Università Politecnica delle Marche

FACOLTÀ DI INGEGNERIA
Corso di Laurea Magistrale in Ingegneria Informatica e dell'Automazione

Progettazione e prototipizzazione di una piattaforma di telemedicina distribuita

Design and prototyping of a distributed telemedicine platform

Candidato:

Cristian Di Silvestre

Relatore:

Prof. Alessandro Cucchiarelli

Anno Accademico 2022-2023

UNIVERSITÁ POLITECNICA delle MARCHE
FACOLTÁ di INGEGNERIA
Corso di Laurea Magistrale in Ingegneria Informatica e dell'Automazione LM-32
Via Brecce Bianche – 60131 Ancona (AN), Italia

Quando guardate, guardate lontano,
e anche quando credete di star guardando lontano,
guardate ancor più lontano!

Lord Robert Stephenson Smyth Baden-Powell di Gilwell

Indice

Elenco delle figure

Elenco delle tabelle

Abstract

La creazione di una piattaforma distribuita che gestisca le attività di telemedicina regionali o nazionali nella loro interezza è un cardine principe dell'innovazione della sanità nazionale in Italia. Gli obiettivi del lavoro illustrato in questa relazione sono la progettazione e l'implementazione di un prototipo di sistema di telemedicina che sia conforme agli standard di interoperabilità definiti a livello europeo.

Per realizzare quanto sopra, ci si è basati sulla tecnologia blockchain e sullo standard FHIR per la condivisione e trasmissione di dati sanitari, oltre che sulla creazione di applicazioni crossplatform necessarie all'interfacciamento con il sistema.

I risultati dell'implementazione, se pur limitati a un prototipo, permettono di confermare la possibilità di utilizzare in maniera efficiente ed efficace le tecnologie appena citate per il raggiungimento dell'obiettivo preposto per tale progetto di tesi.

Sviluppi futuri potranno essere volti a un'implementazione sempre più dettagliata e approfondita del modello di piattaforma proposto, al fine di renderlo più funzionale a un contesto reale.

Capitolo 1

Introduzione

Negli ultimi anni, la pandemia da Covid-19 ha costretto la società e l'intero sistema nazionale a ripensare le relazioni sociali, lavorative, educative e non solo. Le restrizioni imposte per arginare la diffusione del contagio da Sars-CoV-2 hanno indotto a valutare una digitalizzazione, in parte o totale, di molte realtà. Il sistema sanitario non è rimasto esente da questa rivoluzione: la necessità di non sovraccaricare le strutture ospedaliere e dunque di ricorrere a strumenti alternativi di monitoraggio, trattamento e prescrizione hanno dato una forte spinta all'avvento della telemedicina.

Di telemedicina, cioè la capacità di erogare servizi sanitari a distanza attraverso dispositivi digitali e reti di comunicazione, si discute già da prima dell'esplosione della pandemia che ha afflitto il mondo intero negli ultimi anni ma, grazie alle motivazioni appena elencate, questa soluzione, di recente, è entrata a pieno titolo nel sistema sanitario nazionale.

Un ulteriore spinta proviene dai fondi del Piano Nazionale di Ripresa e Resilienza. La sesta area di intervento prevista dal PNRR, infatti, è definita come Missione Salute; ad essa vengono destinati € 15,63 miliardi al fine di sostenere importanti riforme e investimenti a beneficio del Servizio sanitario nazionale da realizzare entro il 2026. Tra esse vi è lo sviluppo della telemedicina per cui è stato stanziato un miliardo di euro. Il Ministero della Salute ha quindi predisposto delle linee guida che sono state approvate dalla Conferenza Stato-Regioni. [1]

In Italia esistono già delle delle realtà in cui si fa ricorso a servizi di telemedicina, ma si tratta di situazioni sporadiche che non riguardano l'intero bacino di utenza o tutte le applicazioni possibili in tale ambito.

Risulta necessaria, quindi, una piattaforma di telemedicina che implementi e re-

goli tale servizio in maniera omogenea e distribuita capillarmente in tutto il territorio nazionale.

La telemedicina, però, non può rappresentare una soluzione tecnologica autonoma. Per consentire la continuità del percorso di cura del paziente, essa deve integrarsi nel contesto clinico, organizzativo e tecnologico adottato dallo specifico centro per la specifica patologia e setting assistenziale. [2]

Si ritengono necessari degli standard che possano definire e normare tali interazioni. Con la costituzione del Fascicolo Sanitario Elettronico 2.0 e attraverso l'adozione dello standard di interoperabilità HL7-FHIR si è trovata una soluzione a tale necessità.

Enunciato il contesto applicativo, l'obiettivo è quello di progettare e implementare un prototipo di un sistema distribuito di telemedicina che sia conforme a tali standard oltre che rispettare i requisiti necessari al fine di garantire un corretto funzionamento dello stesso.

Gli assistiti devono rilasciare il consenso al trattamento dei propri dati in ambito telemedico, quindi nell'applicazione che si intende implementare è necessario gestire questa situazione.

Oltre a ciò, occorre definire una modalità di indicizzazione degli eventi telemedicina creando un registro accessibile a tutti gli attori coinvolti nel sistema, gestendo al contempo i dati sanitari attraverso un processo di autenticazione e autorizzazione che possa definire e monitorare i vari livelli di accesso ai dati.

Partendo dagli obiettivi appena definiti si giunge alla conclusione che l'adozione di una rete blockchain consortium, per gestire il registro di indicizzazione e il meccanismo di autorizzazione attraverso il rilascio di token, sia la soluzione che meglio riesce a far fronte alle specifiche definite durante la fase di analisi dei requisiti. Essa prevede la presenza di un'autorità centrale che permette o meno l'accesso delle varie strutture sanitarie alla rete, assegnando ad esse un nodo blockchain a cui viene associato un address univoco e identificativo della struttura stessa.

Per garantire la privatezza dei dati salvati all'interno del registro, dato che quest'ultimo è accessibile a tutti i nodi, gli eventi saranno salvati tenendo traccia dei seguenti campi:

- identificativo interno della struttura sanitaria del paziente interessato dall'evento;
- indirizzo del nodo della struttura (per mantenere l'univocità del campo appena

citato);

- tipo di evento;
- data di esecuzione dello stesso.

