query può generare result set più grandi sono state utilizzate le SQL Query.

4.2. Architettura

Il software è diviso gerarchicamente in tre strati:

- Data Logic (o Data Model), che si occupa delle operazioni fondamentali sui dati;
- Data Integration (o Data Binding), che si occupa di integrare i risultati provenienti dalla Data Logic per realizzare logiche più complesse;
- User Interface, che permette all'Utente di visualizzare e modificare i dati proventienti dalla Data Integration.

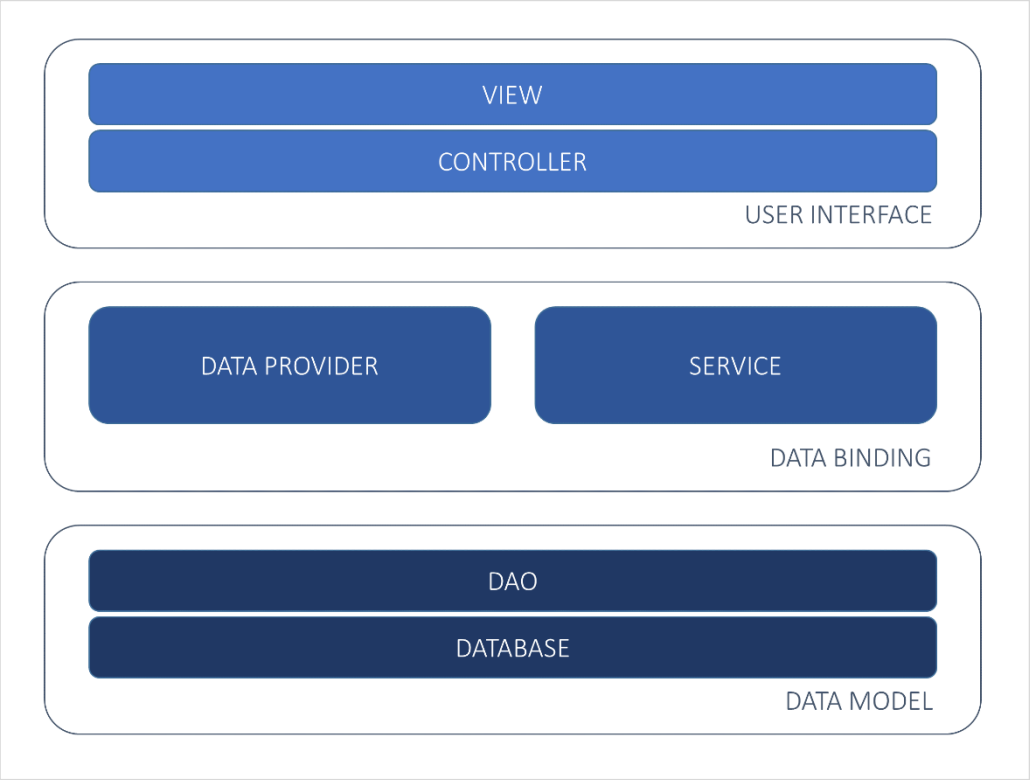


Figura 4.2.1 - Architettura del software

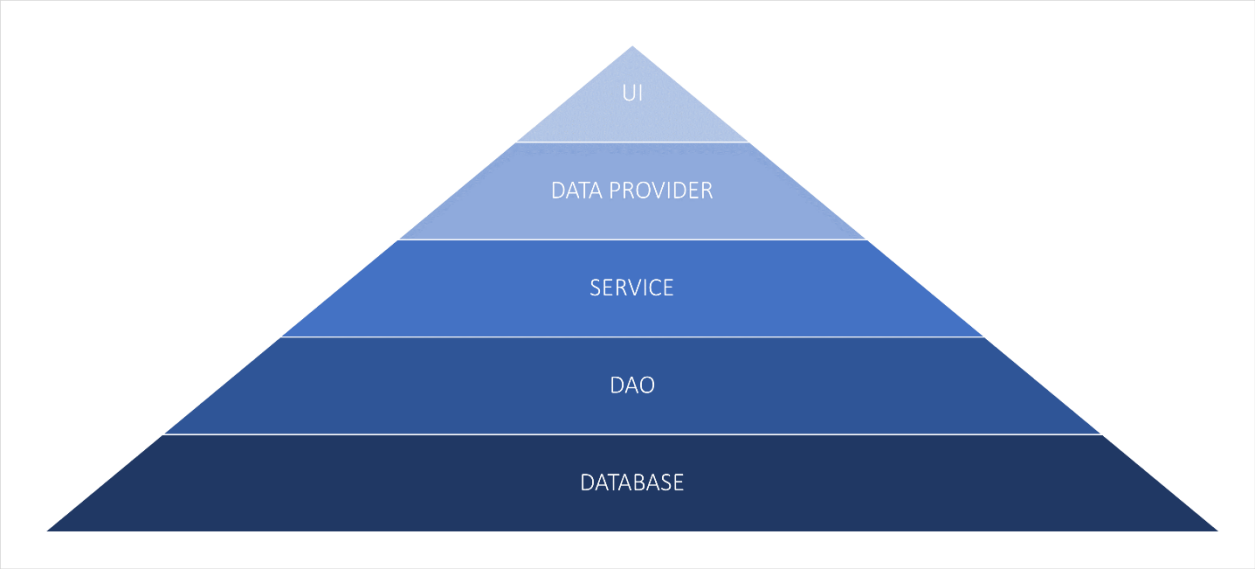


Figura 4.2.2 - Architettura gerarchica del software

4.2.1. Data Logic

Lo strato di Data Logic è quello più vicino ai dati, che contiene:

- Il Database

- I DAO (*data access object*)

4.2.1.1. Database

Lo strato di Database è quello più basso, che garantisce la persistenza e permette le operazioni fondamentali sui dati.

Considerata la complessità dello schema logico dei dati, si è scelto di utilizzare un RDBMS (*Relational Database Management System*), ed in particolare PostgreSQL.

La struttura delle tabelle nel Database può essere suddivisa in quattro gruppi:

- La gestione delle Attività del Calendario:

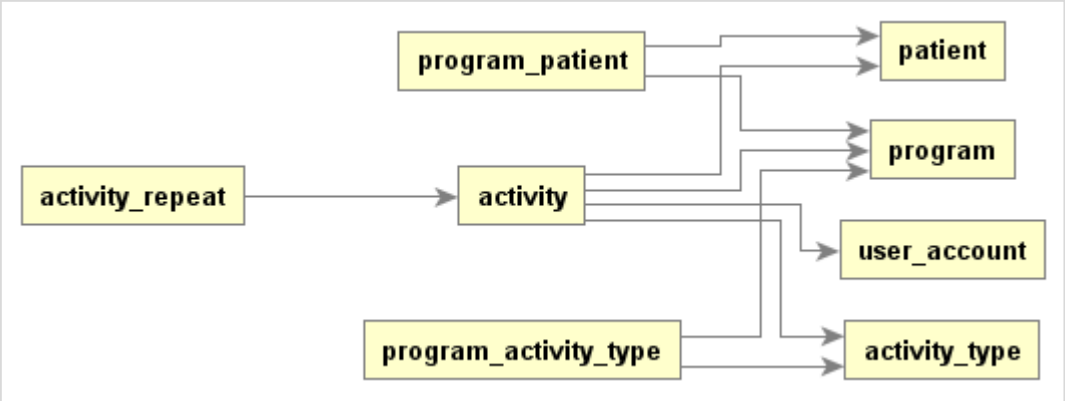


Figura 4.2.1.1.1 - Tabelle riguardanti la gestione delle Attività

- La gestione delle Anagrafiche di Pazienti, Programmi, Farmacie e Centri Clinici:

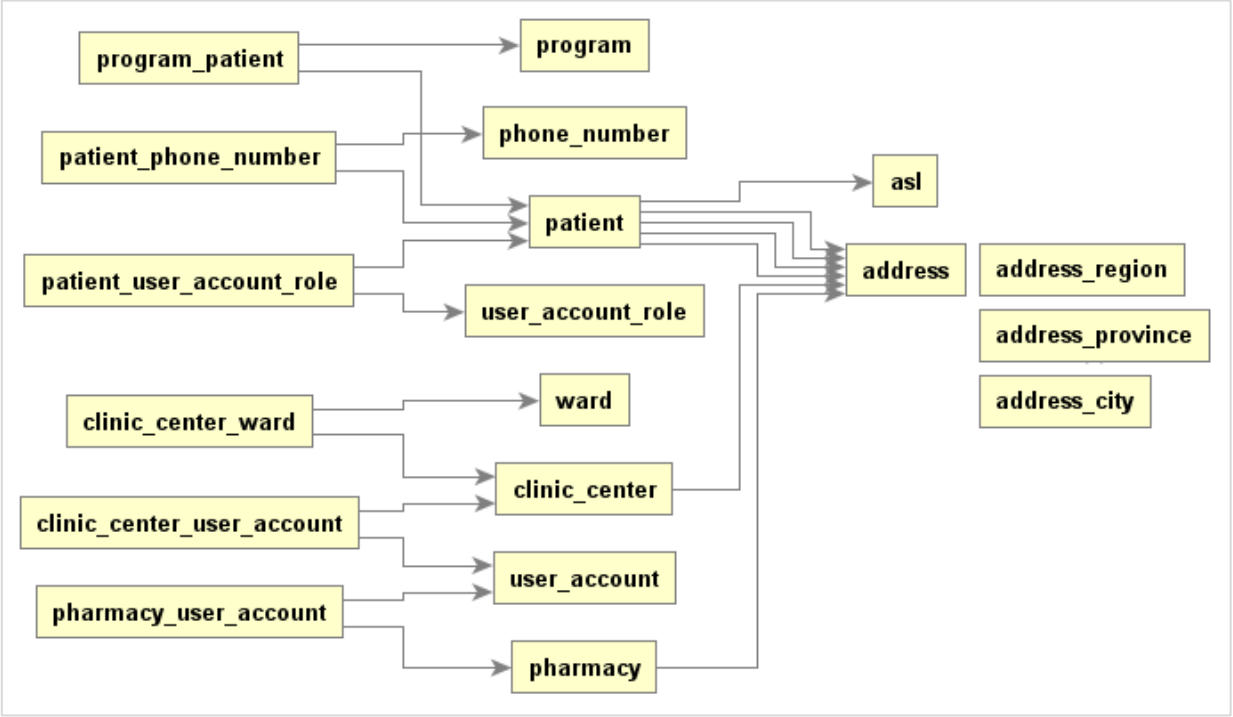


Figura 4.2.1.1.2 - Tabelle riguardanti la gestione delle Anagrafiche

- La gestione di Utenti dell'Applicazione, Ruoli e Permessi:

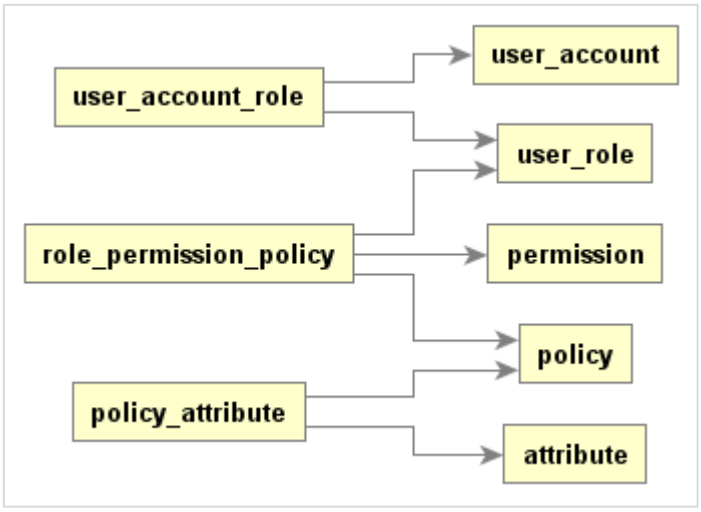


Figura 4.2.1.1.34 - Tabelle riguardanti la gestione di Ruoli e Permessi

- La gestione delle Schede Visita:

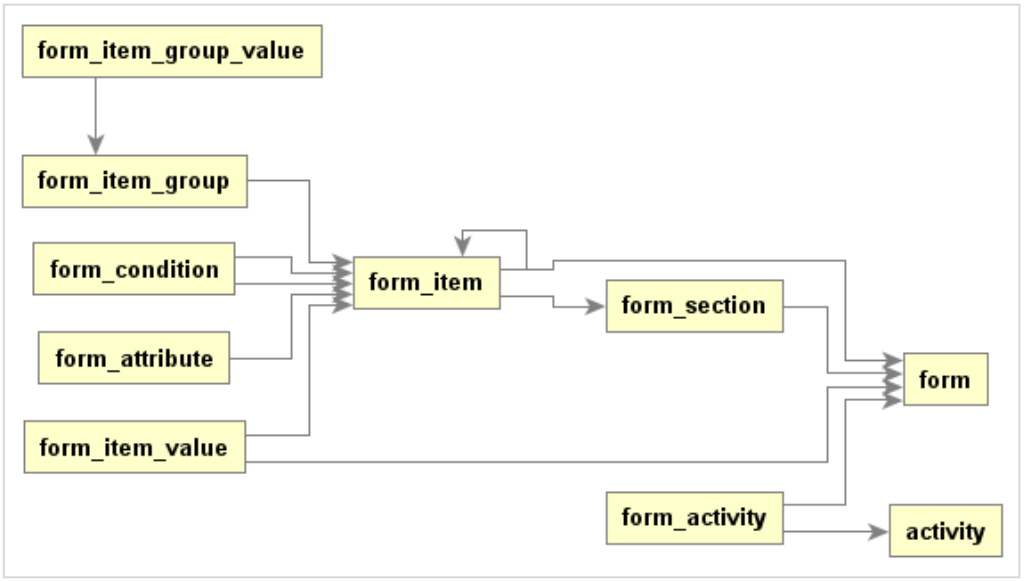


Figura 4.2.1.1.4 - Tabelle riguardanti la gestione delle Schede

I dettagli delle tabelle e delle scelte implementative sono discussi in dettaglio nel capitolo sui Componenti.

4.2.1.2. Data Access Object

I Data Access Object sono classi che si occupano in via esclusiva di interfacciarsi con il Database, e contengono le funzioni per effettuare operazioni di tipo CRUD (Create, Read, Update e Delete).

Le singole classi DAO vengono realizzati attraverso una classe interfaccia, ad esempio *UserAccountDao* ed una classe che implementa quell'interfaccia, ad esempio *UserAccountDaoImpl*.

Le classi DAO sono fortemente dipendenti dall'architettura del Database sottostante, in particolare se realizzati con le più performanti SQLQueries, che utilizzano la sintassi propria del Database. Per questo è opportuno utilizzarli unicamente per le operazioni CRUD, delegando logiche più complesse ai Service, nello strato di Data Integration.

4.2.2. Data Integration

Lo strato di Data Integration è quello intermedio, che contiene:

- I Service
- I Data Provider

4.2.2.1. Service

I Service sono classi che si occupano di aggregare i risultati delle singole funzioni dei DAO in maniera organica e operare quindi logiche di integrazione più complesse.

I Service sono l'unico punto di accesso al livello di Data Logic reso disponibile alle classi dell'interfaccia Utente e ai Provider.

Viene realizzato un Service per ogni Entity relativa ad una tabella del Database che rappresenta un'entità nello schema ER, mentre le classi che rappresentano delle relazioni nello schema ER non sono associate ad un Service autonomo. Per accedere alle funzioni dei DAO di queste ultime tabelle, sono implementati delle funzioni specifiche nel Service dell'entità logicamente più vicina. Ad esempio, le funzioni di *UserAccountRoleDao*, relativo alla tabella di mapping *UserAccountRole* (che associa uno *UserAccount* ad un *Role*) sono accessibili dal Service di *UserAccount.*

4.2.2.2. Data Provider

Il DataProvider è una classe che genera Containers, ovvero strutture dati che contengono un elenco di risultati, a partire dai risultati dei Service. I Container sono tipicamente utilizzati per popolare Checkboxes, Comboboxes e Tabelle e sono ottimizzati per questo scopo.

4.2.3. User Interface

Lo strato di User Interface comprende:

- I Listener
- I Pannelli e le Finestre

4.2.3.1. Listener

Lo strato logico denominato Controller è costituito da tutti i Listener dei vari componenti dell'interfaccia grafica.

I Listener sono classi che si occupano di intercettare ogni richiesta dell'Utente e contattare il Provider o direttamente i Service per richiedere dati.

Ogni Listener è associato ad un componente con il quale l'Utente può interagire, come un Button, una TextFields o una CheckBox. Quando il Listener rileva che l'Utente ha cliccato sul componente ad esso associato, contatta lo strato di Data Binding inoltrando una richiesta.

4.2.3.2. Pannelli e Finestre

I Pannelli e le Finestre contengono gli elementi grafici con cui interagisce l'Utente ed i rispettivi Listener.

Per mostrare i dati provenienti dagli strati inferiori, vengono utilizzate diverse tipologie di componenti, tra cui:

- Label, campi di testo non modificabili dall'Utente;
- TextField, campi di testo modificabili dall'Utente;
- DateField, piccoli calendari tramite i quali l'Utente può selezionare una data precisa;
- ComboBox, menù a tendina che permettono la scelta di una opzione tra quelle a disposizione;
- Button, pulsanti sempre collegati ad un Listener che compiono un'azione quando premuti;
- CheckBox, caselle spuntabili che permettono di selezionare più di una opzione;
- Table, tabelle che permettono una visualizzazione dettagliata dei dati;
- CheckTable, simili alle Table, ma con una CheckBox integrata che permette di selezionare più righe, ed un filtro che permette di compiere ricerche in tempo reale sulla tabella immettendo il testo da cercare.
- VerticalLayout, contenitori di componenti che vengono ordinati verticalmente;
- HorizontalLayout, contenitori di componenti che vengono ordinati orizzontalmente;
- FormLayout, contenitori di componenti che vengono ordinati verticalmente, in modo che i loro nomi ed i loro valori siano allineati.

4.2.3.3. Responsiveness

Un requisito fondamentale che lo strato di User Interface deve soddisfare è l'usabilità attraverso dispositivi mobili, in particolare tablet. Per questo motivo l'applicazione è stata realizzata utilizzando componenti *responsive*, che si adattano alle dimensioni dello schermo sul quale vengono visualizzate, occupando dinamicamente tutto lo spazio a disposizione.

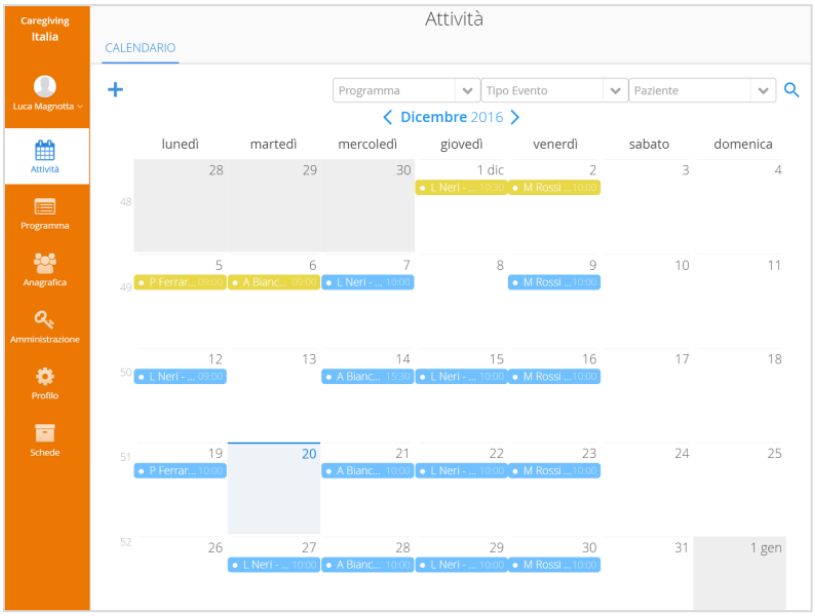


Figura 4.2.3.3.1 - Visualizzazione mensile su tablet

4.3. Ruoli e Permessi

L'Applicazione utilizza un meccanismo di Permessi per gli Utenti basato sul Role-Based Access Control, esteso all'Attribute-Based Access Control per implementare Permessi variabili in funzione del Programma.