In questo modo solo chi conosce l'associazione tra codice identificativo e identità del paziente è in grado di leggere correttamente il contenuto del registro; questa informazione verrà fornita offchain a chi detiene un privilegio di lettura del dato.

La comunicazione dei dati deve essere regolata dallo standard FHIR, per questo le informazioni sanitarie non possono essere salvate nella blockchain; all'interno del registro saranno solo indicizzati gli eventi erogati dalle varie strutture facenti parte del sistema. Ognuna di esse esporrà un endpoint facente capo ad un server FHIR a cui inviare richieste di ottenimento delle risorse.

Al fine di adattare il più possibile il sistema alle esigenze delle aziende sanitarie, i server baseranno il salvataggio permanete e affidabile delle risorse su una base di dati di tipo SQL (tipologia di database maggiormente utilizzata nel contesto della sanità).

Fondamentale importanza viene assunta dalla capacità del sistema di favorire l'accesso di una nuova struttura nella rete; tutte le informazioni dovranno quindi essere salvate nel registro in modo che non sarà necessario notificarle singolarmente ad ogni azienda sanitaria ogni qual volta si verifichi una nuova adesione.

Tutte le componenti descritte vanno a costituire la parte fondante del backend; mentre, lato frontend, verranno implementate diverse possibilità di interfacciamento con il sistema:

- applicativo desktop simulato da un'app di interfaccia da riga di comando;
- app mobile dedicata al livello di utenza dei professionisti sanitari;
- app mobile dedicata agli assistiti.

L'applicazione desktop funge da raccordo tra tutte le componenti del backend offrendo all'utente un livello di astrazione sull'intero sistema. Lo user potrà semplicemente svolgere le operazioni seguendo il workflow dell'interfaccia, sarà l'applicativo ad invocare i servizi necessari all'espletamento delle funzionalità richieste.

L'applicativo mobile renderà il lavoro di telemonitoraggio dei medici più agevole, offrendo le seguenti possibilità:

- visualizzare i dati dei propri assistiti da un qualunque dispositivo con connessione internet;

- ricevere notifiche nel caso in cui si registri un'osservazione anomala per un paziente seguito.

I pazienti avranno la possibilità di associare all'app i sensori utilizzati nel percorso di monitoraggio, oltre che tutte le altre applicazioni mobile conformi agli standard di Google fit (servizio utilizzato per il reperimento dei dati) in cui registrano dati sanitari, ad esempio l'applicativo "Salute" proprietario di iOS.

Le app verranno sviluppate con Ionic, basandosi sul framework Angular, ciò permette di creare una sola istanza dell'applicativo che, attraverso l'utilizzo dei container, verrà esportato per piattaforma web, Android e Apple.

Riassumendo, lo scopo del lavoro illustrato in questa tesi è progettare e implementare un prototipo di rete distribuita di telemedicina basata sugli ultimi standard di interoperabilità in ambito sanitario, conforme alle indicazioni del FSE 2.0, che vede al centro dell'architettura la tecnologia blockchian, in grado di interfacciarsi con le altre soluzioni presenti nel territorio nazionale e con l'imminente Piattaforma Nazionale di Telemedicina.

Capitolo 2

Contesto applicativo

Il Piano nazionale di ripresa e resilienza (PNRR), attraverso la Missione 6 “Salute”, intende rendere le strutture italiane più moderne, digitali e inclusive, favorendo lo sviluppo della sanità digitale, che fonda la sua strategia su due pilastri: il nuovo Fascicolo Sanitario Elettronico (FSE) e la piattaforma nazionale di telemedicina. Pilastri che sono mutuamente connessi e basati su di un’architettura condivisa, su strutture cloud e logiche a microservizi. [3]

Nello specifico la gestione dei dati e l’erogazione dei servizi da parte dell’architettura appena descritta viene realizzata dall’Ecosistema Dati Sanitari (EDS) secondo lo standard internazionale HL7-FHIR.

Nelle sezioni che seguono verranno approfondite le tre componenti appena introdotte al fine di fornire un adeguato livello di dettaglio per comprendere il contesto applicativo del sistema descritto in questo documento di tesi.

2.1 Fascicolo sanitario elettronico

Il Fascicolo Sanitario Elettronico rappresenta uno strumento importante per le iniziative che si inseriscono nell'ambito della Sanità Digitale, oltre che per il miglioramento della qualità delle cure che le strutture sanitarie, a tutti i livelli, offrono all'assistito. Il FSE è costituito da un insieme di dati e documenti digitali di tipo sanitario e socio-sanitario generati da eventi clinici, riguardanti l'assistito, riferiti a prestazioni erogate dal Servizio Sanitario Nazionale (SSN) e/o anche da strutture sanitarie private. [4]

Riassumendo, il FSE è uno strumento con modalità di integrazione e di accesso in rete, che permette di operare con i dati provenienti da applicazioni cliniche eterogenee. In sostanza, esso racchiude al suo interno tutti i dati sanitari di un paziente, raccolti all'interno delle diverse Cartelle Cliniche Elettroniche (CCE) di proprietà di ogni struttura sanitaria a cui l'assistito si è rivolto nel corso della sua vita.

Un errore molto comune è ritenere che implementare la CCE equivalga a portare in digitale la tradizionale Cartella Clinica Cartacea. La CCE, invece, si configura come un sistema informatico integrato aziendale in sostituzione della cartella clinica cartacea, che da un lato ne rispetti i requisiti e le funzioni, e dall'altro risolva alcune criticità ad essa legate, offrendo l'opportunità di aumentarne il valore attraverso l'integrazione con altri strumenti informatici.

La Cartella Clinica Elettronica è oggi in grado di assolvere tutti i compiti formalmente definiti per la cartella clinica cartacea ma è necessario che lo faccia secondo la logica di una efficace ed efficiente gestione elettronica del dato. Per questo motivo, una visione dello strumento di Cartella Clinica Elettronica, come il mero "digitalizzatore" del cartaceo, da implementare senza un'adeguata revisione dei processi interni, è riduttiva – se non errata – e non permette di valorizzare il potenziale (in termini di gestione integrata delle informazioni, tempestività, automazione e semplificazione) offerto da questo strumento digitale. [5]

Approfondendo il concetto di FSE, esso è lo strumento attraverso il quale il cittadino può tracciare e consultare tutta la storia della propria vita sanitaria, condividendola con i professionisti sanitari per garantire un servizio più efficace ed efficiente. Le informazioni presenti nel Fascicolo del cittadino vengono fornite e gestite dalle singole regioni.