4.3.1. Role-Based Access Control

Il Role-Based Access Control (RBAC) è un approccio a sistemi ad accesso ristretto per utenti autorizzati.

I Permessi per eseguire specifiche operazioni sono assegnate a specifici Ruoli. Poiché i Permessi non sono assegnati direttamente agli Utenti ma vengono acquisiti solo tramite i Ruoli ad essi assegnati, la gestione dei diritti individuali per un Utente diventa una semplice assegnazione dei Ruoli appropriati per l'Utente stesso.

Sono definite tre regole fondamentali per il modello RBAC:

1. Assegnazione dei Ruoli: un Utente può avere un Permesso solo se è stato assegnato ad un Ruolo.
2. Autorizzazione dei Ruoli: un Ruolo attivo per un Utente deve essere stato autorizzato per l'Utente. Insieme alla prima regola, questa garantisce che gli Utenti possano avere esclusivamente Ruoli ai quali sono stati autorizzati.
3. Autorizzazione del Permesso: un Utente può avere un Permesso solo se il Permesso è autorizzato per il Ruolo attivo dell'Utente. Insieme alla prima e alla seconda regola, questa garantisce che l'Utente possa avere unicamente Permessi ai quai è stato autorizzato.

4.3.1.1. Architettura del Role-Based Access Control

Il RBAC è implementato attraverso le tabelle UserRole e Permission del Database:

User_Role	
NOME COLONNA	TIPO DI DATO
id_user_role	Serial
role_name	Varchar
role_description	Varchar

Permission	
NOME COLONNA	TIPO DI DATO
id_permission	Serial
permission_description	Varchar

4.3.2. Attribute-Based Access Control

L'Attribute-Based Access Control (ABAC), definito anche Policy-Based Access Control è un'estensione del RBAC. A diffrerenza del RBAC, il Permesso viene definito sulla base di una serie di operazioni booleane tra tutte le Policy definite per quell'Utente. La gestione dei Permessi definita tramite l'ABAC è quindi dinamico e può essere ricondotto al contesto, come nel caso di uno specifico Permesso assegnato solo in corrispondenza di una determinata posizione geografica.

L'implementazione dell'ABAC dell'Applicazione è limitata ad un controllo basato unicamente sul Programma, per cui uno specifico Permesso può essere definito per un Utente solo per un set di Programmi.

4.3.2.1. Architettura dell'Attribute-Based Access Control

L'ABAC è implementato attraverso le tabelle Policy e Attribute del Database:

Policy	
NOME COLONNA	TIPO DI DATO
id_policy	Serial
owner_id	Integer
update_id	Integer
date_created	Timestamp with time zone
date_updated	Timestamp with time zone

Attribute

NOME COLONNA	TIPO DI DATO
id_attribute	Serial
attribute_name	Varchar
attribute_value	Varchar
attribute_label	Varchar

La tabella Attribute, in particolare, è alimentata con i dati dei Programmi grazie ad un trigger SQL che la aggiorna nel momento in cui viene inserito o modificato un Programma, siccome nell'implementazione dell'ABAC dell'Applicazione l'unico attributo utilizzato è il Programma stesso.

Nella tabella Attribute:

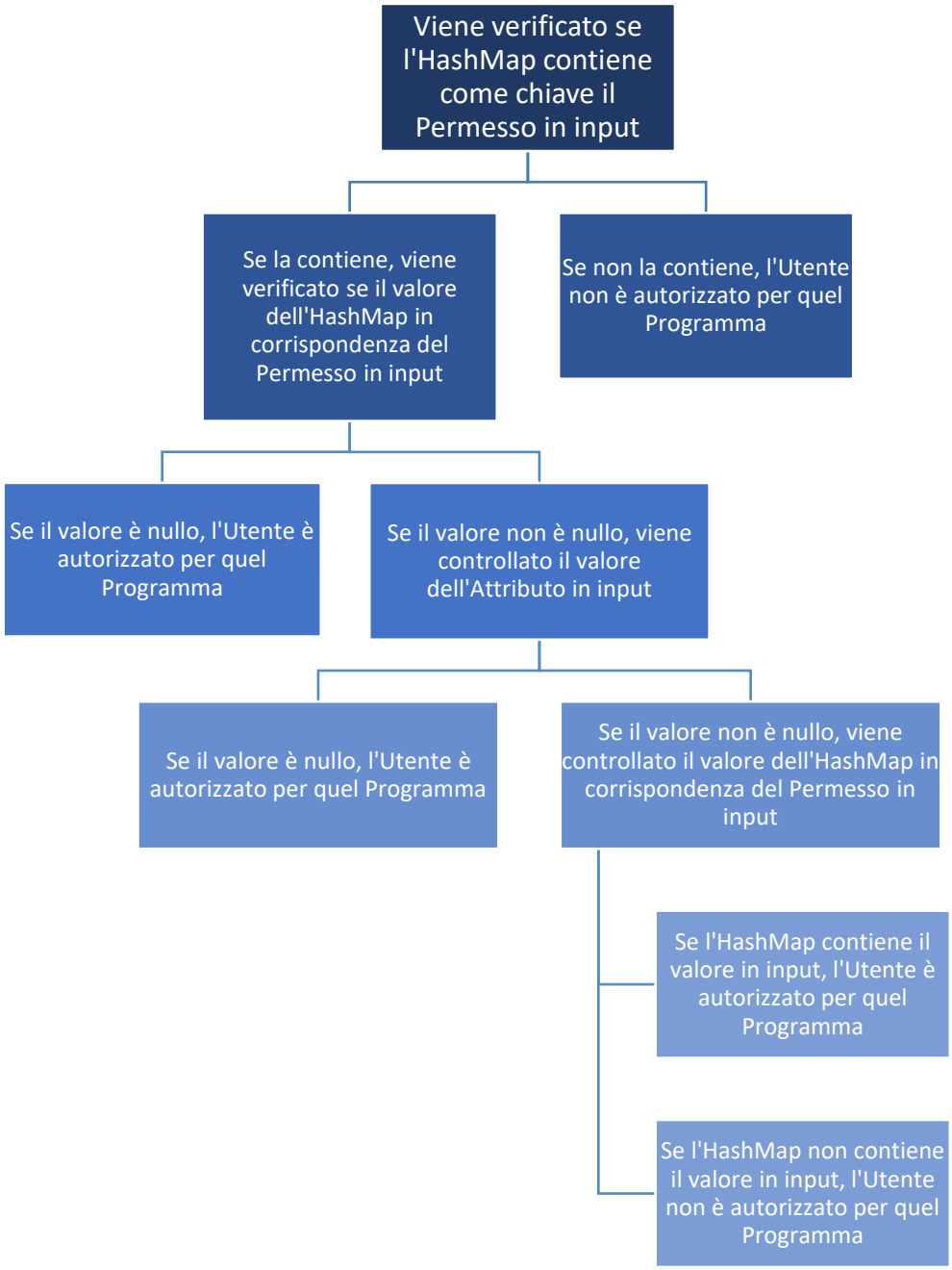
- Il campo **attribute_name** contiene una stringa che concatena il nome della tabella ed il nome della colonna che definisce l'identificativo dell'Attributo, che nell'implementazione dell'Applicazione risulta quindi essere sempre la stringa "program.program_id";
- Il campo **attribute_value** contiene il valore della colonna definita da **attribute_name**, che nell'implementazione dell'Applicazione è sempre l'identificativo del Programma;
- Il campo **attribute_label** contiene il nome della colonna che definisce il nome dell'Attributo, che nell'implementazione dell'Applicazione risulta quindi essere sempre la stringa "program_name".

4.3.3. Logica dei Ruoli e Permessi

La verifica dei Permessi viene effettuata dal metodo checkPermission della classe UserVisibility, che richiede in input la stringa del Permesso e la stringa del valore dell'Attributo, che è l'id del Programma per cui si vuole verificare il Permesso.

Viene generata una HashMap dei Permessi dell'Utente, avente come chiave gli identificativi dei Permessi e come valore la lista degli eventuali Attributi. Per convenzione, se il Permesso viene definito per tutti i Programmi, la lista è vuota.

Sulla base dei contenuti della HashMap:


Viene verificato se l'HashMap contiene come chiave il Permesso in input
Se la contiene, viene verificato se il valore dell'HashMap in corrispondenza del Permesso in input
Se non la contiene, l'Utente non è autorizzato per quel Programma
Se il valore è nullo, l'Utente è autorizzato per quel Programma
Se il valore non è nullo, viene controllato il valore dell'Attributo in input
Se il valore è nullo, l'Utente è autorizzato per quel Programma
Se il valore non è nullo, viene controllato il valore dell'HashMap in corrispondenza del Permesso in input
Se l'HashMap contiene il valore in input, l'Utente è autorizzato per quel Programma
Se l'HashMap non contiene il valore in input, l'Utente non è autorizzato per quel Programma

5. Componenti

5.1. Calendario

Gli Operatori e i Medici Specialisti possono gestire e monitorare le Attività dei propri Pazienti tramite un Calendario condiviso.

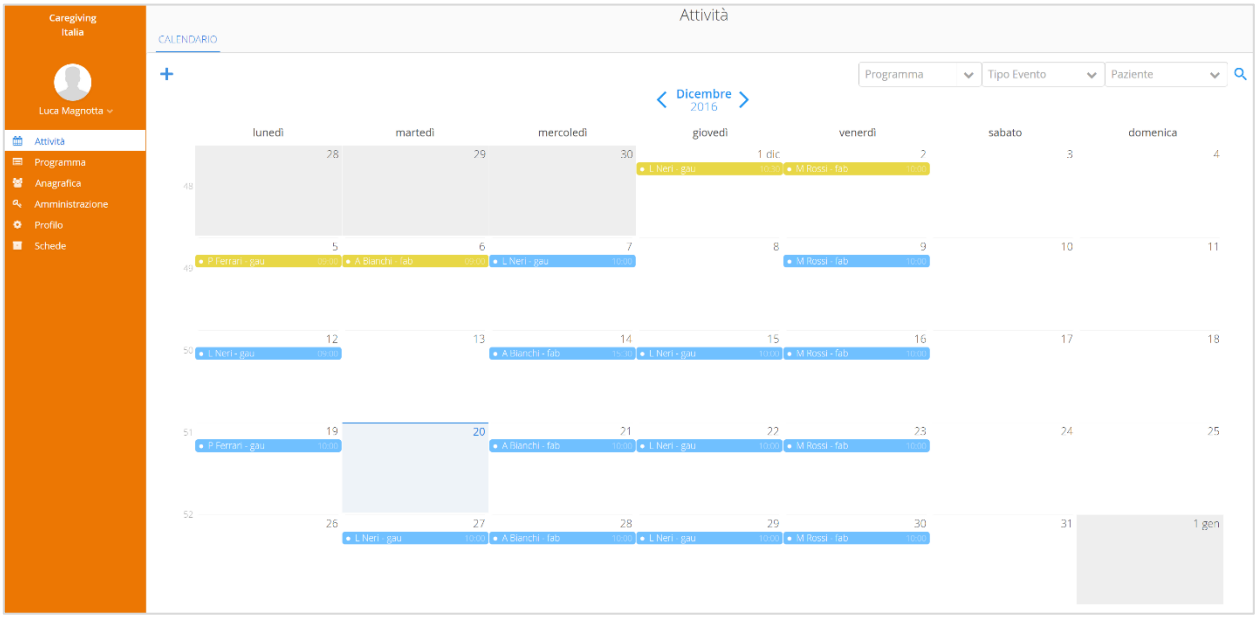


Figura 5.1.1 - Visualizzazione mensile su PC

Il Calendario mostra di default tutte le Attività visibili dall'Utente nel mese corrente, e per farlo richiede al Provider tutte le Entity Attività tra il primo e l'ultimo giorno del mese.

Le Attività in Calendario sono aggiornate quando:

- Si seleziona un mese diverso mediante le due frecce a fianco del mese nella parte superiore dell'interfaccia;
- Viene aggiunta, modificata o eliminata un'Attività. In modo da poterla visualizzare subito;
- Viene applicato un filtro.

Non si richiedono mai Attività per un tempo superiore al mese, in modo tale da diminuire i tempi di risposta dell'applicazione nel caricamento dei dati.

5.1.1. Tipologie e Visibilità delle Attività

Le Attività presenti in Calendario riguardano un Paziente e un Operatore all'interno di un PSP, e sono caratterizzate da una data ed un orario di inizio e di fine.

Il Calendario presenta quattro differenti tipi di Attività:

- **Prese in Carico**, è la prima visita propedeutica all'inserimento del Paziente stesso nel Programma di Assistenza Domiciliare;
- **Visita Domiciliare**, rappresenta la visita periodica presso il domicilio del Paziente prevista dal PSP ;
- **Indisponibilità**, giornata o intervalli orari in cui l'Operatore è indisponibile;
- **Ferie**, è l'evento usato dagli Operatori per indicare i propri giorni di ferie.

Il Calendario raccoglie e centralizza le Attività di Utenti con ruoli e privilegi differenti. Gli Operatori e i Medici possono condividere le Attività dei Pazienti che seguono entrambi e vedere in tempo reale eventuali modifiche dei dati di cui hanno l'autorizzazione. In particolare:

- gli **Operatori** possono vedere e modificare solo le Attività dei pazienti a loro assegnati;
- i **Medici Specialisti** possono visualizzare tutte le Attività delle tipologie Presa in Carico e Visita Domiciliare dei pazienti che seguono presso il Centro Clinico, ma non sono autorizzati ad effettuare modifiche degli eventi che visualizzano.

5.1.2. Navigazione del Calendario

Il Calendario viene visualizzato di default nella versione mensile, dove è possibile passare al mese precedente o a quello successivo. Questo tipo d'interazione con l'interfaccia grafica richiede che il Provider e i relativi Service eseguano un'operazione sul Database per recuperare le Attività del mese da visualizzare in Calendario. Cliccando sul numero della settimana, si passa alla visualizzazione settimanale, in cui gli eventi sono aggregati per quella determinata settimana in un intervallo temporale compreso dal lunedì alla domenica (Figura 8).

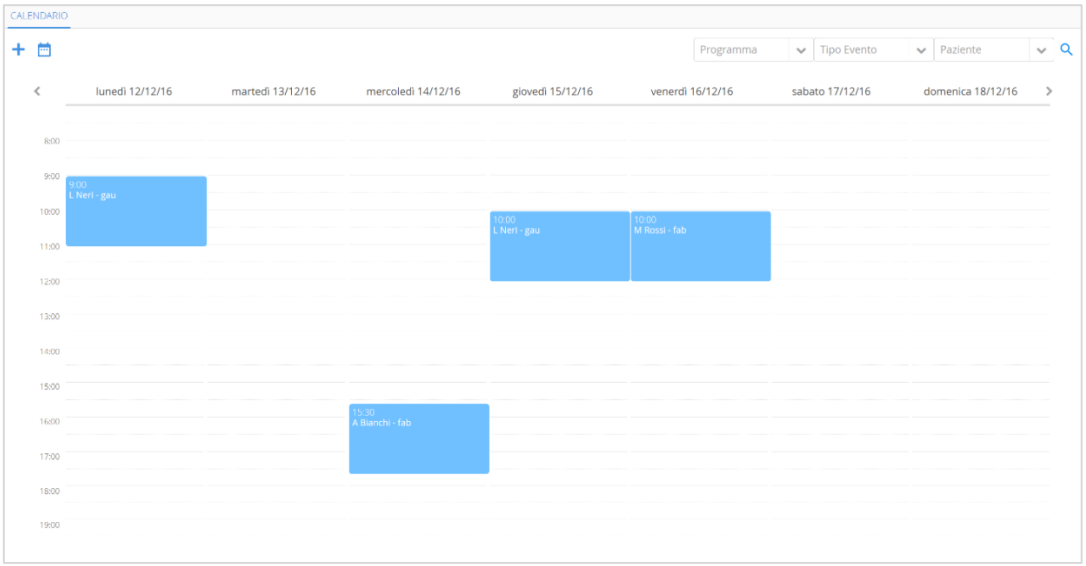


Figura 5.1.2.1 - Visualizzazione settimanale

Cliccando su un giorno, si passa alla visualizzazione giornaliera, che mostra con maggiore dettaglio le Attività di quella giornata con intervallo orario, come rappresentato in Figura 9.

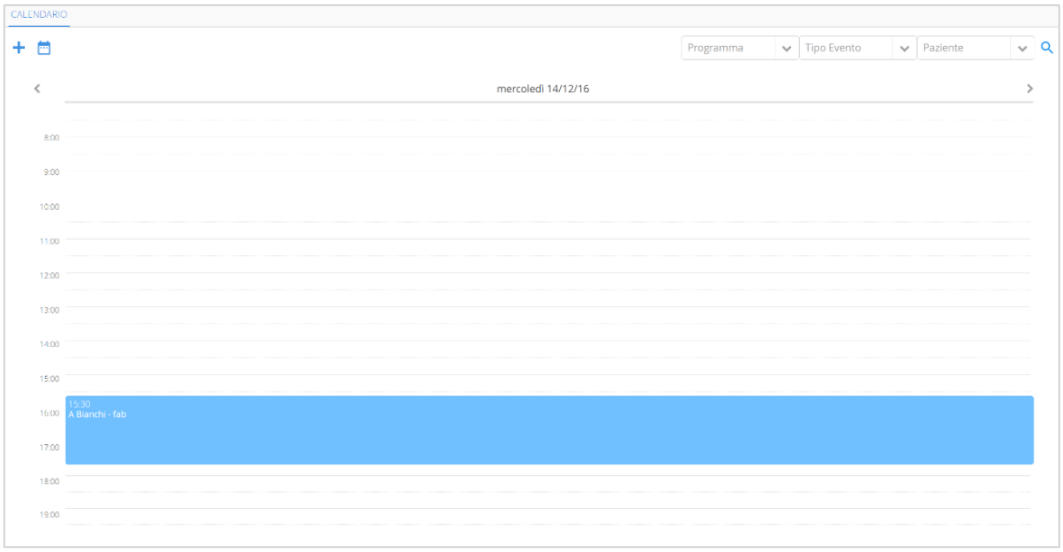


Figura 5.1.2.2 - Visualizzazione giornaliera

Siccome i dati delle Attività che vengono caricati dai Service e tenuti in memoria si riferiscono ad un periodo di un mese, il cambiamento di visualizzazione del Calendario non richiede un ulteriore accesso al Database. In questo modo il cambiamento di visualizzazione avviene in maniera rapida ed efficiente.

5.1.3. Filtri del Calendario

Il Calendario permette di filtrare gli eventi in base al PSP, alla tipologia di Attività e al Paziente.

I filtri sono rappresentati da tre menù a tendina nella parte superiore destra della schermata del Calendario. È possibile selezionare uno, due o anche tutti e tre i filtri contemporaneamente, e vedere i loro effetti combinati.