Il nuovo FSE nazionale (versione 2.0) deve rappresentare il punto unico di accesso ai servizi sanitari online attraverso la trasformazione da "archivio di documenti" a

“ecosistema di dati e servizi". Gli obiettivi del FSE sono quindi:

- la realizzazione di meccanismi uniformi di trasmissione di dati e documenti al FSE a livello nazionale;
- la possibilità di avere una comprensione univoca del significato del dato;
- la possibilità di un confronto quantitativo sicuro dei risultati.

Dal punto di vista architetturale, l'obiettivo di rendere diffuso e uniforme sul territorio nazionale l'utilizzo e l'alimentazione del FSE si raggiunge rafforzando i meccanismi di interoperabilità della gestione documentale, realizzando un servizio di validazione del formato del dato e del documento strettamente collegato al processo di refertazione o in generale ai processi clinici. Questi elementi permetteranno:

- la conversione di documenti in dato secondo formato standard HL7-FHIR (approfondito nella sezione 2.2);
- la raccolta del dato;
- l'erogazione di servizi, sia con finalità di cura che di prevenzione, basati sul dato. [3]

2.2 FHIR

Per via delle motivazioni già enunciate, all'interno del Sistema Sanitario Nazionale è necessario attuare una transizione verso le CCE con l'implementazione del FSE 2.0. Man mano che i pazienti si spostano nell'ecosistema sanitario, le loro cartelle cliniche elettroniche devono essere disponibili, individuabili e comprensibili. Inoltre, per affiancare il supporto decisionale clinico automatizzato e altre elaborazioni realizzate tramite software, i dati devono anche essere strutturati e standardizzati.

FHIR (Fast Healthcare Interoperability Resource) è uno standard di interoperabilità sviluppato e progettato da HL7 (l'associazione che gestisce gli standard Health Level 7) per consentire lo scambio di dati sanitari in formato elettronico tra sistemi diversi del settore sanitario.

Purtroppo, nel sistema sanitario odierno, i dati dei pazienti possono essere spesso frammentati e incompleti. Ad esempio, si ponga il caso in cui un medico abbia bisogno dei risultati di un test allergico: per fare una valutazione corretta in realtà ha bisogno dell'intera storia del paziente, oltre all'ordine del test allergico. FHIR è stato progettato per riunire tutte queste informazioni, in modo che gli operatori

possano disporre facilmente di tutto ciò di cui hanno bisogno per poter operare nelle condizioni migliori.

Nel dettaglio, FHIR utilizza le interfacce di programmazione delle applicazioni (API) per consentire a diverse applicazioni di "collegarsi" a un sistema operativo di base, inserendo tutte le informazioni rilevanti nel flusso di lavoro del fornitore.

Il protocollo, quindi, offre anche strumenti e risorse riutilizzabili per fornitori, pazienti, organizzazioni e dispositivi. Questi strumenti comprendono una varietà di concetti clinici, tra cui farmaci, problemi, diagnosi, piani di cura e informazioni finanziarie. Essi sono molto efficaci per migliorare l'interoperabilità. FHIR può essere utilizzato anche in combinazione con delle applicazioni mobili attraverso la condivisione di dati basati sulla cartella clinica elettronica, la comunicazione cloud e tra fornitori istituzionali di servizi sanitari. La funzione principale di FHIR è quella di incorporare le risorse (o i dati relativi a un paziente) nei sistemi esistenti, eliminando la necessità di sviluppare o riprogettare il sistema esistente; supporta, inoltre, la condivisione di informazioni in diversi formati, tra cui documenti, messaggi, servizi e interfacce RESTful. [6]

2.3 Telemedicina

L'intero contesto di innovazione sanitaria descritto finora, è affiancato dalla necessità di alleggerire il carico di assistiti nelle strutture fisiche ove possibile. Una soluzione in tal senso giunge dalla telemedicina. Con tale termine si indica l'insieme di prestazioni sanitarie in cui, grazie all'utilizzo di tecnologie innovative, il professionista della salute e il paziente non si trovano nello stesso luogo.

La telemedicina consente di:

- assistere e fare visite di controllo ai pazienti;
- controllare a distanza i parametri vitale di pazienti;
- far dialogare professionisti sanitari per consulti su particolari casi clinici;
- inviare e ricevere e condividere documenti, diagnosi e referti. [7]

2.3.1 Vantaggi della telemedicina

La telemedicina si rende indispensabile soprattutto per categorie di persone che richiedono un'assistenza continuativa, in quanto, ad esempio, affette da patologie

croniche. Questi pazienti possono aver bisogno di un costante monitoraggio di alcuni parametri vitali, per ridurre il rischio d'insorgenza di complicazioni.

In questo caso si può fornire al paziente un servizio migliore, attraverso una più rapida disponibilità di informazioni sullo stato della sua salute, consentendo di accrescere la qualità e tempestività delle decisioni dei professionisti sanitari, particolarmente utili in condizioni di emergenza-urgenza. In questo senso, la tecnologia e le telecomunicazioni svolgono una funzione fondamentale in quanto contribuiscono a migliorare l'efficienza e la sicurezza delle cure, nonché la riservatezza e protezione dei dati personali dei pazienti.

La telemedicina, che rappresenta uno dei principali ambiti di applicazione della sanità in rete, offre potenzialità di grande rilevanza soprattutto per:

- accrescere l'equità nell'accesso ai servizi socio-sanitari nei territori remoti, grazie al decentramento e alla flessibilità dell'offerta di servizi resi, la cui erogazione viene resa possibile da forme innovative di assistenza domiciliare;
- ridistribuire le risorse umane e tecnologiche tra diversi presidi, consentendo di coprire la necessità di competenze professionali spesso carenti ed assicurare la continuità dell'assistenza sul territorio;
- offrire, grazie alla disponibilità di servizi di teleconsulto (definito nella sezione 2.3.2.2), un supporto ai servizi mobili d'urgenza o per le zone remote, attraverso la riorganizzazione dei servizi sanitari, eventualmente mediante l'utilizzo di risorse cliniche a distanza, anche dislocate direttamente a bordo delle ambulanze. [7]

2.3.2 Tipologie di telemedicina

In questa sezione vengono declinate le diverse tipologie di servizi di telemedicina, andando a descriverne dettagli, funzionalità e obiettivi.