In particolare, il filtro sul Paziente e sul Tipo di Attività mostrano solo i Pazienti e le Attività del PSP selezionato, e non danno nessuna possibilità di scelta se prima non è selezionato un Programma Assistenziale.

L'applicazione dei filtri implica un solo accesso al Database che avviene attraverso il Data Provider a prescindere dal numero di filtri applicati, poiché la funzione che si occupa del reperimento dei dati esegue una query dove vengono specificati in modo dinamico i campi che servono per soddisfare i criteri di ricerca indicati dai filtri stessi.

5.1.4. Attività

A partire dal Calendario è possibile creare una nuova Attività accedendo alla finestra mostrata in Figura 10.

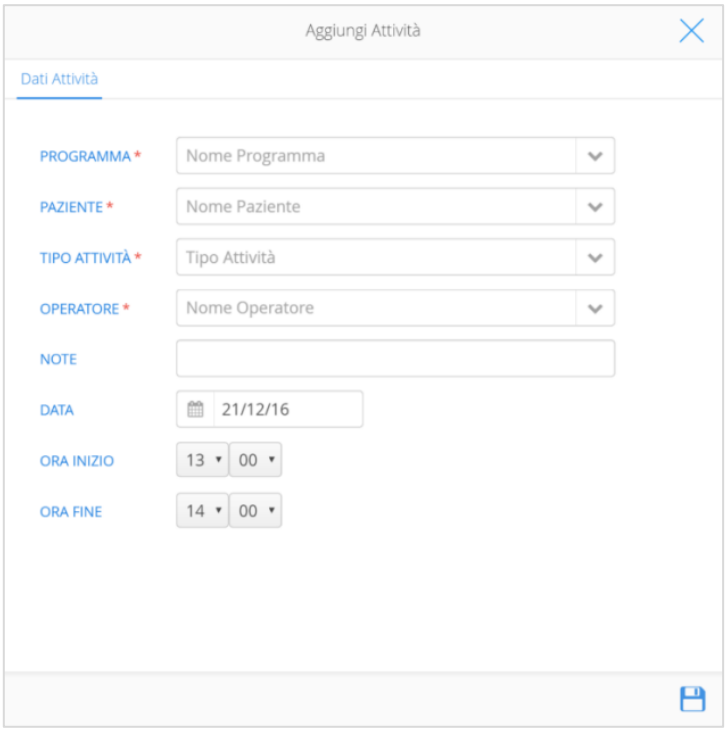


Figura 5.1.4.1 - Finestra per la creazione di un'Attività

I quattro campi iniziali sono obbligatori, per garantire la consistenza dei dati. Come per i filtri, la scelta del Paziente, del Tipo di Attività e dell'Operatore sono subordinate a quella del PSP. Se l'Utente che sta inserendo un'Attività è un Operatore, il campo di scelta dell'Operatore sarà non modificabile e impostato sull'Utente stesso. Al momento del salvataggio, viene chiamato un Service che si occupa dell'inserimento della nuova Attività nel Database. In seguito il Provider ricarica tutte le Attività del mese, in modo da mostrare anche quella appena inserita.

Selezionando un'Attività da Calendario, l'Operatore può accedere alla schermata di Figura 11, dove sono presenti i dettagli dell'Attività stessa, e può modificarne alcuni campi. Le modifiche riguardano unicamente le Note, la Data e l'Orario d'Inizio e Fine di un'Sttività, non è quindi possibile cambiare Programma, Paziente, Tipo di Attività ed Operatore.

Sempre da questa finestra è possibile eliminare un'Attività. Prima della cancellazione effettiva dell'Attività compare una finestra di conferma: richiedendo l'eliminazione, l'Utente è riportato al Calendario e gli viene notificata l'avvenuta operazione. La modifica e l'eliminazione di un'Attività avviene mediante una chiamata a un Service, susseguito dal ricaricamento da parte del Provider di tutte le Attività del mese, in modo da mostrare anche la modifica o l'eliminazione dell'Attività.

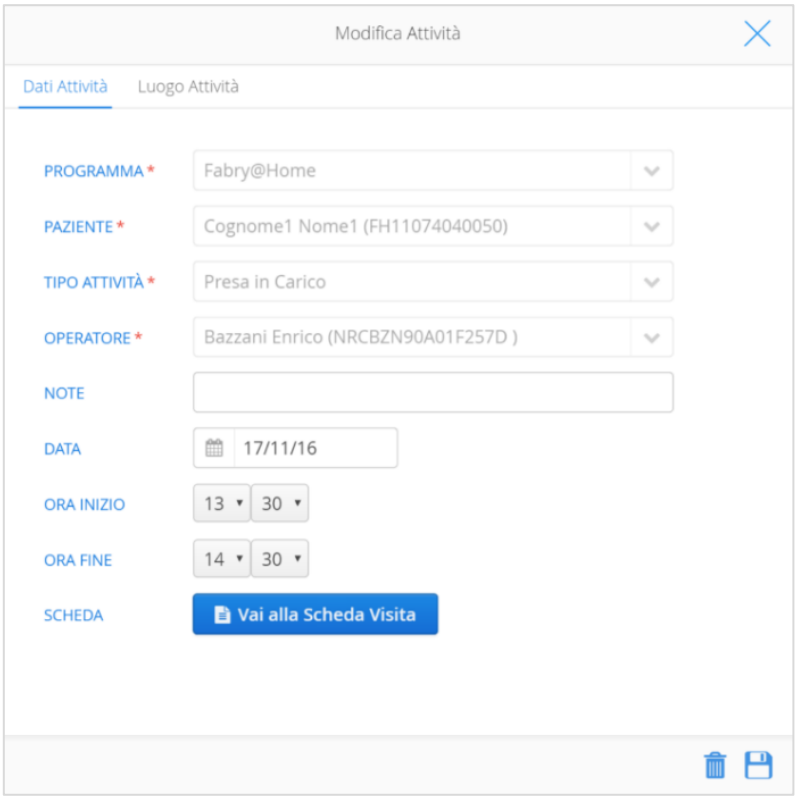


Figura 5.1.4.2 - Finestra per la modifica di un'Attività

5.1.4.1. Architettura delle Attività

Le Attività sono salvate in un unica tabella nel Database:

Activity	
NOME COLONNA	TIPO DI DATO
id_activity	Serial
start_date	Timestamp with time zone
end_date	Timestamp with time zone
comments	Varchar
video_visit	Boolean
video_visit_url	Varchar
id_program	Integer
id_activity_type	Integer
id_operator	Integer
id_patient	Integer
deleted	Boolean

In particolare, per ogni Attività, sono salvati:

- **start_date**, che contiene data ed ora di inizio dell'Attività;

- end_date, che contiene data ed ora di fine dell'Attività;
- comments, una stringa libera che permette il salvataggio di note legate alle singole Attività;
- video_visit, un booleano che vale "true" se per quell'Attività è prevista una Video-Visita;
- video_visit_url, che contiene il codice di accesso alla *waiting room* di Vsee, nel caso sia prevista la Video-Visita;
- id_program, che fa riferimento all'id del Programma Assistenziale a cui appartiene l'Attività;
- id_activity_type, che fa riferimento all'id del Tipo di Attività, che a sua volta dipende dal Programma Assistenziale (diversi Programmi Assistenziali determinano diversi Tipi di Attività);
- id_operator, che fa riferimento all'id dell'Operatore associato a quell'Attività, che a sua volta dipende dal Programma Assistenziale (gli Operatori possono interagire con un'Attività se appartengono al suo Programma Assistenziale e se sono stati associati al suo Paziente);
- id_patient, che fa riferimento all'id del Paziente associato a quell'Attività, che a sua volta dipende dal Programma Assitenziale (diversi Pazienti sono associati a diversi Programmi Assistenziali).

5.1.4.2. Logica delle Attività

Le Attività sono richeste dal pannello del Calendario al Provider, specificando:

- Data iniziale delle Attività;
- Data finale delle Attività;
- La lista di Tipi di Attività desiderate (in quanto alcuni Utenti possono non avere i Permessi per visualizzarle tutte);
- L'id del Programma per cui visualizzare le Attività (se nullo, verranno reperite le Attività di tutti i Programmi);
- L'id del Paziente per cui visualizzare le Attività (se nullo, verranno reperite le Attività di tutti i Pazienti);
- L'id dell'Operatore per cui visualizzare le Attività (se nullo, verranno reperite le Attività di tutti gli Operatori);
- L'id dell'Utente corrente, in modo tale da visualizzare solo le Attività di Pazienti a cui l'Utente corrente è assocuato (se nullo, verranno reperite le Attività di tutti i Pazienti, anche non associati).

Il Provider, a sua volta, richiede le Attività con gli stessi parametri al Service, e dopo averle ottenute, per ciascuna:

- Attribuisce un colore all'Attività sulla base del Tipo di Attività;

- Attibuisce una *label* all'Attività sulla base del Paziente e del Programma, mostrando la prima lettera del nome del Paziente, il suo intero cognome e, separate con un trattino, le prime tre lettere del Programma Assistenziale, ad esempio:
    - C Tarantino – FAB

  O, in caso di Paziente non selezionato:

    - Paziente Assente – GAU

Il Service, a sua volta, richiede le Attività con gli stessi parametri al DAO, e dopo averle ottenute, per ciascuna:

- A partire dall'id del Tipo di Attività recupera il record di Activity Type e lo assegna al Bean dell'Attività.

Il DAO, ricevuta la richiesta dal Service, esegue una query modulare che, sulla base di condizioni logiche poste rispetto ai parametri opzionali ricevuti (lista di Tipi di Attività, id di Programma, Paziente, Operatore e Utente corrente) definisce più o meno restrizioni, e quindi recupera più o meno Attività.

Dopo l'esecuzione della query, vengono creati i Bean delle Attività recuperate e vengono alimentati con i dati ottenuti.

5.1.4.3. Geolocalizzazione delle Attività

Accedendo alla finestra di un'Attività è presente una sezione con una mappa generata attraverso Google Maps (Figura 12) corrispondente all'indirizzo di domicilio del Paziente. La mappa è navigabile direttamente dalla finestra dell'Attività, ed è possibile richiedere il calcolo del percorso più veloce per raggiungere il domicilio del Paziente.

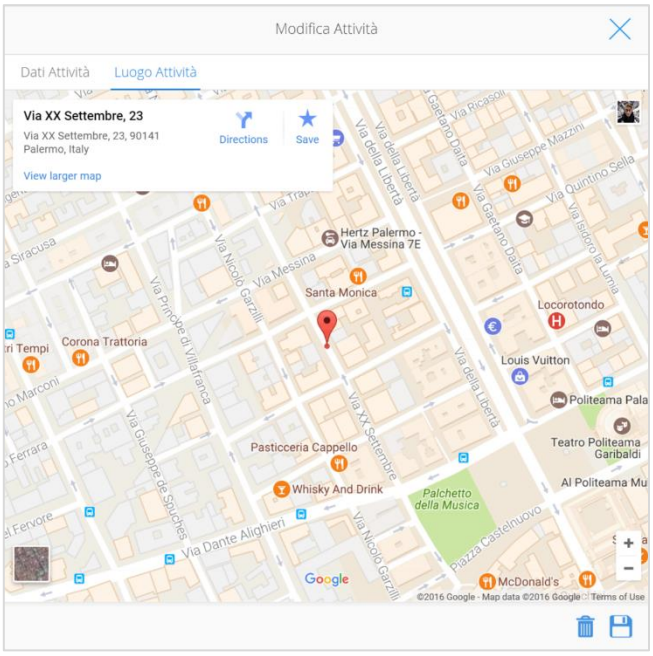


Figura 5.1.4.3.1 – Finestra di visualizzazione del domicilio del Paziente

5.1.5. Attività Ripetute

Al momento della creazione di una Attività, l'Utente ha a disposizione la possibilità di ripetere quell'Attività spuntado la CheckBox "Ripeti":

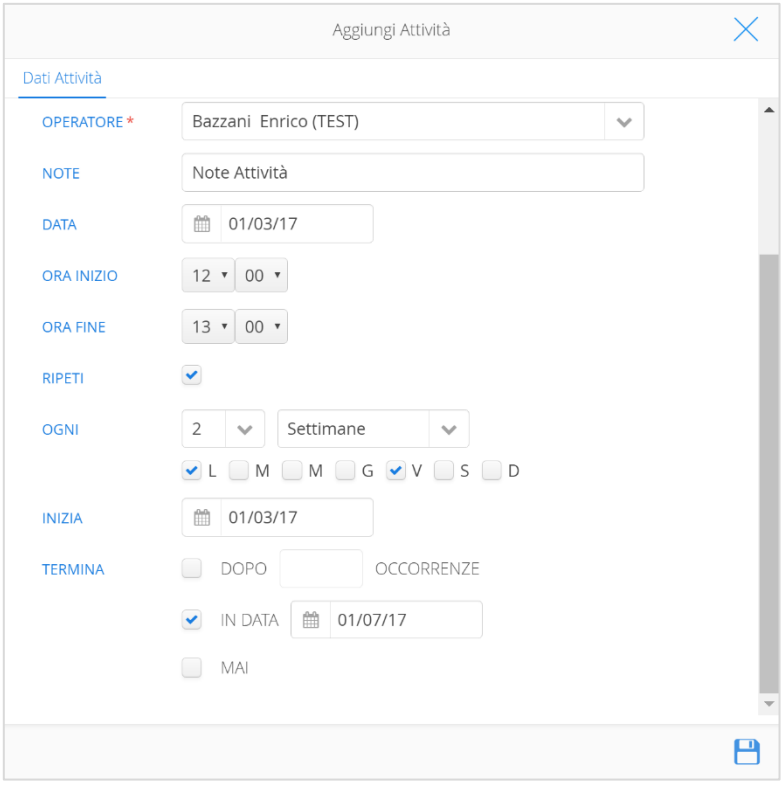


Figura 5.1.5.1 - Finestra di Aggiunta Attività con selezione su Attività Ripetuta

In questo modo compare il menù di configurazione della Ripetizione, che si compone di tre parti:

- La selezione della periodicità dell'Attività, che avviene specificando la quantità di giorni, settimane, mesi o anni dopo cui si desidera ripetere l'Attività;
- La selezione della data di inizio, che di default è settata al giorno scelto per l'Attività;
- La selezione del termine della Ripetizione, che può essere effettuata:
    - Specificando il numero totale di occorrenze;
    - Selezionando una data di fine;
    - Selezionando "Mai". In questo modo l'Attività verrà ripetuta un numero indefinito di volte, fino a che l'Utente non ne interromperà la Ripetizione.

Salvando la nuova Attività, il sistema provvede a generare le successive Attività Ripetute e mostrarle immediatamente nel Calendario.

Nel momento in cui si modifica un'Attività Ripetuta, viene chiesto all'Utente se:

- Modificare una sola occorrenza di quella Attività, lasciando invariate le altre occorrenze;
- Modificare tutte le occorrenze successive di quella Attività.

5.1.5.1. Architettura delle Attività Ripetute

La necessità di ripetere un'Attività per un numero indefinito di volte rende assolutamente impossibile inserire nel Database ogni singola Ripetizione come un record della tabella Activity, poichè per le Ripetizioni indefinite si dovrebbero aggiungere un numero infinito di record.

L'approccio utilizzato è quindi di tipo generativo: in una tabella separata, chiamata Activity Repeat, sono salvate le regole di Ripetizione per le singole Attività. La funzione DAO che si occupa di reperire le Attività, esegue un join con la tabella Activity Repeat e, sulla base dei risultati, genera tutti i Bean delle Attività presenti nel mese, comprese le Ripetizioni.

ActivityRepeat	
NOME COLONNA	TIPO DI DATO
id_activity_repeat	Serial
id_activity	Integer
repeat_start_date	Timestamp with time zone
repeat_end_occurrences	Integer
repeat_frequency_number	Integer
repeat_frequency_period	Varchar
repeat_frequency_days	Varchar
repeat_end_date	Timestamp with time zone
owner_id	Integer
update_id	Integer
date_created	Timestamp with time zone

date_updated	Timestamp with time zone
deleted	Boolean

La tabella Activity Repeat salva le regole di Ripetizione di una Attività attraverso:

- id_activity, che fa riferimento all'id dell'Attività originale;
- repeat_start_date, che contiene la data di inizio delle Ripetizioni, che può essere diversa dalla data dell'Attività originale;
- repeat_frequency_period, che esprime il tipo di intervallo tra una Ripetizione e la successiva, può valere:
    - D, day;
    - W, week;
    - M, month;
    - Y, year.
- repeat_frequency_number, che esprime la quantità degli intervalli di tempo definiti da repeat_frequency_period tra una Ripetizione e la successiva;
- repeat_frequency_days, che esprime i giorni della settimana selezionati in caso di Ripetizione settimanale. È composto da 7 lettere o zeri, che rappresentano rispettivamente i giorni della settimana di Ripetizione o la non-Ripetizione in quel giorno, intervallati da trattini. Ad esempio:
    - 0-T-0-T-F-0-0 rappresenta la Ripetizione Martedì (T), Giovedì (T) e Venerdì (F);
    - M-0-W-0-0-0-0 rappresenta la Ripetizione Lunedì (M) e Mercoledì (W);
    - 0-0-0-0-0-S-S rappresenta la Ripetizione Sabato (S) e Domenica (S).
- repeat_end_occurrences, che contiene il numero di Ripetizioni da generare per quell'Attività ed è settato solo se l'Utente ha selezionato il numero di occorrenze;
- repeat_end_date, che contiene la data di fine delle Ripetizioni.

Sulla base dei valori presenti nei campi repeat_end_date e repeat_end_occurrences viene poi definito il termine delle Ripetizioni, secondo la seguente gerarchia logica:

- Se repeat_end_occurrences non è nullo, la Ripetizione viene effettuata per il numero di occorrenze specificate. Viene comunque calcolata una data di fine nel DAO, per facilitare le operazioni successive;
- Se repeat_end_occurrences è nullo e repeat_end_date non è nullo, la Ripetizione viene effettuata fino alla data specificata;
- Se repeat_end_occurrences e repeat_end_date sono entrambi nulli, le Ripetizioni sono generate all'infinito.

5.1.5.2. Logica delle Attività Ripetute

Le Attività Ripetute vengono generate dal DAO delle Attività sulla base dei risultati della stessa *query* che recupera dal Database le Attività.

La *query* esegue un *join* tra i *record* di Activity ed i *record* di Activity Repeat sulla *foreign key* che da Activity Repeat punta ad Activity.

Siccome una Ripetizione può occorrere anche a mesi o anni di distanza, è stata posta una condizione sulla data di inizio e fine delle Attività in base al fatto che abbiano Ripetizioni, in particolare:

- Se l'Attività Ripetuta è nulla (ovvero non esiste) e la data dell'Attività è compresa tra la data di inizio e fine in *input* alla *query*, l'Attività viene selezionata;
- Se invece l'Attività Ripetuta non è nulla, sia l'Attività che l'Attività Ripetuta vengono selezionate.

In questo modo, se esiste un'Attività Ripetuta che fa riferimento ad una Attività che cade in un mese precedente a quello che si sta visualizzando, viene comunque selezionata dalla *query*, ma sarà poi scartata dalla logica del DAO, mentre le sue Ripetizioni saranno valutate una ad una.