2.3.2.1 Televisita

"La televisita è un atto medico in cui il professionista interagisce a distanza in tempo reale con il paziente, anche con il supporto di un caregiver. La televisita, come previsto anche dal codice di deontologia medica, non può essere mai considerata l'unico mezzo per condurre la relazione medico-paziente, né può essere considerata in modo automatico sostitutiva della prima visita medica in presenza. Il medico è deputato a decidere in quali situazioni e in che misura la televisita può essere

impiegata in favore del paziente, utilizzando anche gli strumenti di telemedicina per le attività di rilevazione, o monitoraggio a distanza, dei parametri biologici e di sorveglianza clinica. La televisita è da intendersi limitata alle attività di controllo di pazienti la cui diagnosi sia già stata formulata nel corso di visita in presenza. [...]

È necessaria una dotazione tecnologica di base per la videochiamata integrata da strumenti che consentano di consultare la documentazione clinica (referti, immagini, ecc.). Prima della televisita e durante la stessa deve essere sempre garantita la possibilità di scambiare in tempo reale dati clinici, referti medici, immagini audio e video. Le informazioni del paziente devono essere disponibili nella cartella domiciliare alla quale devono poter accedere il medico e i professionisti sanitari che intervengono, e tutta l'équipe laddove presente." [8]

2.3.2.2 Teleconsulto medico

"È un atto medico in cui il professionista interagisce a distanza con uno o più medici per dialogare, anche tramite una videochiamata, riguardo la situazione clinica di un paziente, basandosi primariamente sulla condivisione di tutti i dati clinici, i referti, le immagini, gli audio-video riguardanti il caso specifico. Tutti i suddetti elementi sono condivisi per via telematica sottoforma di file digitali idonei per il lavoro che i medici in teleconsulto ritengono necessari per l'adeguato svolgimento del loro lavoro. Il teleconsulto tra professionisti può svolgersi anche in modalità asincrona, quando la situazione del paziente lo permette in sicurezza. Quando il paziente è presente al teleconsulto, allora si svolge in tempo reale utilizzando le modalità operative analoghe a quelle di una televisita e si configura come una visita multidisciplinare. [...]

È necessaria una dotazione tecnologica di base per la videochiamata integrata da strumenti che consentano di consultare ed inviare la documentazione clinica (referti, immagini, ecc.). Le informazioni del paziente devono essere disponibili nella cartella domiciliare." [8]

2.3.2.3 Teleconsulenza medico-sanitaria

"È un'attività sanitaria, non necessariamente medica ma comunque specifica delle professioni sanitarie, che si svolge a distanza ed è eseguita da due o più persone che hanno differenti responsabilità rispetto al caso specifico. Essa consiste nella richiesta di supporto durante lo svolgimento di attività sanitarie, a cui segue una videochiamata in cui il professionista sanitario interpellato fornisce all'altro, o agli

altri, indicazioni per la presa di decisione e/o per la corretta esecuzione di azioni assistenziali rivolte al paziente. La teleconsulenza può essere svolta in presenza del paziente, oppure in maniera differita. In questa attività è preminente l'interazione diretta tramite la videochiamata, ma è sempre necessario garantire all'occorrenza la possibilità di condividere almeno tutti i dati clinici, i referti le immagini riguardanti il caso specifico. È un'attività su richiesta ma sempre programmata e non può essere utilizzata per surrogare le attività di soccorso. [...]

La dotazione base che consente la videochiamata è integrata dagli strumenti che servono a documentare le condizioni cliniche che sono fonte del quesito di consultazione: dispositivi per la gestione e lo scambio di dati e immagini, dispositivi per il monitoraggio, per la riabilitazione, ecc. Le informazioni relative alla presa in carico domiciliare, essenzialmente contenute nella cartella domiciliare, devono essere disponibili al medico e ai professionisti sanitari, tra cui si realizza la teleconsulenza." [8]

2.3.2.4 Teleassistenza

"È un atto professionale di pertinenza della relativa professione sanitaria (infermiere, fisioterapista, logopedista, ecc.) e si basa sull'interazione a distanza tra il professionista e paziente/caregiver per mezzo di una videochiamata, alla quale si può all'occorrenza aggiungere la condivisione di dati referti o immagini. Il professionista che svolge l'attività di teleassistenza può anche utilizzare idonee APP per somministrare questionari, condividere immagini o video tutorial su attività specifiche. Lo scopo della teleassistenza è quello di agevolare il corretto svolgimento di attività assistenziali, eseguibili prevalentemente a domicilio. La teleassistenza è prevalentemente programmata e ripetibile in base a specifici programmi di accompagnamento al paziente. [...]

La strumentazione prevista per la teleassistenza è costituita da dispositivi per la registrazione, archiviazioni dei dati e delle immagini, supporti per lo scambio dei dati e delle immagini, video e parametri vitali, dispositivi fissi e/o mobili che prevedano un facile utilizzo, dispositivi medici e sensori di rilevamento; inoltre è prevista la fruizione di APP, video e materiali informativi/formativi accessibili per l'assistito e/o caregiver. Ulteriori possibilità sono date dal collegamento e consultazione di dati o informazioni rilevate durante il servizio stesso. Le informazioni presenti in piattaforma di teleassistenza sono:

- dati periodicamente rilevati dal professionista;
- dati periodicamente auto-rilevati e inseriti dall'assistito/caregiver;
- documenti: referti e note delle varie visite/interventi eseguiti, valutazioni da parte dei professionisti sanitari, o da parte del medico che ha in gestione il caso e delle azioni intraprese (aggiornamento dieta, aggiornamento terapie farmacologiche o altro, in base alle variazioni dello stato di salute dovuto alla patologia);
- dati periodicamente rilevati dal telemonitoraggio (ove previsto)." [8]

2.3.2.5 Telemonitoraggio

"Permette il rilevamento e la trasmissione a distanza di parametri vitali e clinici in modo continuo, per mezzo di sensori che interagiscono con il paziente (tecnologie biometriche con o senza parti applicate). Il set di tecnologie a domicilio, personalizzato in base alle indicazioni fornite dal medico, deve essere connesso costantemente al sistema software che raccoglie i dati dei sensori, li integra se necessario con altri dati sanitari e li mette a disposizione degli operatori del servizio di telemedicina in base alle modalità organizzative stabilite. I dati devono sempre comunque essere registrati in locale presso il paziente e resi disponibili all'occorrenza, per maggiore garanzia di sicurezza. Il sistema di telemonitoraggio, che può essere integrato dal telecontrollo medico e affiancato dal teleconsulto specialistico, è sempre inserito all'interno del sistema di telemedicina che garantisce comunque l'erogazione delle prestazioni sanitarie necessarie al paziente. Obiettivo del telemonitoraggio è il controllo nel tempo dell'andamento dei parametri rilevati, permettendo sia il rilevamento di parametri con maggiore frequenza e uniformità di quanto possibile in precedenza, sia la minore necessità per il paziente di eseguire controlli ambulatoriali di persona. [...]

Il telemonitoraggio avviene attraverso un set di dispositivi tecnologici certificati e collegati ad una piattaforma centrale per la ricezione e la conservazione dei dati raccolti. L'accesso alla piattaforma di telemonitoraggio e ai dati raccolti dai dispositivi deve essere consentito al medico che ha richiesto il telemonitoraggio e all'équipe individuata nel PAI[1].

La funzionalità di telemonitoraggio può essere integrata all'interno di piattaforme per la telemedicina e con la cartella domiciliare. Le informazioni presenti in

[1]Il PAI (Piano Assistenziale Individualizzato), è il documento di sintesi che raccoglie e descrive in ottica multidisciplinare le informazioni relative agli assistiti

piattaforma di telemonitoraggio utili alla presa in carico domiciliare sono:

- dati periodicamente rilevati dal sistema di telemonitoraggio, Al fine di garantire la continuità informativa nella presa in carico del paziente, una parte di essi può anche essere riportata nella cartella domiciliare prevedendone l'importazione automatica (es. quando si manifesti un significativo cambiamento dello stato clinico del paziente);
- valutazioni periodiche del telemonitoraggio elaborate dal MMG/PLS/specialista che ha in gestione il caso ed azioni intraprese (es. a seguito degli allarmi)." [8]

2.3.2.6 Telecontrollo

"Il telecontrollo medico consente il controllo a distanza del paziente. Tale attività è caratterizzata da una serie cadenzata di contatti con il medico, che pone sotto controllo l'andamento del quadro clinico, per mezzo della videochiamata in associazione con la condivisione di dati clinici raccolti presso il paziente, sia prima che durante la stessa videochiamata. Questo per patologie già diagnosticate, in situazioni che consentano, comunque, la conversione verso la visita di controllo tradizionale in tempi consoni a garantire la sicurezza del paziente e in ogni caso sempre sotto responsabilità del medico che esegue la procedura. [...]

Il telecontrollo avviene attraverso un set di dispositivi tecnologici certificati e collegati ad una piattaforma centrale per la ricezione e la conservazione dei dati raccolti. L'accesso alla piattaforma di telecontrollo e la valutazione dei dati raccolti/trasmessi dal paziente (PGHD) devono essere consentiti al medico che ha richiesto il telecontrollo." [8]

I dati di cui si tiene traccia durante il telecontrollo, utili alla presa in carico domiciliare, sono gli stessi riportati nel paragrafo precedente riguardante il telemonitoraggio.

2.3.2.7 Teleriabilitazione

"Consiste nell'erogazione a distanza di prestazioni e servizi intesi ad abilitare, ripristinare, migliorare, o comunque mantenere il funzionamento psicofisico di persone di tutte le fasce d'età, con disabilità o disturbi, congeniti o acquisiti, transitori o permanenti, oppure a rischio di svilupparli. È un'attività sanitaria di pertinenza dei professionisti sanitari, può avere carattere multidisciplinare e, quando ciò costituisca un vantaggio per il paziente, può richiedere la collaborazione dei caregiver,

familiari e non, e/o di insegnanti. Per il completamento dei trattamenti volti a tutelare la salute dei cittadini, come qualsiasi intervento riabilitativo "tradizionale", le prestazioni di teleriabilitazione trovano complementarità con altre prestazioni di cura, riabilitazione, assistenza o prevenzione, attuate esse stesse in presenza o in telemedicina. Le prestazioni e i servizi di teleriabilitazione possono essere fruiti da qualsiasi luogo assistenziale e/o educativo in cui si trova il paziente (es. strutture sanitarie, residenze sanitarie o sociosanitarie, istituti penitenziari, case-famiglia, comunità residenziali, scuole, istituti di formazione, università, contesti comunitari o luoghi di lavoro basati sulla comunità, domicilio). Per alcuni di essi è inoltre possibile la fruizione in mobilità, ovvero da luoghi, non ordinariamente prestabiliti per la riabilitazione. Nelle attività di teleriabilitazione vanno comprese anche quelle volte alla valutazione a distanza del corretto utilizzo di ausili, ortesi e protesi durante le normali attività di vita condotte all'interno dell'ambiente domestico o lavorativo. [...]

La strumentazione base che consente la videochiamata viene integrata da dispositivi per la gestione e lo scambio di dati e immagini, principalmente attraverso l'uso di dispositivi medici con relativa marcatura CE, incluso quelli mobili e indossabili, sensori, robotica, serious games[2] e sarà fornita dall'Azienda sanitaria alla stessa stregua dei percorsi di presa in carico in presenza." [8]

2.4 Stato dell'arte delle piattaforme di telemedicina

2.4.1 Piattaforma Nazionale di Telemedicina

La Piattaforma Nazionale di Telemedicina (PNT) è strutturata in componenti, integrate ma distinte nel processo di esecuzione:

- Infrastruttura Regionale di Telemedicina che, nelle sue istanze regionali, realizza almeno i servizi minimi della Piattaforma Nazionale; le strutture pubbliche e private regionali alimentano il FSE con dati, documenti, ed eventi legati al percorso di telemedicina.
- Piattaforma/Infrastruttura Nazionale di Telemedicina che ha un compito di governance, pianificazione e monitoraggio dei servizi e dei processi di telemedicina. [9]

[2]Tecnologie che sfruttano gli elementi ludici per facilitare gli obiettivi di abilitazione e riabilitazione nel rispetto delle norme vigenti e in sicurezza.