Per ogni elemento del *result set* che presenta valori non nulli nei campi che dipendono da Activity Repeat (*id_activity_repeat*, *repeat_start_date*, *repeat_frequency_number* e *repeat_frequency_period*) vengono generate le Attività Ripetute sulla base delle regole definite dai campi di ActivityRepeat:

- Se la fine della Ripetizione è espressa in termini di occorrenze (ovvero *repeat_end_occurrences* non è nullo), una funzione la traduce in una data di fine:
    - La data di fine viene inizializzata alla data di inizio della Ripetizione;
    - Per ogni Occorrenza, la data di fine viene incrementata di tanto tempo quanto espresso dalla combinazione di *repeat_frequency_number* e *repeat_frequency_period*.
      Ad esempio, se *repeat_frequency_number* valesse 2 e *repeat_frequency_period* valesse "M", per ogni Occorrenza la data di fine verrebbe incrementata di due mesi.
- Attraverso un ciclo, viene generato un nuovo ActivityBean alla volta, con i campi *startDate* ed *endDate* calcolati sulla base dei campi *repeat_frequency_number* e *repeat_frequency_period* dell'ActivityRepeat:
    - Il calcolo della data e degli orari di inizio e fine del nuovo ActivityBean viene eseguito da un'apposita funzione che si occupa di calcolare l'occorrenza successiva sulla base di quella attuale e delle regole di Activity Repeat;
    - La funzione gestisce anche i casi di Ripetizione settimanale nei giorni specificati dall'utente (ad esempio: ogni due settimane, solo il Martedì ed il Giovedì).

- Gli altri campi del nuovo ActivityBean sono uguali a quelli dell'ActivityBean originario;
- Il nuovo ActivityBean viene segnalato come Ripetizione tramite un apposito valore booleano, in modo che la logica del Calendario possa poi distinguere l'Attività originario e le sue Ripetizioni;
- Se la data di inizio e fine del nuovo ActivityBean è compresa tra la data di inizio e fine in *input* alla *query*, viene aggiunto alla lista di ActivityBean restituita;
- Se la data di inizio e fine del nuovo ActivityBean non è compresa tra la data di inizio e fine in *input* alla *query*, viene scartato e la generazione di ActivityBean basate sulla regola di ActivityRepeat corrente viene interrotta

5.1.5.3. Materializzazione delle Attività Ripetute

Siccome le Attività Ripetute che popolano il Calendario corrispondono effettivamente a dei Bean Attività generati dal DAO, ma che non hanno un riferimento specifico nel Database, non è possibile interagirvi direttamente.

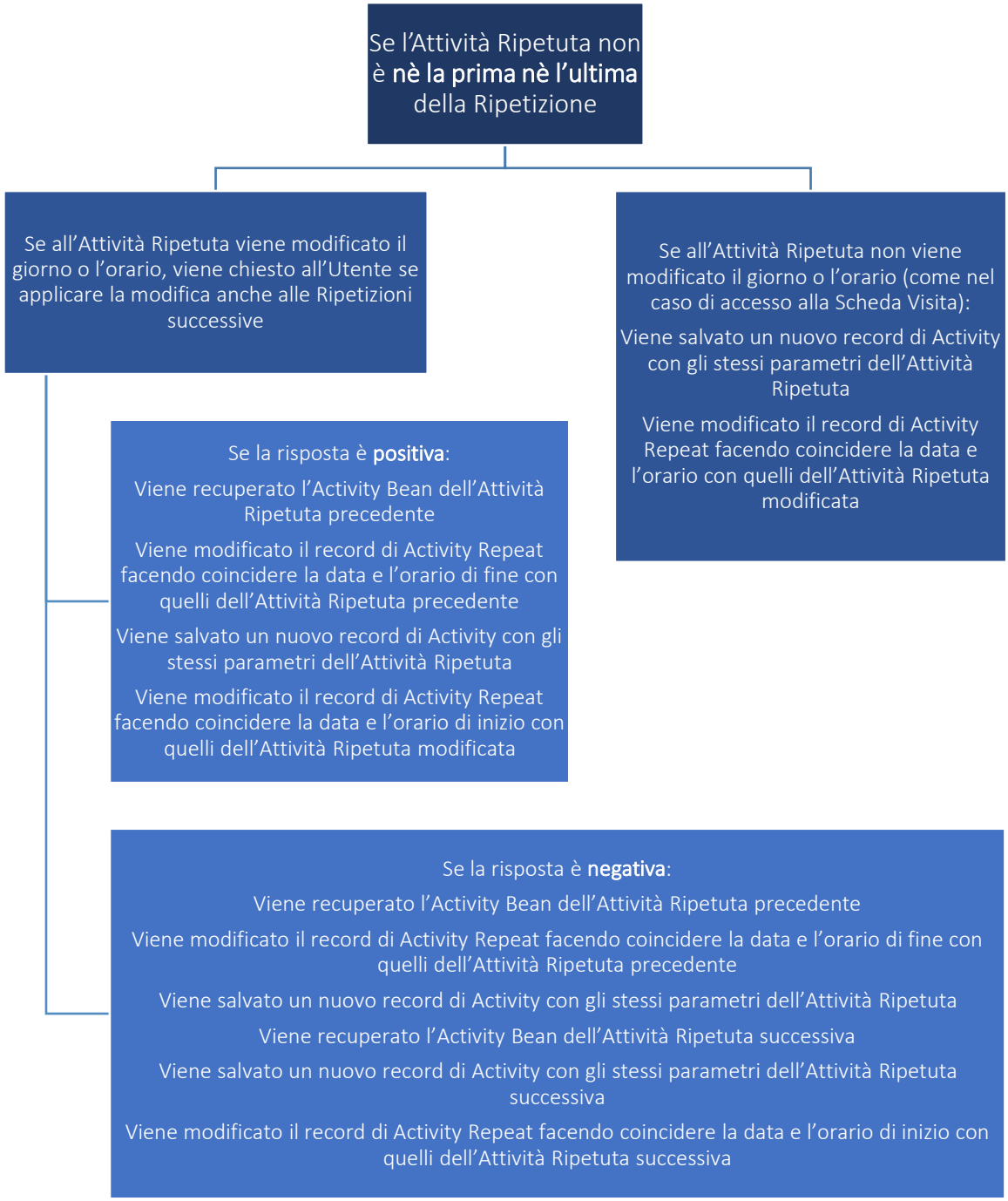
Per poter modificare e accedere alla Scheda Visita delle Attività Ripetute è stato introdotto il concetto di "Materializzazione", ovvero l'aggiunta di un nuovo record che faccia riferimento alla specifica Attività Ripetuta nella tabella Activity e la conseguente modifica della struttura della Ripetizione, in modo da mantenerla visivamente coerente.

La Materializzazione è un processo assolutamente trasparente per l'Utente, che avviene quando:

- L'Utente accede alla Scheda Visita di una Attività Ripetuta;
- L'Utente modifica un'Attività Ripetuta;
- L'Utente sposta un'Attività Ripetuta;

Secondo le seguenti casistiche:


Se l'Attività Ripetuta è la **prima** della Ripetizione
Se all'Attività Ripetuta viene modificato il giorno o l'orario, viene chiesto all'Utente se applicare la modifica anche alle Ripetizioni successive
Se la risposta è **positiva**:
Viene salvato un nuovo record di Activity con gli stessi parametri dell'Attività Ripetuta
Viene modificato il record di Activity Repeat facendo coincidere la data e l'orario di inizio con quelli dell'Attività Ripetuta modificata
Se la risposta è **negativa**:
Viene salvato un nuovo record di Activity con gli stessi parametri dell'Attività Ripetuta
Viene recuperato l'Activity Bean dell'Attività Ripetuta successiva
Viene salvato un nuovo record di Activity con gli stessi parametri dell'Attività Ripetuta successiva
Viene modificato il record di Activity Repeat facendo coincidere la data e l'orario con quelli dell'Attività Ripetuta successiva
Se all'Attività Ripetuta non viene modificato il giorno o l'orario (come nel caso di accesso alla Scheda Visita):
Viene salvato un nuovo record di Activity con gli stessi parametri dell'Attività Ripetuta
Viene modificato il record di Activity Repeat facendo coincidere la data e l'orario di inizio con quelli dell'Attività Ripetuta modificata
Se l'Attività Ripetuta è l'**ultima** della Ripetizione
Viene salvato un nuovo record di Activity con gli stessi parametri dell'Attività Ripetuta
Viene recuperato l'Activity Bean dell'Attività Ripetuta precedente
Viene modificato il record di Activity Repeat facendo coincidere la data e l'orario di fine con quelli dell'Attività Ripetuta precedente



5.1.5.4. Esempio di Funzionamento delle Attività Ripetute

Viene mostrato un esempio di come i dati presenti sul Database, nelle tabelle Activity ed ActivityRepeat, vegnono effettivamente utilizzati per generare le Attività Ripetute sul Calendario.

Di seguito è riportato il risultato di una query che, a scopo esemplificativo, contiene:

- Un record di una Attività senza Ripetizioni (id 531)
- Un record di una Attività con 5 Ripetizioni giornaliere, specificate come Occurrences (id 532)

- Un record di una Attività con 5 Ripetizioni giornaliere, specificate come End Date (id 533)
- Un record di una Attività ripetuta ogni Mercoledì e Vernerdì, senza una specifica fine

	id_activity integer	start_date timestamp with tim	end_date timestamp with tim	comments character varying(800)	id_ar integer	repeat_start_date timestamp with tim	f_number integer	f_period character varying(60)	f_days character varying(60)	repeat_end_occurrences integer	repeat_end_date timestamp with tim
1	533	2017-03-21 11:00	2017-03-21 12:00	Daily Repetitions - End date	302	2017-03-21 11:00	1	D			2017-03-25 11:00
2	534	2017-03-28 11:00	2017-03-28 12:00	Weekly Repetitions - Unlimited	303	2017-03-28 11:00	1	W	0-0-W-0-F-0-0		
3	531	2017-03-07 11:00	2017-03-07 12:00	Single Activity							
4	532	2017-03-14 11:00	2017-03-14 12:00	Daily Repetitions - End occurrences	301	2017-03-14 11:00	1	D		5	

Figura 5.1.5.4.1 - Query delle Attività Ripetute

Sono state evidenziate in blu le colonne del result set che dipendono dalla tabella Activity, ed in arancione quelle che dipendono dalla tabella Activity Repeat. Quando una Attività non presenta ripetizioni, il *join* con la tabella Activity Repeat è nullo e di conseguenza le righe evidenziate in arancione sono vuote.

Sulla base dei risultati della query vengono quindi riportate sul calendario le Attività Originali e generate le corrispondenti Attività Ripetute, come segue:

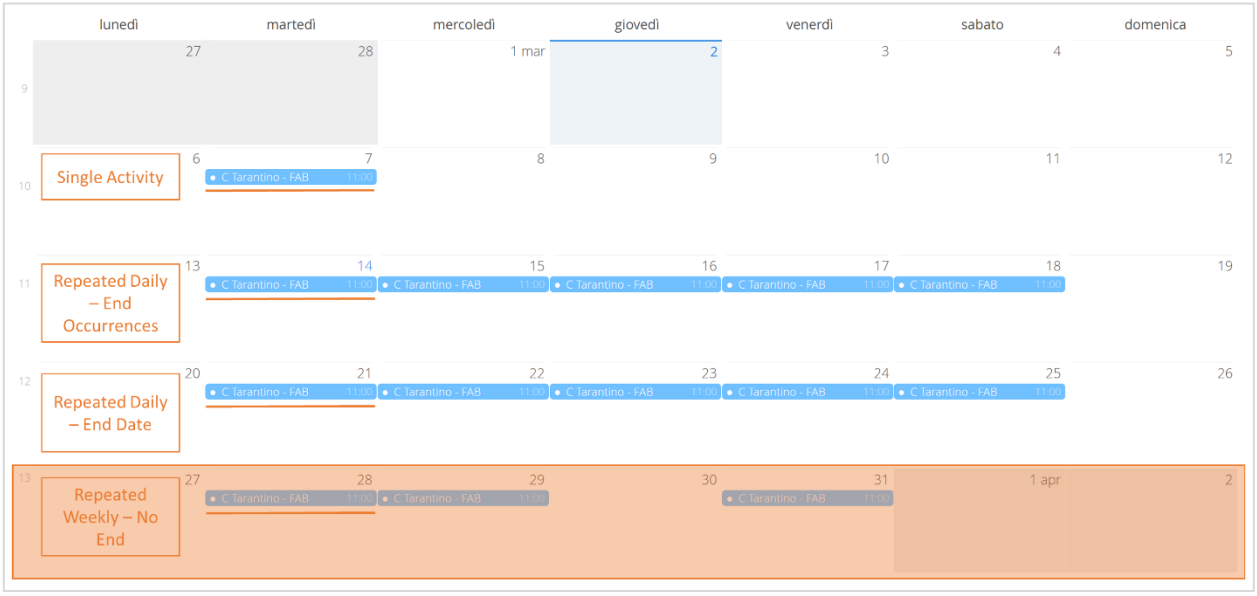


Figura 5.1.5.4.2 - Trasposizione dei risultati della query in Attività del Calendario

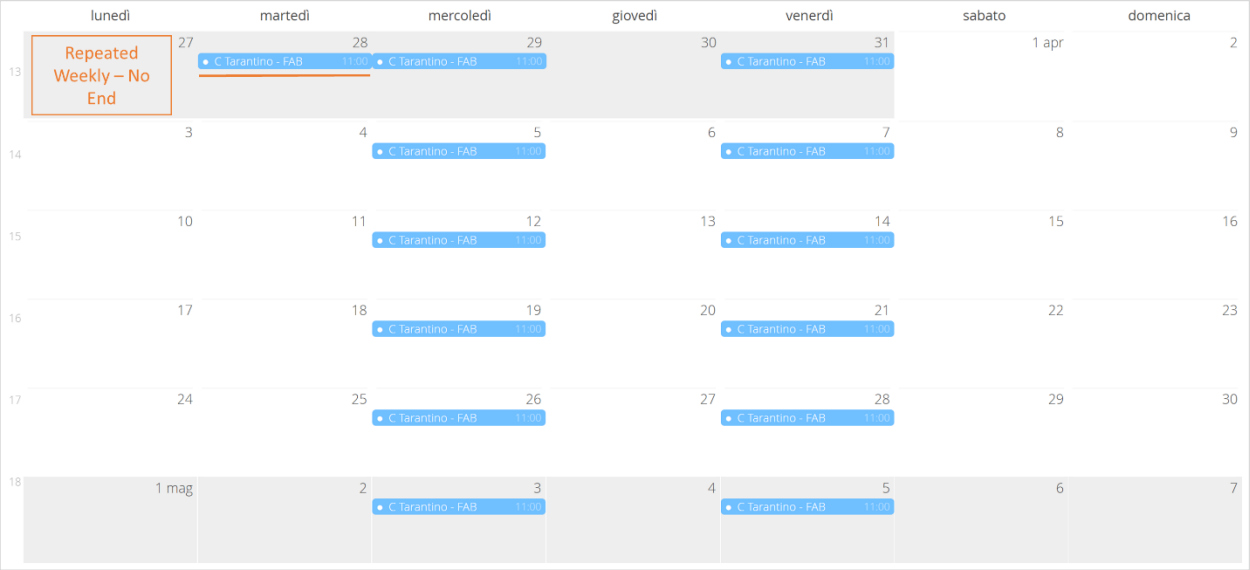


Figura 5.1.5.4.3 - Trasposizione dei risultati della query in Attività del Calendario – 2

Sono state sottolineate in arancione le Attività che corrispondono ad un record di Activity (e sono dunque Originali), mentre le altre sono Attività Ripetute generate dinamicamente.

5.1.5.5. Prestazioni delle Attività Ripetute

Per valutare la scalabilità delle prestazioni in presenza di un numero cospicuo di Attività ripetute, è stato eseguito un test in cui due Attività sono state ripetute un numero crescente di volte, nell'arco di un mese.

In particolare, si sono testati i seguenti casi:

- Nessuna Ripetizione
- 2 Attività Ripetute ogni 6400 secondi, ovvero circa 2 Attività ogni 2 ore
- 2 Attività Ripetute ogni 3200 secondi, ovvero circa 2 Attività ogni ora.
- 2 Attività Ripetute ogni 60 secondi, ovvero un'Attività ogni minuto

Dei casi presi in esame, i due intermedi si riferiscono a casi d'uso tipici per la maggior parte degli Utenti, in particolare per quelli che visualizzeranno solo le Attività proprie o del proprio Gruppo. L'ultimo caso va certamente oltre le richieste, anche in caso di picco, che l'applicazione si troverà a dover gestire ed è stato realizzato solo a scopo dimostrativo.

Le rilevazioni sono state eseguite a gruppi di 5 per ogni caso, ed in particolare:

- La prima rilevazione è stata effettuata subito dopo il lancio dell'applicazione
- Le successive quattro sono state effettuate ricaricando la pagina del Calendario, e tendono a mostrare risultati migliori per via del *caching* effettuato dal browser.

Si è poi verificato che durante l'esperienza d'uso le prestazioni sono notevolmente migliori rispetto a quelle prospettate dai test, in quanto in mancanza di un refresh completo delle pagine,

l'applicazione ottiene unicamente i dati necessari tramite richieste AJAX al Server, rendendo la navigazione molto fluida. I dati rilevati sono dunque da intendersi come *worst-case-scenario* legati alla completa ricarica di tutti i dati dell'applicazione.

Si sono fatte tre valutazioni distinte del tempo:

- Il Tempo DAO è stato rilevato dall'inizio alla fine del metodo *getActivities* della classe *ActivityDao*, e misura esattamente quanto il sistema impiega ad eseguire le query ed a generare i Bean delle Attività;
- Il Tempo CalendarView è stato rilevato dall'inizio alla fine della creazione dell'interfaccia del Calendario, e misura quanto il sistema impiega a contattare il Service e poi il DataProvider per alimentare il Calendario con le Attività e poi a generare l'interfaccia;
- Per via dei ritardi dovuti al *rendering* grafico di tutte le Attività in Calendario (specialmente nel caso assolutamente eccezionale di un'Attività al minuto), Vaadin impiega effettivamente un tempo superiore a quello misurato nella generazione della CalendarView, pertanto è stato misurato manualmente anche il Tempo Cronometro, che dà un'idea di quanto dovrà attendere l'Utente alla prima visualizzazione della schermata del Calendario.