In figura 1 è riportata l'architettura a cui entrambe le componenti fanno riferimento.

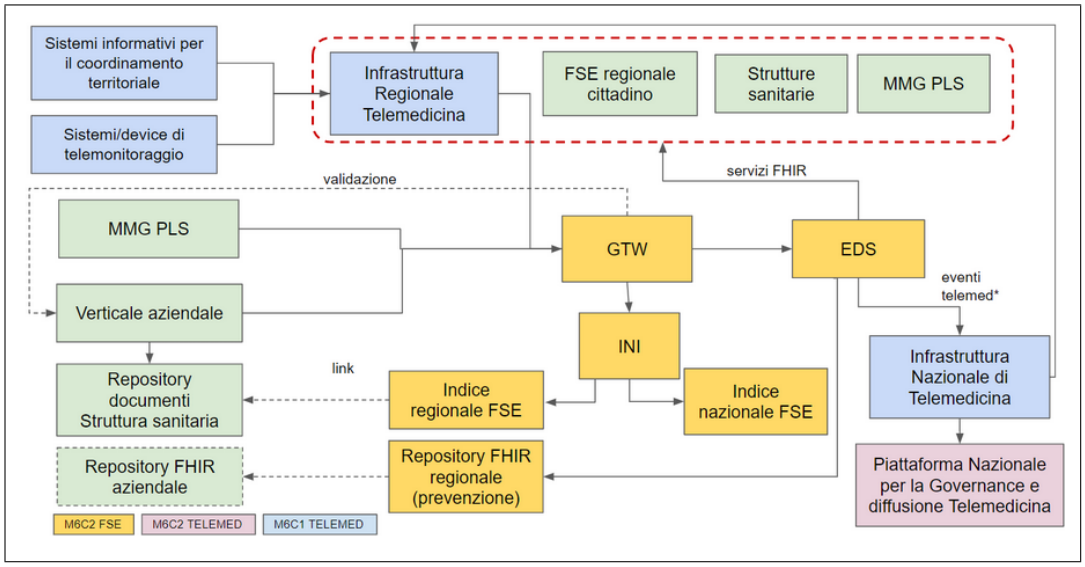

Figura 1: Architettura Piattaforma Nazionale di Telemedicina

Il sistema FSE raccoglie tramite il gateway (in Figura 1 GTW), dati ed eventi prodotti dai soggetti erogatori, introducendo la gestione di dati ed eventi tramite l'Ecosistema Dati Sanitari (EDS), che offre sia componenti di memorizzazione (Data Repository) che un layer di servizi per l'accesso al dato. Difatti, la Piattaforma Nazionale di Telemedicina (PNT) ed EDS sono integrati per lo scambio dei dati di comune interesse.

L'EDS, nello specifico, contiene dati atomici, anche personali degli assistiti, secondo standard internazionale HL7-FHIR e realizza servizi secondo processi di erogazione regolati dallo stesso standard. Il Ministero della salute è titolare del trattamento dei dati raccolti e generati dall'EDS, la cui gestione operativa è affidata all'AGENAS.

La gestione documentale è realizzata dagli Indici (Registry) regionali che indicizzano i documenti memorizzati presso i sistemi informativi delle strutture sanitarie e dall'Infrastruttura Nazionale per l'Interoperabilità (INI) e dall'Indice (Registry) nazionale, che consentono l'accesso ai documenti da una regione diversa da quella di produzione del dato e abilitano le funzioni di trasferimento degli indici dei documenti che costituiscono il fascicolo.

In figura 2 è mostrato un esempio interoperabilità delle componenti appena descritte, alimentando la piattaforma con un referto di televisita. Il referto prodotto

viene sottoposto al gateway per una validazione sintattica e semantica, in seguito alla quale viene memorizzato nel repository documentale e quindi inviato nuovamente al gateway per la conversione in FHIR, che ne permette la memorizzazione e gestione attraverso INI (Infrastruttura Nazionale per l'Interoperabilità dei Fascicoli Sanitari Elettronici) all'interno del registry regionale e nazionale. L'EDS provvede, infine, a comunicare alla Infrastruttura Nazionale di Telemedicina l'evento. [9]