Ripetizioni	Tempo DAO	Tempo CalendarView	Tempo Cronometro
0	0.57	0.68	4
0	0.11	0.13	3.25
0	0.14	0.15	2.5
0	0.08	0.09	2.5
0	0.07	0.12	2.25
	Avg: 0.194	Avg: 0.234	Avg: 2.9
	Ripetizioni: 542		
Ripetizioni	Tempo DAO	Tempo CalendarView	Tempo Cronometro
542	0.49	0.89	5
542	0.14	0.24	3.5
542	0.12	0.19	3
542	0.09	0.16	2.3
542	0.1	0.13	2.45
	Avg: 0.188	Avg: 0.322	Avg: 3.25
	Ripetizioni: 1082		
Ripetizioni	Tempo DAO	Tempo CalendarView	Tempo Cronometro
1082	0.22	1.13	4
1082	0.11	0.19	3.5
1082	0.11	0.19	3.5
1082	0.09	0.43	3
1082	0.1	0.15	3
	Avg: 0.126	Avg: 0.418	Avg: 3.4
	Ripetizioni: 57602		
Ripetizioni	Tempo DAO	Tempo CalendarView	Tempo Cronometro
57602	0.6	2.2	27.3
57602	0.38	4.31	37
57602	0.29	1.18	29
57602	0.23	0.87	25.3
57602	0.24	0.63	25
	Avg: 0.348	Avg: 1.838	Avg: 28.72

Figura 5.1.5.5.1 - Performance Test Activity Repeat

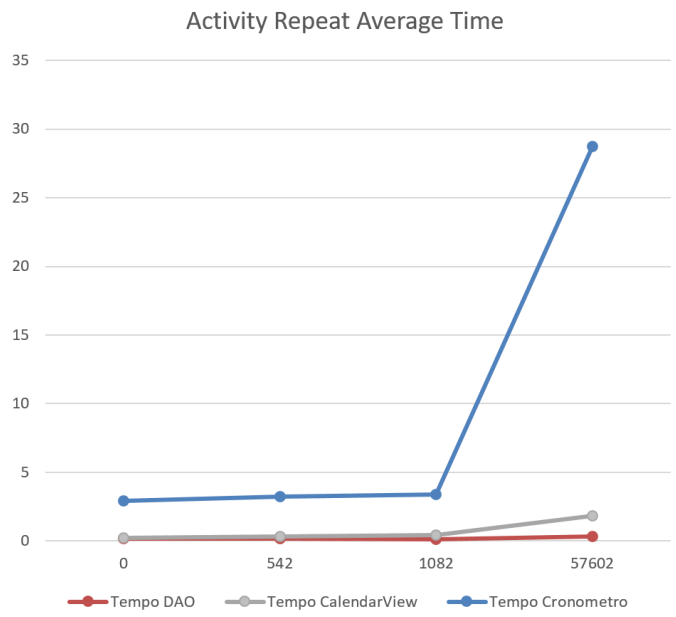


Figura 5.1.5.5.2 - Grafico Activity Repeat - Asse X: numero di ripetizioni - Asse Y: tempo (ms)

5.2. Video Visita

L'applicazione Web consente al Medico Specialista di interagire da remoto con il Paziente durante la visita domiciliare effettuata dall'Operatore.

In Figura 26 è mostrata la schermata di una visita domiciliare visualizzata da un Medico Specialista: la voce "Prenota" permette di prenotare la Video Visita.

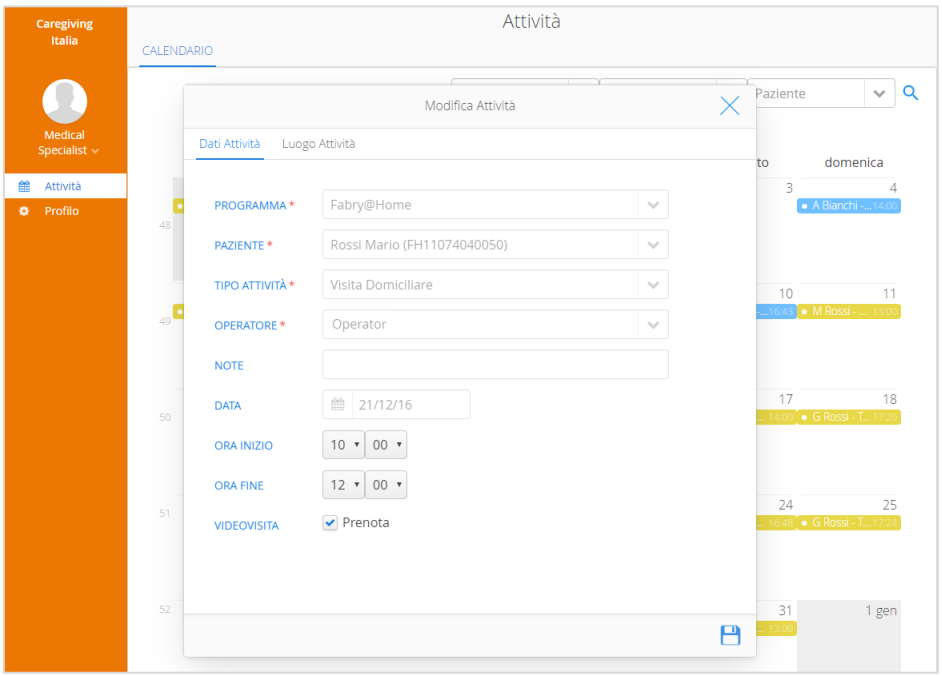


Figura 5.2.1 – Finestra di prenotazione della video visita

Una volta che il Medico Specialista richiede la Video Visita, viene generata una e-mail di notifica con la conferma della prenotazione.


Gentile Dr. Luca Neri,

le confermiamo la prenotazione per il giorno 21/12/2016 alle ore 10:00 della videovisita con il Sig. Mario Rossi.

La preghiamo di cliccare qui per accedere all'applicazione

Se il collegamento non dovesse funzionare, copi il link sottostante e lo incolli nella barra degli indirizzi del suo browser
https://demo.datariverweb.com/caregiving/italia/#!Activity

Cordiali saluti,
Team Caregiving

CAREGIVING

Figura 27 – Notifica video visita

L'Operatore che andrà presso il domicilio del Paziente riceve una notifica simile, che lo informa che il Medico Specialista richiede la Video Visita per il giorno indicato.

L'applicazione Web renderà disponibile all'Operatore la finestra di avvio della video visita con il Medico Specialista solo nel momento in cui è definita l'Attività. Prima o dopo l'Attività non è possibile per l'Operatore accedere alla Video Visita per quella stessa Attività.

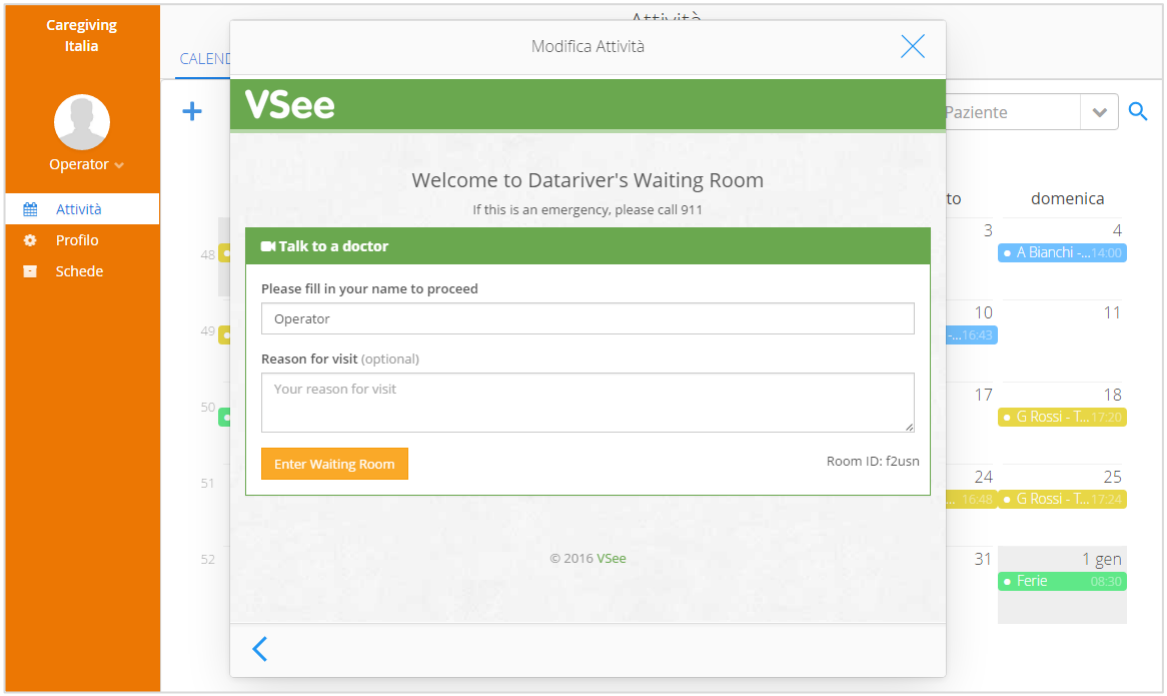


Figura 5.2.2 – Finestra di avvio della video visita

Al click sul bottone per attivare la Video Visita, viene automaticamente avviato il software per effettuare la Video Visita: in questo modo il Medico Specialista riceve la chiamata e può accettarla per monitorare la visita del Paziente effettuata dall'Operatore.

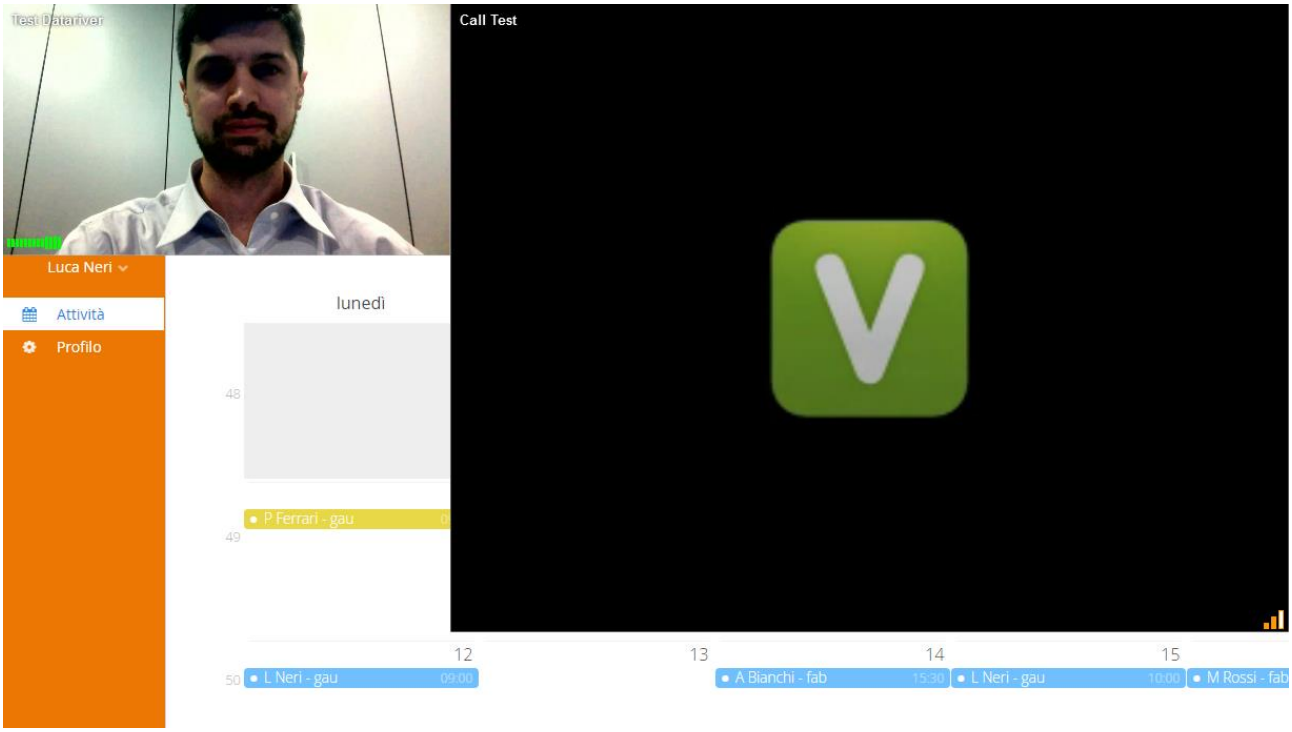


Figura 5.2.3 – Schermata di avvio della video visita

La funzionalità di Video Visita è stata implementata attraverso VSee, una piattaforma software HIPAA compliant (*Health Insurance Portability and Accountability Act*) che rispetta gli standard internazionali per la sicurezza e protezione dei dati sensibili dei Pazienti.

Le informazioni audio e video trasmesse durante la Visita sono criptate end-to-end in modo che le informazioni di salute dei Pazienti siano protette rispettando la privacy dei dati.

L'applicazione VSee è disponibile per dispositivi mobili a partire dalle versioni di iOs 8.0 e Android 4.1 ed è pensata per garantire una stabilità di connessione con reti WIFI, 3G e 4G.

La Video Visita è supportata anche su PC con Windows 7 o versioni successive e Mac a partire dalla versione OS X 10.7, presentando anche la funzionalità di messaggistica.

La funzionalità di Video Visita è stata implementata per fare in modo che il Medico Specialista monitori lo stato di salute del Paziente e allo stesso tempo possa fornire un supporto immediato all'Operatore nel corso dell'Attività assistenziale.

5.3. Anagrafiche

L'applicazione presenta un modulo per la compilazione, la presentazione e la modifica delle Anagrafiche dei Pazienti e degli Utenti, e dei dati sui Centri Clinici e le Farmacie.

5.3.1. Programma

Attraverso la tab Programma è possibile visualizzare, inserire, modificare ed eliminare i Programmi di Assistenza ai Pazienti.
La schermata iniziale della tab Programma presenta:

- Una tabella con l'elenco dei Programmi;
- Il bottone per aggiungere un nuovo Programma;
- Un filtro *responsive* che viene applicato su tutte le colonne della tabella quando l'Utente vi digita.

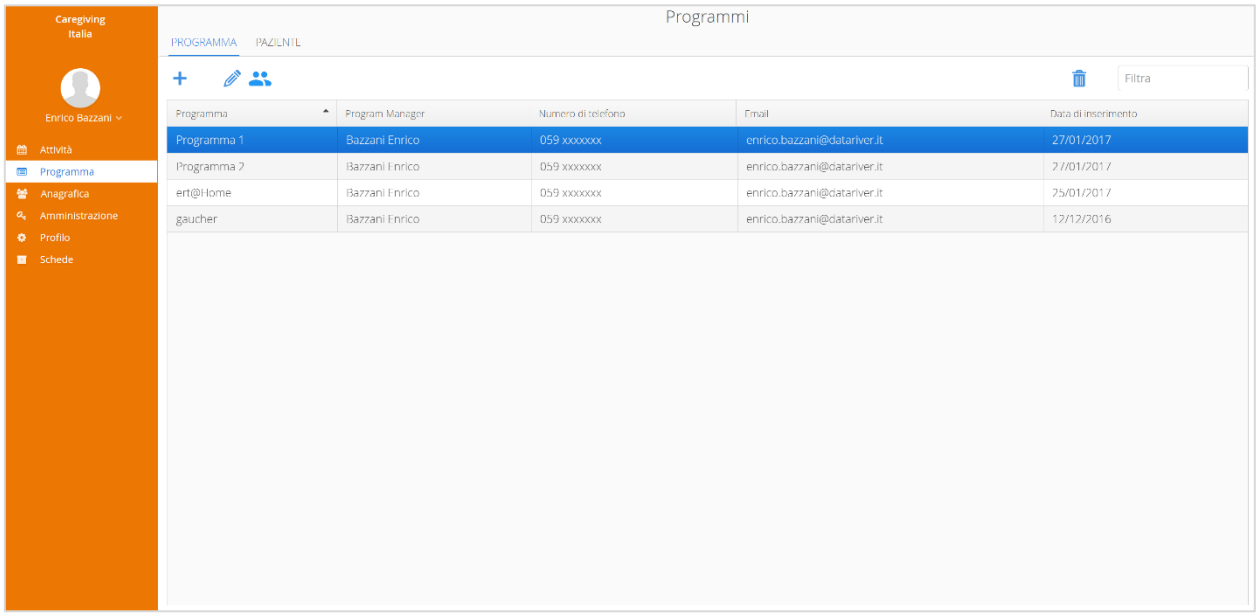


Figura 5.3.1.1 - Elenco dei Programmi

Quando un Programma è selezionato, a fianco del bottone per aggiungere un Programma, compaiono:

- Il bottone per modificarlo;
- Il bottone per associare gli Utenti al Programma;
- Il bottone per cancellarlo.

La creazione e la modifica di un Programma avvengono attraverso la stessa finestra, che in un caso presenta tutti i campi vuoti, mentre nell'altro presenta i campi già precompilati con le informazioni del Programma che si sta modificando.

I campi da compilare o modificare sono due:

- Il nome del Programma;
- Il Program Manager, che viene selezionato tramite una *CheckTable* tra gli Utenti che hanno il Ruolo di *Program Manager*.

Il Program Manager deve essere uno ed uno solo per ogni Programma.
È disponibile un filtro per la ricerca rapida del Program Manager.

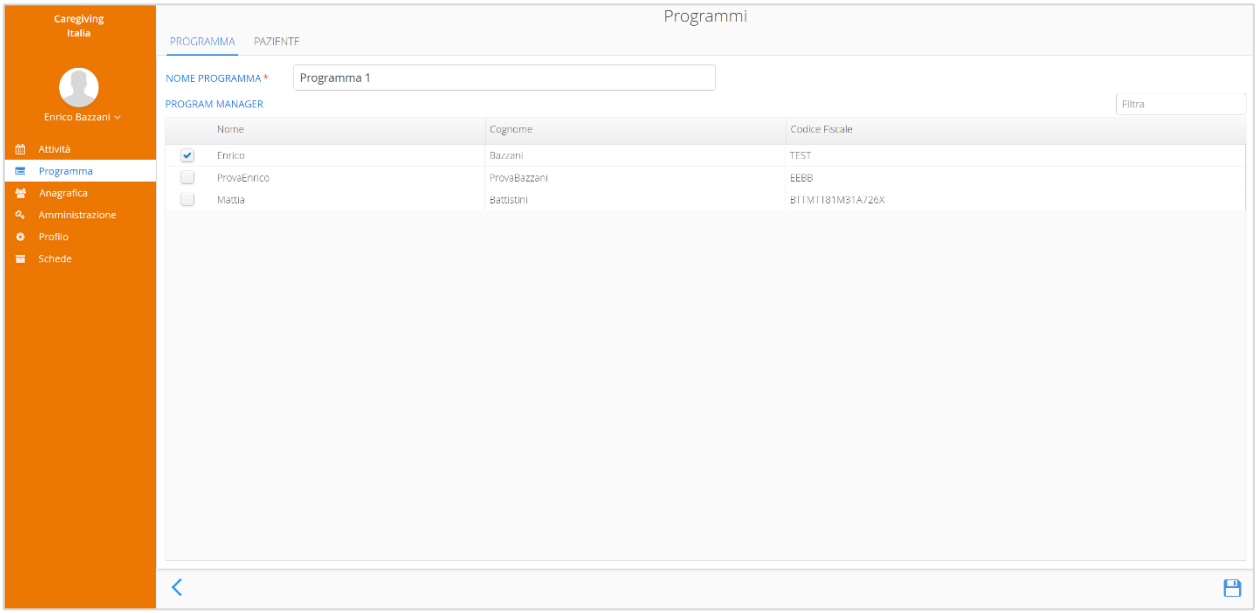


Figura 5.3.1.2 - Modifica di un Programma

L'associazione degli Utenti al Programma è fondamentale per permettere agli Utenti di interagire con i vari Pazienti di quel Programma.
Per ogni Programma è possibile associare un numero indefinito di Utenti, con i Ruoli di:

- Territory Manager
- Program Doctor
- Operatore
- Contact Center
- Medico Specialista
- Pharmacist

L'associazione avviene mediante 6 *CheckTable*, una per ogni Ruolo, disposte su due colonne.

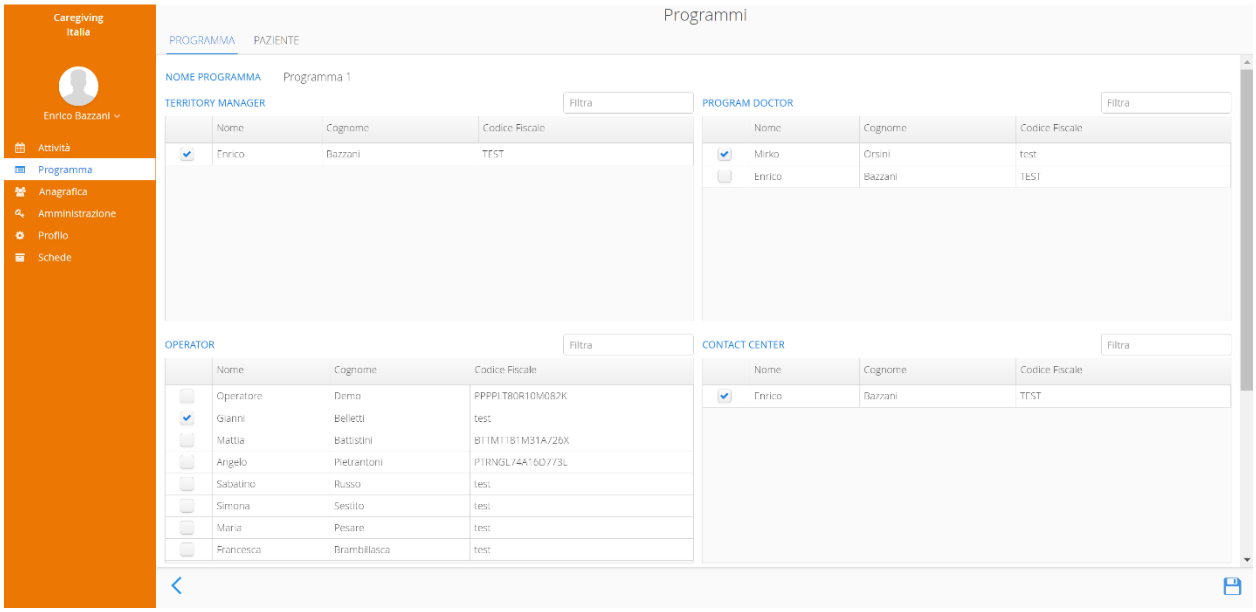


Figura 5.3.1.2 - Associazione di Utenti ad un Programma (sono mostrate solo le prime 4 tabelle)

Una volta salvata l'associazione, gli Utenti associati al Programma potranno scegliere il Programma stesso nelle apposite ComboBox di selezione, come quella presente tra i filtri del Calendario o quella presente nella creazione di una nuova Attività, e potranno di conseguenza selezionare i Pazienti che sono a loro volta associati a quel Programma.

5.3.2. Paziente

Attraverso la tab Paziente è possibile visualizzare, inserire, modificare ed eliminare i Pazienti. La schermata iniziale della tab Paziente presenta:

- Una tabella con l'elenco dei Pazienti;
- Il bottone per aggiungere un nuovo Paziente;
- Un filtro *responsive* che viene applicato su tutte le colonne della tabella quando l'Utente vi digita.

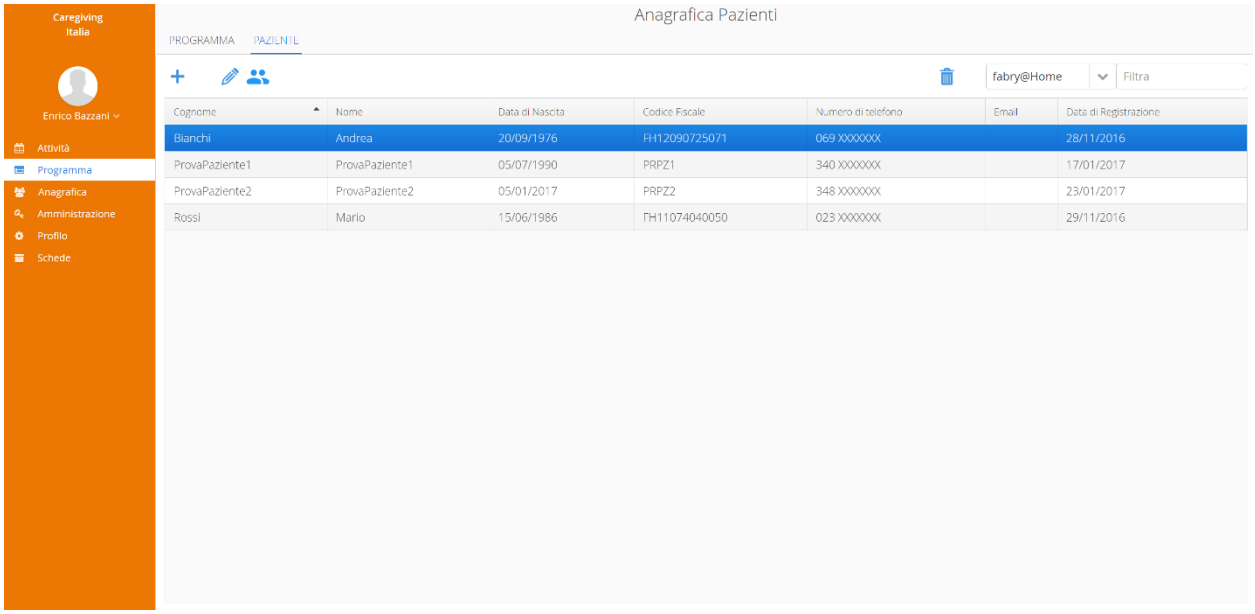


Figura 5.3.2.1 - Elenco dei Pazienti

Quando un Paziente è selezionato, a fianco del bottone per aggiungere un Paziente, compaiono:

- Il bottone per modificarlo;
- Il bottone per associare gli Utenti al Paziente;
- Il bottone per cancellarlo.

La creazione e la modifica di un Paziente avvengono attraverso la stessa finestra, che in un caso presenta tutti i campi vuoti, mentre nell'altro presenta i campi già precompilati con le informazioni del Programma che si sta modificando.

Ogni Paziente è riferito ad un solo Programma. Nel caso –estremamente raro- di un solo Paziente che fa uso di più di un Programma Assistenziale, vengono creati due Pazienti distinti, uno per programma, con gli stessi dati.

I campi da compilare o modificare per il Paziente sono:

- Il Programma di cui fa parte il Paziente;
- Nome;
- Cognome;
- Codice Paziente;
- Data di Nascita;
- Sesso;
- Codice Fiscale;
- Telefono;
- Email;
- Titolo;
- Luogo di Nascita;

- Residenza;
- Domicilio (se diverso dalla Residenza);

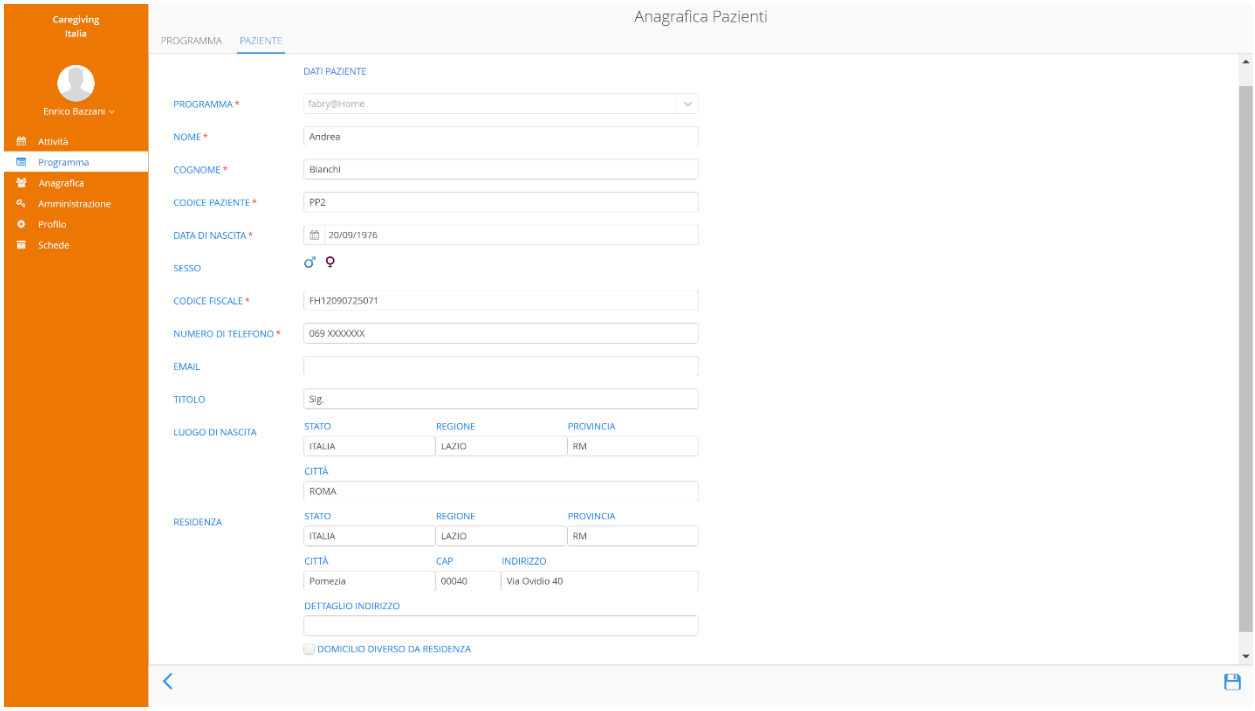


Figura 5.3.2.2 - Modifica di un Paziente

L'associazione degli Utenti al Paziente è fondamentale per permettere agli Utenti di interagire con il Paziente stesso.
Per ogni Programma è possibile associare un numero indefinito di Utenti, con i Ruoli di:

- Territory Manager
- Program Doctor
- Operatore
- Contact Center
- Medico Specialista
- Pharmacist

L'associazione avviene mediante 6 *CheckTable*, una per ogni Ruolo, disposte su due colonne.

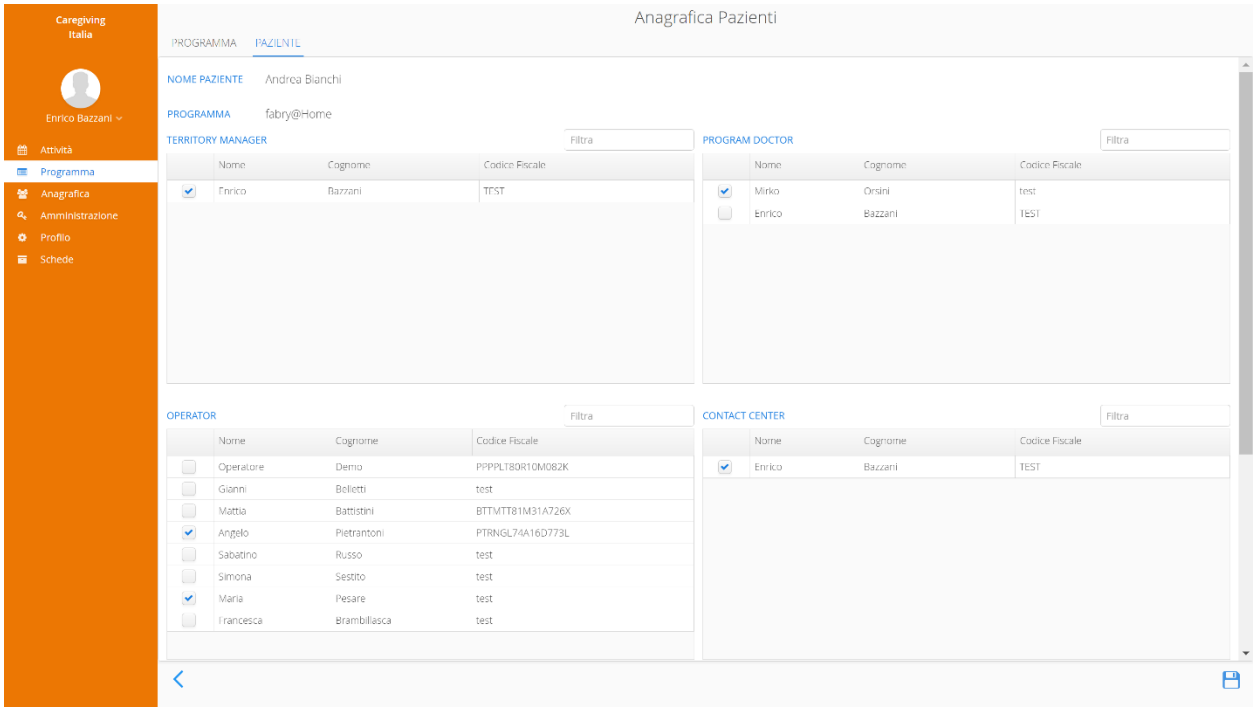


Figura 5.3.2.3 - Associazione di Utenti ad un Paziente (sono mostrate solo le prime 4 tabelle)

Una volta salvata l'associazione, gli Utenti associati al Paziente potranno scegliere il Paziente stesso nelle apposite ComboBox di selezione, come quella presente tra i filtri del Calendario o quella presente nella creazione di una nuova Attività.

5.3.3. Farmacia

Attraverso la tab Farmacia è possibile visualizzare, inserire, modificare ed eliminare le Farmacie. La schermata iniziale della tab Farmacia presenta:

- Una tabella con l'elenco delle Farmacie;
- Il bottone per aggiungere una nuova Farmacia;
- Un filtro *responsive* che viene applicato su tutte le colonne della tabella quando l'Utente vi digita.

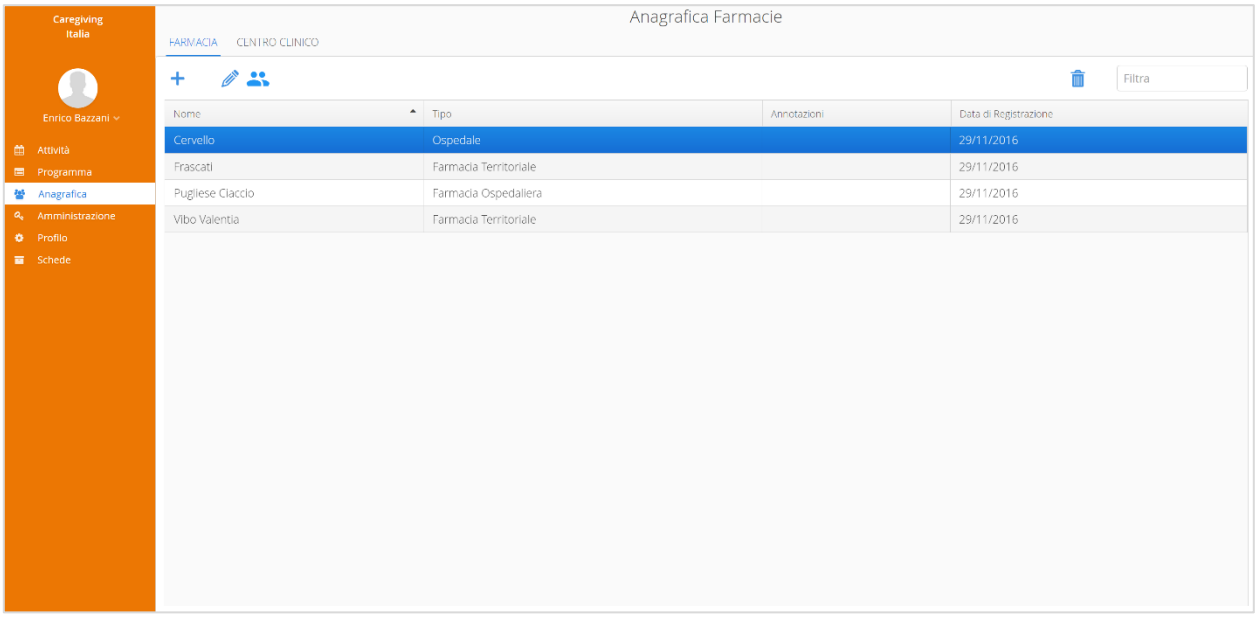


Figura 5.3.3.1 - Elenco delle Farmacie

Quando una Farmacia è selezionata, a fianco del bottone per aggiungere una Farmacia, compaiono:

- Il bottone per modificarla;
- Il bottone per associare gli Utenti con Ruolo Pharmacist alla Farmacia;
- Il bottone per cancellarla.

La creazione e la modifica di una Farmacia avvengono attraverso la stessa finestra, che in un caso presenta tutti i campi vuoti, mentre nell'altro presenta i campi già precompilati con le informazioni della Farmacia che si sta modificando.

I campi da compilare o modificare per la Farmacia sono:

- Nome;
- Tipo;
- Telefono;
- Fax;
- Email;
- Annotazioni;
- Indirizzo.

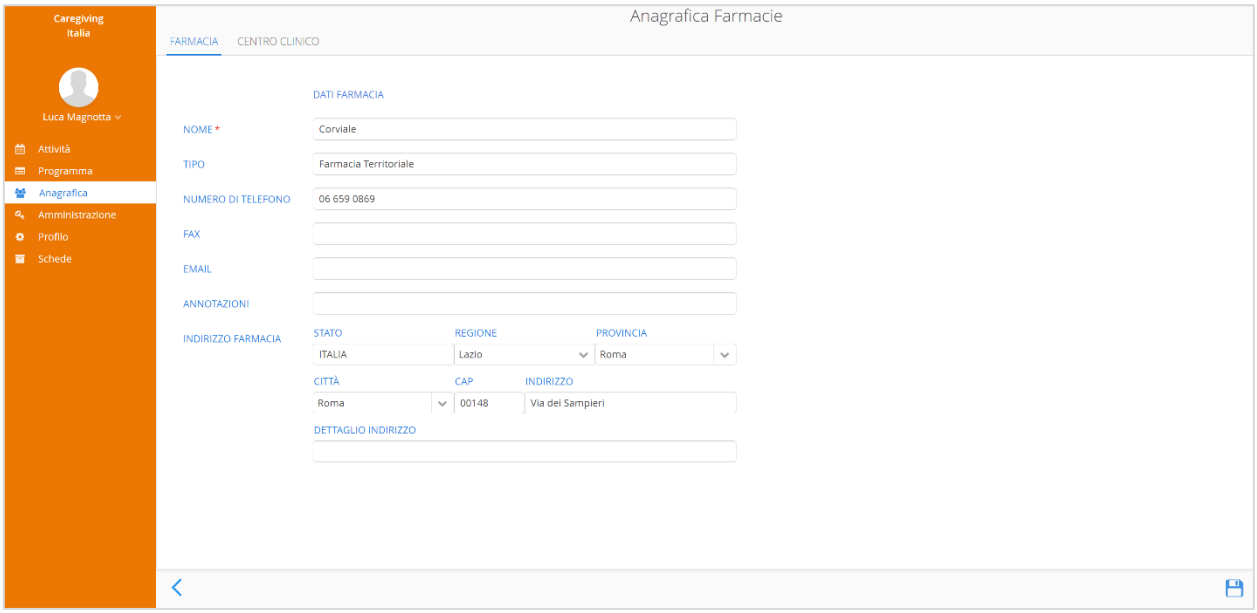


Figura 5.3.3.25 - Modifica di una Farmacia

Per ogni Farmacia è possibile associare un numero indefinito di Utenti con il Ruolo di Pharmacist.

L'associazione avviene mediante una *CheckTable*.