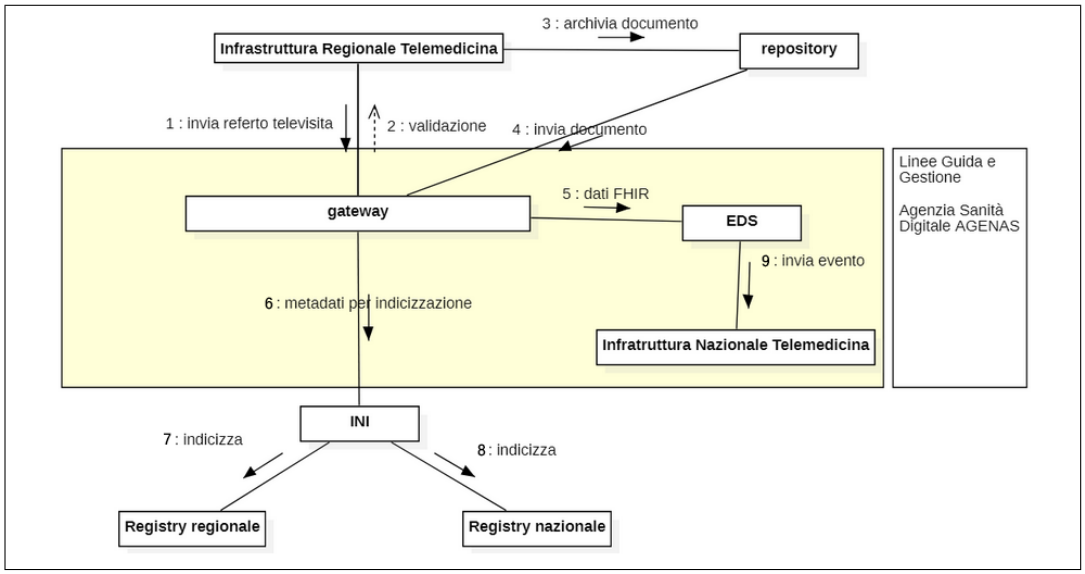

Figura 2: Alimentazione della PNT con un referto di televisita

Processo simile avviene all'innesco della gestione nelle due piattaforme dei dati di telemonitoraggio. I dati prodotti dai dispositivi medici in regime di telemonitoraggio continuo possono essere ingenti e non tutti rilevanti. L'Infrastruttura di Telemedicina Regionale memorizza tutti i dati di monitoraggio e, nel caso in cui si presenti un valore critico o di particolare rilievo, viene generato un evento di alert e inviato ad EDS insieme ai dati correlati all'evento. Ciò assicura l'alimentazione dell'EDS con i soli dati di rilievo tralasciando la memorizzazione di tutto lo storico.

Il processo appena descritto è mostrato in figura 3.

2.4.1.1 Tempistiche nello sviluppo della piattaforma

La durata della concessione Agenas[3] per la realizzazione e gestione della Piattaforma nazionale di Telemedicina è di 10 anni ed è organizzata in tre fasi:

- La prima fase, denominata "Start Up", si dovrebbe concludere entro novembre 2023 con il collaudo e l'attivazione della Piattaforma.

[3] Agenzia Nazionale per i Servizi Sanitari Regionali.

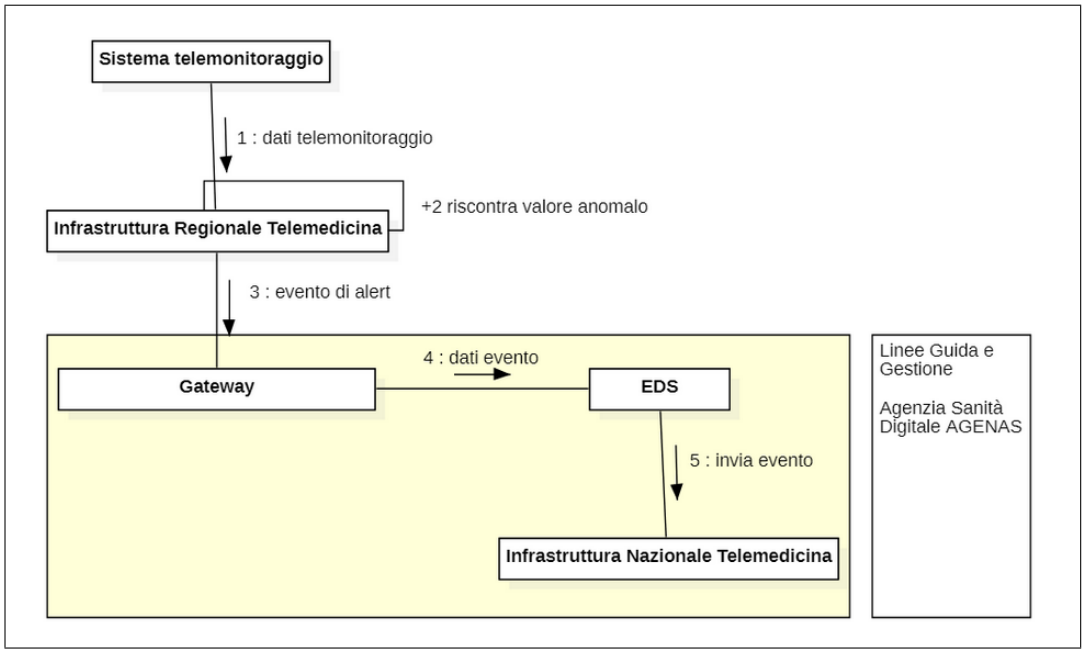

Figura 3: Alimentazione della PNT con un evento alert relativo al telemonitoraggio

- La seconda fase, definita di "Avvio e Consolidamento", durerà due anni e dovrebbe terminare entro novembre 2025.
- A partire dal mese di dicembre 2025, invece, verrà avviata la fase di "Gestione a regime" fino al termine della concessione, in cui la Piattaforma continuerà a incrementare i suoi servizi, in linea con le evoluzioni della Sanità digitale. [10]

Come si può notare dalle date appena riportate, attualmente, è possibile inquadrare lo stato di sviluppo della PNT all'interno della prima fase.

Nel momento in cui si redige tale documento di tesi, il processo di implementazione sta incontrando in una serie di ostacoli e ritardi dovuti ad azioni attuate dal Garante della Privacy, volte a rendere la piattaforma conforme alle leggi in vigore.

Sebbene il processo di sviluppo della PNT si trovi in uno stadio iniziale, si ritiene che abbia comunque senso affrontare il progetto descritto nei capitoli che seguono. Difatti, l'implementazione della Piattaforma nazionale di Telemedicina garantisce un forte orientamento all'integrazione e all'interoperabilità con i sistemi dispiegati a livello nazionale oltre a mantenere, al tempo stesso, una spiccata facilità di integrazione con gli attuali e futuri ecosistemi digitali sia regionali che nazionali coinvolti. [9]

In conclusione, ponendosi come obiettivo la realizzazione di un prototipo di una piattaforma di telemedicina distribuita, che sia fortemente integrabile con le soluzioni

già esistenti e con quelle che saranno sviluppate in futuro (tra cui la PNT), si evita di incorrere nel rischio di creare un sistema che poi non sia in grado di integrarsi nell'ecosistema nazionale di telemedicina.

2.4.2 Soluzioni esistenti

Attualmente sono presenti diverse soluzioni che operano nel campo della telemedicina all'interno dello scenario nazionale e internazionale. Poche di esse, però, sono rivolte all'interezza delle declinazioni che la telemedicina può avere e ai campi sanitari a cui può essere applicata. Molte delle soluzioni esistenti, infatti, si riferiscono a specifiche necessità sorte in ben delineati ambiti sanitari.