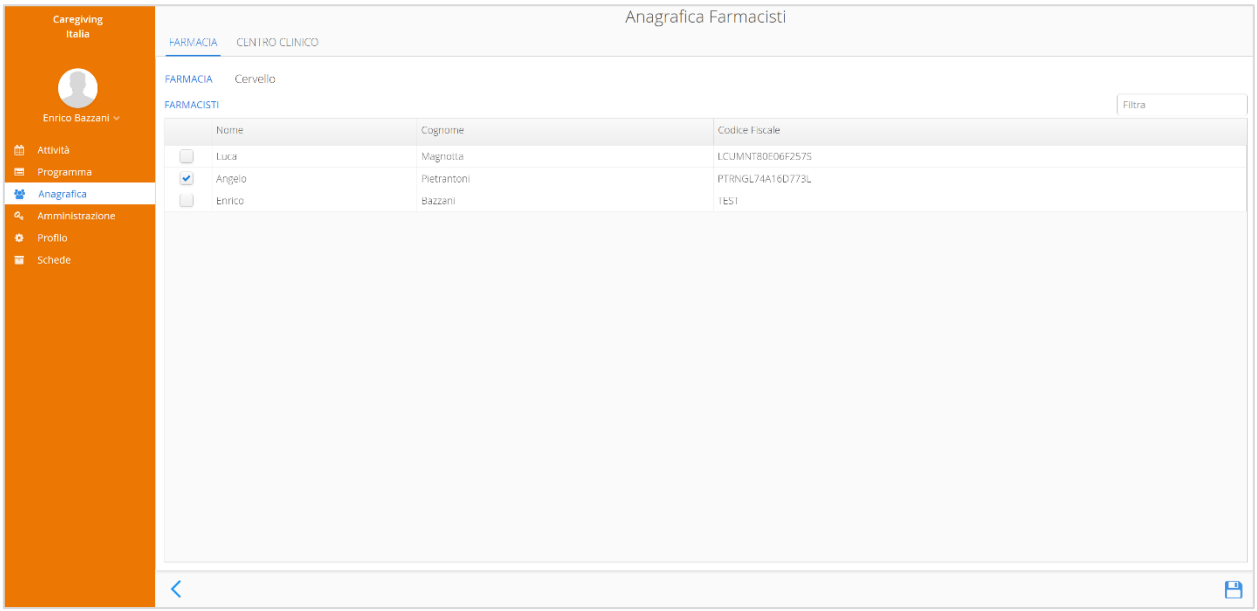


Figura 5.3.3.36 - Associazione di Pharmacists ad una Farmacia

5.3.4. Centro Clinico

Attraverso la tab Centro Clinico è possibile visualizzare, inserire, modificare ed eliminare i Centri Clinici.

La schermata iniziale della tab Centro Clinico presenta:

- Una tabella con l'elenco dei Centri Clinici;

- Il bottone per aggiungere un nuovo Centro Clinico;
- Un filtro *responsive* che viene applicato su tutte le colonne della tabella quando l'Utente vi digita.

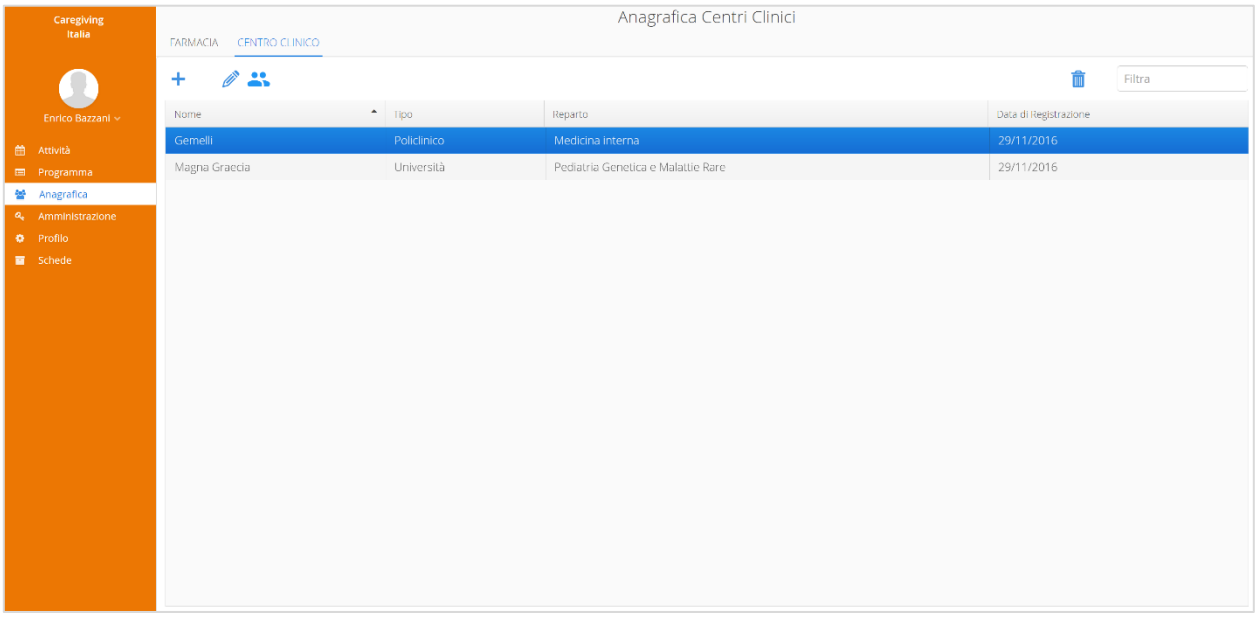


Figura 5.3.4.1 - Elenco dei Centri Clinici

Quando un Centro Clinico è selezionato, a fianco del bottone per aggiungere un Centro Clinico, compaiono:

- Il bottone per modificarlo;
- Il bottone per associare gli Utenti con Ruolo Medico Specialista al Centro Clinico;
- Il bottone per cancellarlo.

La creazione e la modifica di un Centro Clinico avvengono attraverso la stessa finestra, che in un caso presenta tutti i campi vuoti, mentre nell'altro presenta i campi già precompilati con le informazioni della Farmacia che si sta modificando.

I campi da compilare o modificare per la Farmacia sono:

- Nome;
- Tipo;
- Reparto;
- Telefono;
- Fax;
- Email;
- Indirizzo.

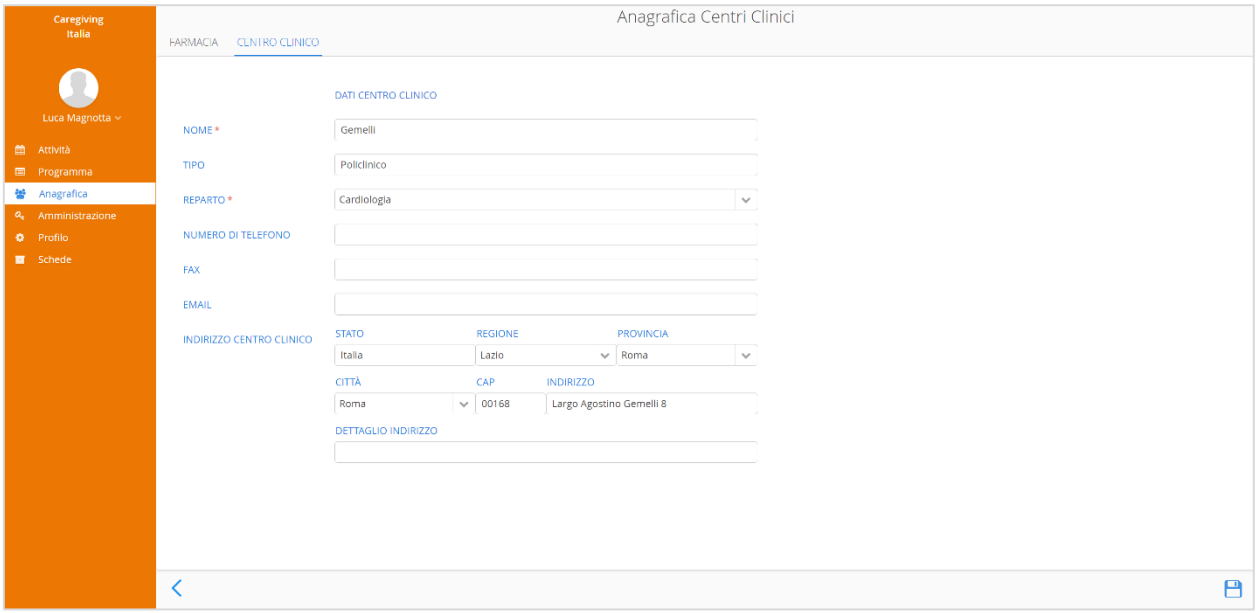


Figura 5.3.4.2 - Modifica di un Centro Clinico

Per ogni Centro Clinico è possibile associare un numero indefinito di Utenti con il Ruolo di Medico Specialista.

L'associazione avviene mediante una *CheckTable*.

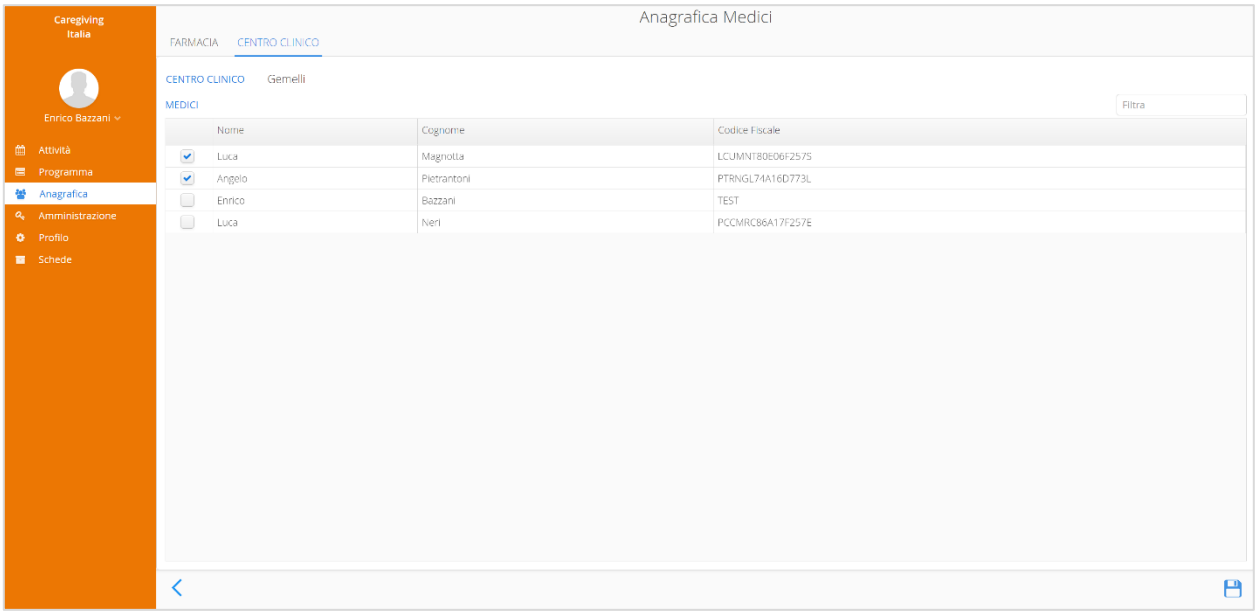


Figura 5.3.4.3 - Associazione di Medici Specialisti ad un Centro Clinico

5.3.5. Layout Indirizzi

Il Layout per inserire gli Indirizzi viene usato nei pannelli del Paziente, dell'Utente, del Centro Clinico e della Farmacia. Per via della sua ubiquità e della relativa complessità, è stato sviluppato

come componente a sè stante, invocabile in due diverse configurazioni dalle classi che lo utilizzano:

- Configurazione ridotta, che mostra:
  - Stato;
  - Regione;
  - Provincia;
  - Città.
- Configurazione estesa, che mostra:
  - Stato;
  - Regione;
  - Provincia;
  - Città;
  - CAP;
  - Indirizzo;
  - Dettaglio Indirizzo.

Regione, Provincia e Città sono selezionabili tramite una ComboBox, mentre Stato, CAP, Indirizzo e Dettaglio Indirizzo sono delle TextField libere.

Un'istanza del Layout Indirizzi definisce un singolo Indirizzo, e più istanze possono essere impiegate per definire uno schema di Indirizzi più complesso, come nel caso dell'Anagrafica Paziente, che presenta:

- Il **Luogo di Nascita**, in configurazione ridotta;
- La **Residenza**, in configurazione estesa;
- Il **Domicilio**, in configurazione estesa, attivabile tramite la CheckBox "Domicilio diverso da Residenza".

Figura 5.3.5.1 - Layout Indirizzi con Domicilio uguale a Residenza

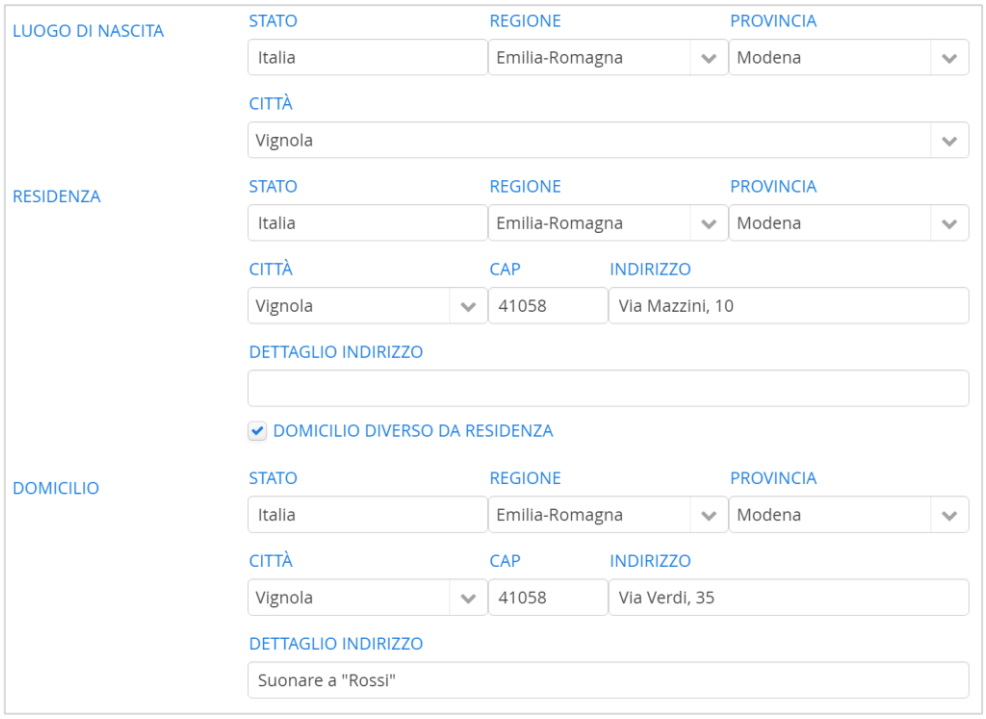


Figura 5.3.5.2 - Layout Indirizzi con Domicilio diverso da Residenza

5.3.5.1. Architettura degli Indirizzi

Gli Indirizzi sono salvati nella tabella Address del Database:

Address	
NOME COLONNA	TIPO DI DATO
id_address	Serial
country	Varchar
region	Varchar
province	Varchar
city	Varchar
zip_code	Varchar
address_name	Text
address_detail	Text

Per ogni Indirizzo, i campi immessi dall'Utente sono salvati come testo. La scelta di Regione, Provincia e Città da ComboBox ha come unico scopo quello di assistere l'Utente nell'immissione e contemporaneamente evitare il salvataggio di Indirizzi malformati.

Le comboBox di Regione, Provincia e Città vengono alimentate dalle tre tabelle AddressRegion, AddressProvince, AddressCity:

Address_Region

NOME COLONNA	TIPO DI DATO
id_address_region	Serial
name	Text
latitude	Double
longitude	Double

Address_Province	
NOME COLONNA	TIPO DI DATO
id_address_province	Serial
name	Text
cod_region	Integer
cod_metropolitan_city	Integer
province_acronym	Varchar
latitude	Double
longitude	Double

Address_City	
NOME COLONNA	TIPO DI DATO
id_address_city	Serial
name	Text
cod_region	Integer
cod_province	Integer
is_province_capital	Integer
cadastral_code	Text
latitude	Double
longitude	Double

In AddressCity, i campi **cod_region** e **cod_province** fanno riferimento ad AddressRegion e AddressProvince.

In AddressProvince, i campi **cod_region** e **cod_metropolitan_city** fanno riferimento ad AddressRegion e AddressCity.

5.3.5.2. Logica degli Indirizzi

Le tre ComboBox Regione, Provincia e Città sono strutturate secondo una logica gerarchica.

Quando il Layout degli Indirizzi viene caricato, solamente la ComboBox della Regione viene popolata, attraverso una query che seleziona tutte le Regioni dalla tabella AddressRegion.

Nel momento in cui un Utente seleziona effettivamente una delle Regioni caricate, il Listener della ComboBox Regione provvede a chiamare una query che popola la ComboBox delle Province passandogli in input l'id della Regione. In questo modo vengono selezionate tutte le Province di quella Regione.

Nel momento in cui un Utente seleziona effettivamente una delle Province caricate, il Listener della ComboBox Provincia provvede a chiamare una query che popola la ComboBox delle Città passandogli in input l'id della Regione e quello della Provincia. In questo modo vengono selezionate tutte le Città di quella Provincia.

Se un Utente cambia il valore della ComboBox Regione dopo aver selezionato il valore della ComboBox Provincia o della stessa ComboBox Provincia dopo aver selezionato il valore della ComboBox Città, la selezione sulla ComboBox di livello inferiore viene annullata.

5.3.6. Layout Numeri di Telefono

Analogamente a quanto fatto per il Layout Indirizzi, anche il Layout dei Numeri di Telefono è stato strutturato come classe a sè stante, e viene utilizzato dai pannelli del Paziente e dell'Utente.

A differenza del Layout Indirizzi, il Layout Numeri di Telefono deve essere istanziato una sola volta, perchè implementa una logica che permette di aggiungere un numero indefinito di Numeri di Telefono.

Il singolo Numero di Telefono viene immesso tramite una TextField, ed è possibile aggiungere nuovi Numeri, cliccando sull'apposita icona di aggiunta, solo se tutti i campi precedenti sono stati riempiti.

È possibile inoltre eliminare i Numeri presenti, cliccando sull'icona di eliminazione a fianco di ogni riga.

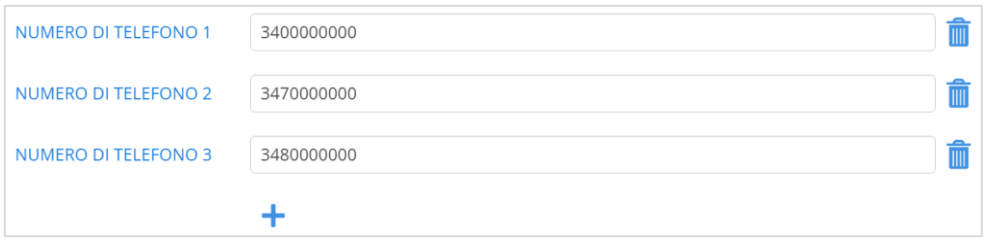


Figura 5.3.6.1 - Layout dei Numeri di Telefono

5.3.6.1. Architettura dei Numeri di Telefono

I Numeri di Telefono sono salvati nella tabella Address del Database:

Phone_Number	
NOME COLONNA	TIPO DI DATO
id_phone_number	Serial

number	Varchar
type	Varchar

Dove la colonna **type** descrive il tipo di Numero di Telefono, ma non è attualmente utilizzata dall'Applicazione.

Siccome i Numeri di Telefono devono essere associati sia agli Utenti che ai Pazienti, si è scelto di utilizzare due tabelle di mapping per realizzare l'associazione:

User_Account_Phone_Number	
NOME COLONNA	TIPO DI DATO
id_user_account_phone_number	Serial
id_user_account	Integer
id_phone_number	Integer

Patient_Phone_Number	
NOME COLONNA	TIPO DI DATO
id_patient_phone_number	Serial
id_patient	Integer
id_phone_number	Integer

In entrambe le tabelle di mapping, il campo **id_phone_number** fa riferimento al record di PhoneNumber che contiene il numero associato all'Utente o al Paziente.

Al momento di un inserimento, modifica o eliminazione, viene aggiornato il record nella tabella PhoneNumber e successivamente viene aggiornato il record nella relativa tabella di mapping.

5.4. Scheda Visita

All'interno dell'applicazione Web è stato progettato il modulo software che gestisce le Schede compilate dagli Operatori durante la Visita.

La compilazione delle Schede tramite dispositivo mobile sostituisce la compilazione delle Schede in versione cartacea, pertanto il layout dei menù di compilazione della Scheda Visita ne deve ricordare la formattazione.

I componenti grafici implementati per nelle Schede sono organizzati in una gerarchia dove il livello superiore è costituito da un FormSummary o da un FormDetail:

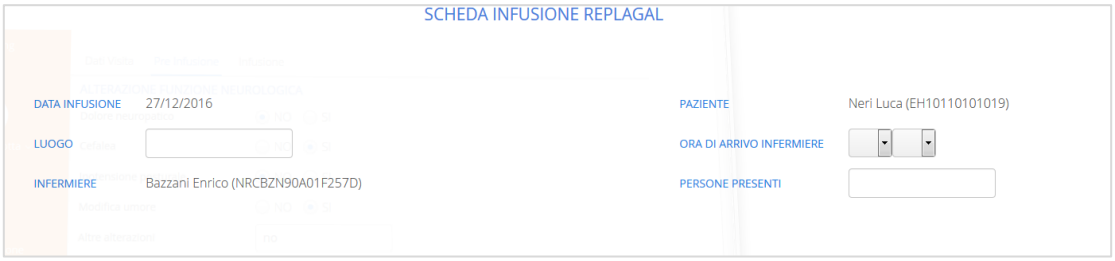


Figura 5.4.1 – Componente FormSummary

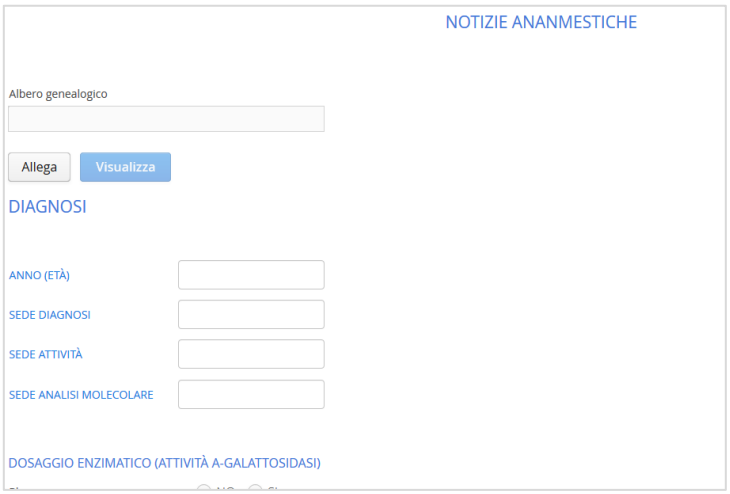


Figura 5.4.2 – Componente FormDetail

Entro questi layout di livello superiore, vengono organizzati gli elementi definiti nel Database:

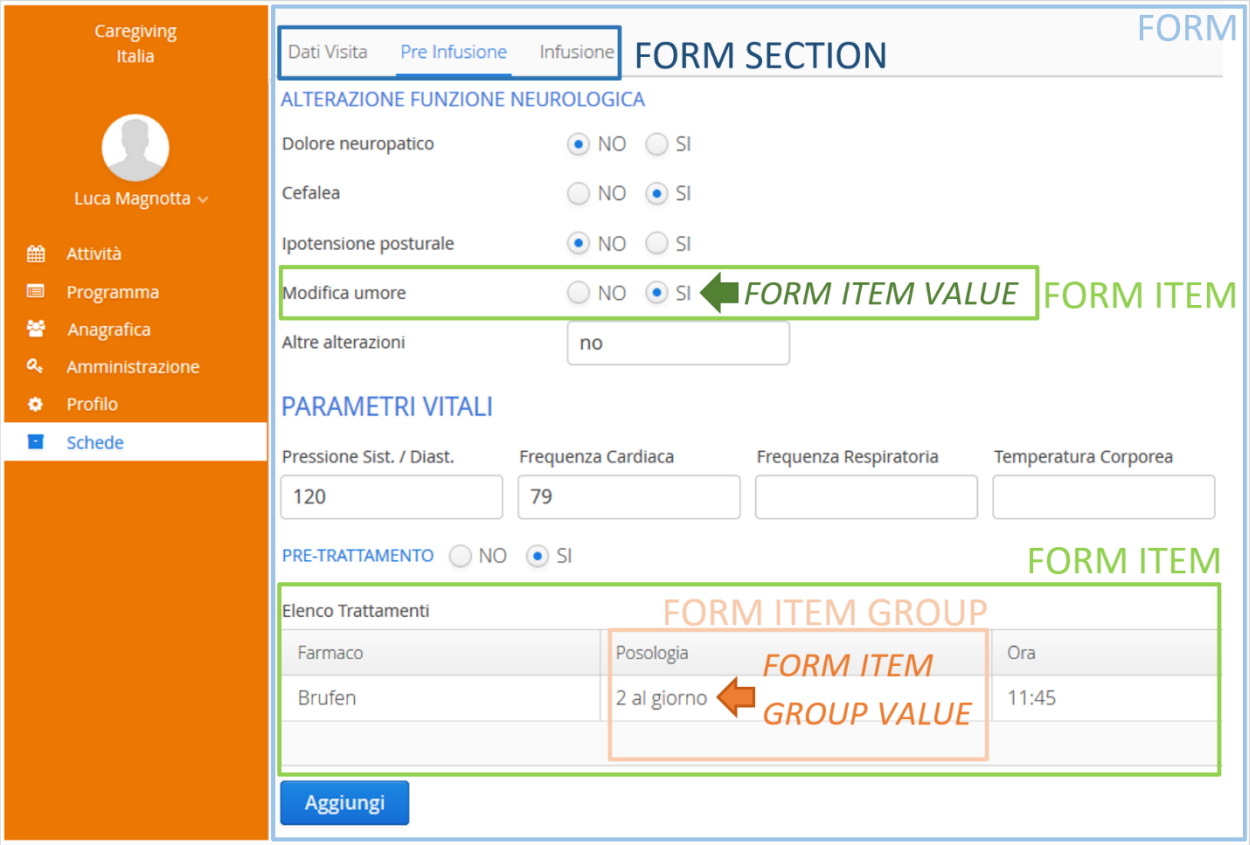


Figura 5.4.3 – Layout di esempio di una Scheda

5.4.1. Architettura della Scheda Visita

Il Database contiene le informazioni riguardanti la struttura dei componenti delle Schede nella tabella FormItem e i valori di ciascuno di essi in FormItemValue. I componenti che mostrano più di un valore (come le Table) sono invece contenuti nella tabella FormItemGroup, ed il loro valore in FormItemGroupValue.

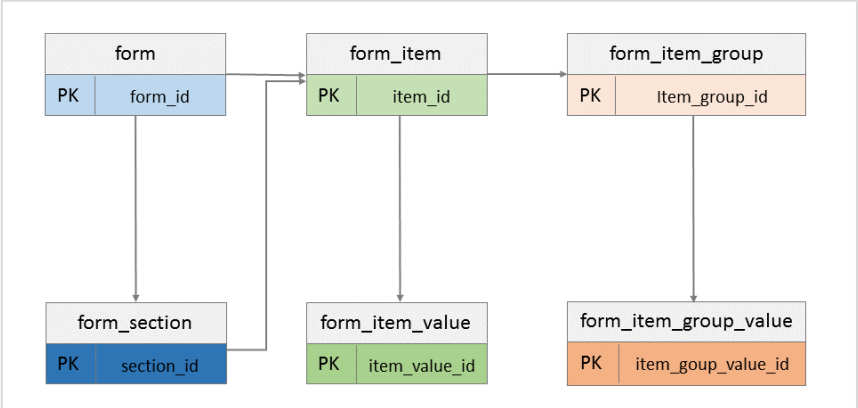


Figura 5.4.1.1 - Schema E/R delle tabelle del Database relative alle Schede

I dati relativi alle Schede sono contenuti nelle seguenti tabelle:

Form	
NOME COLONNA	TIPO DI DATO
form_id	Serial
form_code	Varchar
form_version	Varchar
form_type	Integer
description	Varchar
owner_id	Integer
update_id	Integer
date_created	Timestamp with time zone
date_updated	Timestamp with time zone
deleted	Boolean

La tabella Form contiene le informazioni generali della Scheda ovvero: data di creazione, autore, versione e descrizione.

Form_Section	
NOME COLONNA	TIPO DI DATO
form_section_id	Serial
form_id	Integer
section_code	Varchar
form_version	Varchar
ordinal	Integer
description	Varchar
owner_id	Integer
update_id	Integer
date_created	Timestamp with time zone
date_updated	Timestamp with time zone
deleted	Boolean

La tabella FormSection che definisce le sezioni in cui viene divisa la Scheda:

- form_id punta alla Form che contiene la FormSection

Form_Item	
NOME COLONNA	TIPO DI DATO
form_item_id	Serial
parent_item_id	Integer

section_id	Integer
form_id	Integer
ordinal_in_parent	Integer
item_code	Varchar
item_code_suffix	Varchar
item_type	Varchar
description	Varchar
owner_id	Integer
update_id	Integer
date_created	Timestamp with time zone
date_updated	Timestamp with time zone
is_group	Boolean
deleted	Boolean

La tabella FormItem definisce i contenuti all'interno di una Sezione di una Scheda:

- form_id punta alla Form che contiene il FormItem;
- section_id punta alla Sezione, nella Form, che contiene il FormItem;
- item_type è una stringa che definisce il tipo di componente rappresentato dal FormItem, indicando il percorso comlpeto alla classe di quel componente;
- parent_item_id è un riferimento ad un FormItem di livello superiore, che contiene il FormItem corrente;
- ordinal_in_parent è un progressivo che definisce l'ordine del FormItem dentro il FormItem di livello superiore.

Form_Item_Value	
NOME COLONNA	TIPO DI DATO
form_item_value_id	Serial
Item_id	Integer
activity_id	Integer
form_id	Integer
item_value_code	Varchar
description	Varchar
value	Text
owner_id	Integer
update_id	Integer
date_created	Timestamp with time zone
date_updated	Timestamp with time zone

deleted	Boolean

La tabella FormItemValue contiene i valori dei FormItems:

- **form_id** punta alla Form che contiene il FormItem;
- **item_id** punta al FormItem;
- **value** contiene il valore del FormItem, formattato come stringa.

Form_Item_Group	
NOME COLONNA	**TIPO DI DATO**
form_item_group_id	Serial
item_id	Integer
item_group_code	Varchar
data_type	Varchar
col_order	Integer
description	Varchar
owner_id	Integer
update_id	Integer
date_created	Timestamp with time zone
date_updated	Timestamp with time zone
deleted	Boolean

La tabella **FormItemGroup** definisce estende FormItem, rappresentando elementi che contengono più valori:

- **item_id** punta al FormItem che viene esteso con più valori;
- **description** contiene il nome dell'elemento multiplo: nel caso di una Table, ad esempio, conterrà il nome della colonna.

Form_Item_Group_Value	
NOME COLONNA	**TIPO DI DATO**
form_item_group_value_id	Serial
item_group_id	Integer
activity_id	Integer
form_id	Integer
item_group_value_code	Varchar
value	Varchar

col_order	Integer
row_order	Integer
description	Varchar
owner_id	Integer
update_id	Integer
date_created	Timestamp with time zone
date_updated	Timestamp with time zone
deleted	Boolean

La tabella FormItemGroupValue contiene i valori multipli dei FormItem estesi da FormItemGroup:

- form_id punta alla Form che contiene il FormItemGroup;
- item_group_id punta al FormItemGroup;
- value contiene un valore del FormItemGroup, formattato come stringa.

Ad esempio, se una Table contenesse 4 colonne, ci sarebbero 4 record in FormItemGroup, dove ciascuno rappresenta una colonna diversa.

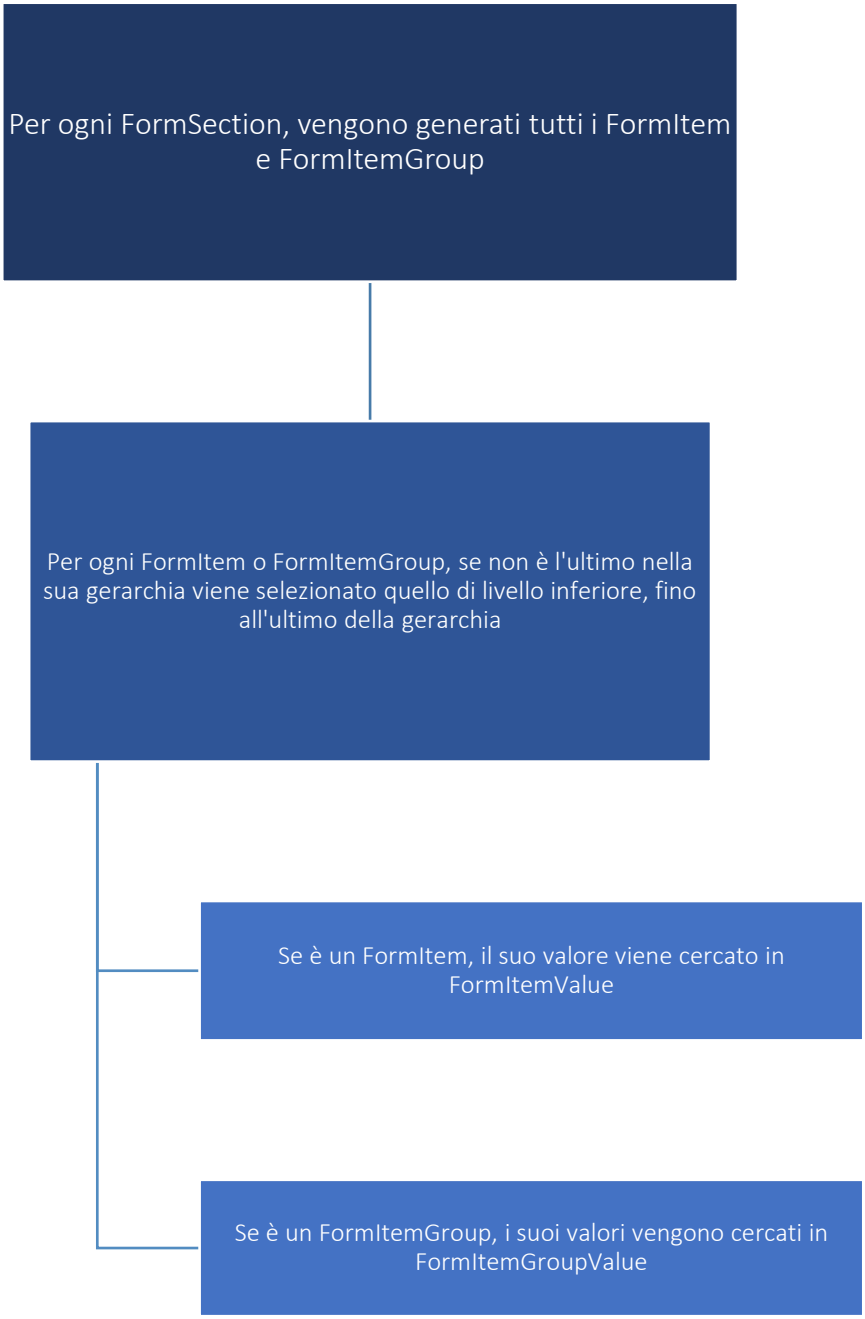
Se poi la Table avesse una riga compilata, ci sarebbero 4 record in FormItemGroupValue, ciascuno con un proprio valore e riferito ad uno specifico FormItemGroup, ovvero ad una specifica colonna.

La struttura verticale delle tabelle del Database relative alle Schede permette l'aggiunta dinamica di nuove Schede e nuovi componenti a ciascuna Scheda, senza dover prevedere un aggiornamento del codice dell'Applicazione.

5.4.2. Logica della Scheda Visita

Per realizzare le Schede a partire dalle informazioni contenute nel Database viene realizzata inizialmente una Query, che restituisce i vari Form con i relativi Form Item, FormItemValue, FormItemGroup e FormItemGroupValue e FormSection.

Ogni FormSection viene renderizzata come una Tab nell'interfaccia della Scheda.


Per ogni FormSection, vengono generati tutti i FormItem e FormItemGroup
Per ogni FormItem o FormItemGroup, se non è l'ultimo nella sua gerarchia viene selezionato quello di livello inferiore, fino all'ultimo della gerarchia
Se è un FormItem, il suo valore viene cercato in FormItemValue
Se è un FormItemGroup, i suoi valori vengono cercati in FormItemGroupValue

6. Conclusioni

Le stime dei tempi da dedicare alle singole attività, per quanto non sempre precise a livello di compiti specifici, sono state nel complesso rispettate puntualmente. Questo mi ha permesso di terminare il mio lavoro a Marzo implementando tutte le funzionalità che mi erano state richieste in fase di definizione delle specifiche più le modifiche che si sono rivelate necessarie in seguito.

Al momento della redazione della tesi, è in corso l'implementazione delle poche funzionalità mancanti, che non durerà più di un mese. In seguito si procederà ad una fase di testing e validazione complessiva dell'Applicazione, coninvolgendo anche il cliente e gli utilizzatori finali. Si prevede dunque che il lavoro sarà completato definitivamente per il mese di Giugno.

A livello personale, l'esperienza presso DataRiver S.R.L mi ha formato tecnicamente sugli aspetti fondanti della progettazione e dell'implementazione di un software gestionale, in particolare:

- Ho bilanciato in modo ottimale la complessità dell'architettura dei singoli componenti e del Database per ottenere una alta modularità e manutenibilità del codice senza complicare eccessivamente il software;
- Ho migliorato le mie capacità nella progettazione e sviluppo di software attraverso la programmazione ad oggetti, ed in particolare ho sviluppato la capacità di scrivere codice più lineare e leggibile, in quanto dovrà essere certamente rivisto e modificato in futuro da altri sviluppatori;
- Ho imparato ad utilizzare i framework Vaadin ed Hibernate, implementando anche complessi componenti personalizzati, come una versione avanzata del Calendario o i layout dinamici di Indirizzi e Numeri di Telefono;
- Ho infine acquisito la capacità di valutare a priori l'impatto di un'attività e stimarne di conseguenza i tempi. Nei casi in cui le stime non sono state rispettate ho inoltre dovuto identificare le modalità migliori per rientrare nei tempi previsti, introducendo eventualmente modifiche alle attività successive.

Bibliografia

- Relazione sulla Milestone 1
- Specifica dei requisiti della Milestone 1
- truevault.com/blog/how-do-i-become-hipaa-compliant.html
- it.wikipedia.org/wiki/Metodologia_agile
- it.wikipedia.org/wiki/DevOps
- theagileadmin.com/what-is-devops
- vaadin.com/book/vaadin6/-/page/intro.html
- hibernate.org/orm
- salute.gov.it/portale/temi/p2_6.jsp?lingua=italiano&id=4302&area=statisticheSSN&menu=definizioni
- stackoverflow.com/questions/4155783/differences-between-gwt-and-vaadin
- github.com/napolux/italia/blob/master/mysql/italia.sql
- stackoverflow.com/questions/5183630/calendar-recurring-repeating-events-best-storage-method