Inoltre, nessuna soluzione degna di nota è conforme al nuovo Fascicolo Sanitario Elettronico gestendo al contempo i dati sanitari attraverso standard FHIR. Possibile causa di questa assenza di soluzioni può essere ricercata nel tempismo con cui si discutono questi argomenti; difatti, come già descritto nella sezione 2.4.1, la stessa Piattaforma Nazionale di Telemedicina è in piena fase di sviluppo.

Di seguito vengono descritte alcune soluzioni che rispecchiano quanto riportato finora. Nella scelta degli esempi di piattaforme di telemedicina da inserire in questo elaborato, si è tenuta in considerazione anche la presenza di blockchain all'interno delle relative architetture, questo perché il prototipo progettato e realizzato all'interno del progetto di tesi si basa proprio su questa tecnologia; si è ritenuto opportuno, per tanto, fornire un esempio di architetture paragonabili con quella implementata.

2.4.2.1 DermoNet

DermoNet è uno strumento per il Teleconsulto dermatologico, finanziato da Regione Sardegna, utilizzato principalmente nelle zone meno servite dell'isola, che si pone l'obiettivo di sviluppare un modello di organizzazione virtuale per finalità di dermatologia telematica sviluppando una piattaforma software basata su tecnologia cloud, al fine di creare una rete che coinvolga pazienti, medici di famiglia, dermatologi e istituzioni sanitarie.

Il sistema software integrato DermoNet, in breve DermoNet System, è un prototipo pre-industriale di una piattaforma software collaborativa di servizi orientata a supportare la gestione dei pazienti dermatologici e il processo diagnostico da parte di medici generici e specialisti che lavorano insieme.

In questa architettura la blockchain ha il doppio ruolo di canale di comunicazione e framework. La piattaforma utilizza la blockchain come canale di comunicazione

tra le applicazioni decentralizzate. I messaggi tra le app sono strutturati come transazioni Ethereum. I dati blockchain sono disponibili al pubblico e, al fine di proteggere la privacy dei pazienti, ogni transazione contiene richieste o dati in forma crittografata.[11]

2.4.2.2 AaYusH

AaYusH sfrutta la connessione su rete 5G ad alta velocità e bassa latenza per garantire operazioni di chirurgia telematica efficaci. Inoltre, introduce un sistema di feedback per i pazienti, che recensiscono i chirurghi creando una classifica consultabile dai futuri pazienti.

La chirurgia telematica consente ai chirurghi, in località fisicamente distanti tra loro, di collaborare per il successo delle procedure attraverso canali di comunicazione wireless. Questa modalità di chirurgia, sfruttando la rete internet tattile 5G, ha un enorme potenziale, in quanto è in grado di fornire servizi chirurgici ultra reattivi, real time, da remoto con alta qualità e precisione, portando vantaggi importanti alla società moderna. Tuttavia, i sistemi "telechirurgia" esistenti presentano problemi di sicurezza, privacy, latenza e costi di archiviazione blockchain, che ne limitano l'applicabilità nell'immediato.

Il sistema di Telechirurgia AaYusH, per sopperire al problema della scalabilità, sfrutta una blockchain pubblica, Ethereum, basata su smart contract scritti in Solidity con la suite Truffle, con diversi permessi e ruoli definiti per medici e pazienti, mentre i dati vengono raccolti in un database che offre grande capacità di archiviazione e bassa latenza. [12]

2.4.2.3 MedicalChain

Medicalchain è una piattaforma decentralizzata (utilizzata prevalentemente in Inghilterra) che consente lo scambio e l'utilizzo sicuro, veloce e trasparente dei dati medici. Viene utilizzata la tecnologia blockchain per creare una cartella clinica elettronica associata all'utente e mantenere un'unica versione dei dati relativi a esso. Gli utenti possono fornire accesso condizionato ai propri dati a diversi agenti sanitari come medici, ospedali, laboratori, farmacisti e assicuratori. Ogni interazione con i dati medici viene registrata come transazione sul libro mastro distribuito di Medicalchain. Questa tecnologia si basa sull'architettura Hyperledger Fabric basata su autorizzazioni che consente diversi livelli di accesso; gli utenti, in definitiva, controllano chi può visualizzare i propri record, in che quantità e per quanto tempo.

Una criticità riscontrata in questa soluzione risiede nel fatto che la gestione della stessa sia affidata nella sua interezza agli utenti. Si ritiene necessario dare lo stesso diritto di accesso alla tecnologia a tutti gli assistiti a prescindere dalla capacità di quest'ultimi di operare in una piattaforma che, fisiologicamente, non può risultare fruibile a chiunque. Si ritiene necessario, quindi, che ad interfacciarsi con la piattaforma sia un personale autorizzato e formato in grado di operare al meglio all'interno della stessa. [13]

2.4.2.4 FHIRChain

Si premette che la tecnologia descritta in questa sezione è stata solo teorizzata e non sembrano esistere delle soluzioni che, alla data in cui viene redatto tale elaborato di tesi, la implementino.

FHIRChain è una piattaforma che si basa su due elementi chiave: il protocollo FHIR e la tecnologia blockchain. Nello specifico, FHIRCain crea puntatori di riferimento a database e scambia questi puntatori tramite il componente blockchain anziché inviare attraverso essa i dati effettivi.

Un vantaggio di questa soluzione lo si può riscontrare per le cliniche di telemedicina o le cliniche nelle aree rurali dove questo approccio può superare i limiti della potenza della rete consentendo la condivisione scalabile dei dati senza richiedere il caricamento degli stessi in un altro repository centralizzato, attraverso il quale i dati possono essere condivisi e scaricati da altre parti.

Una possibile criticità si può ritrovare nel fatto che la blockchain utilizzata sia di tipo pubblico e, sebbene i riferimenti ai dati siano criptatati con chiave asimmetrica basata su identità digitale, è preferibile non consentire un accesso incontrollato alla rete. [14]

Capitolo 3

Obiettivi del progetto

L'obiettivo principale del progetto descritto in questo elaborato di tesi è lo sviluppo di un prototipo per una piattaforma di Telemedicina distribuita, basata sui paradigmi e standard definiti da FSE 2.0. L'architettura della piattaforma, nel dettaglio, deve permettere la cooperazione tra nodi trusted mediante rete Blockchain al fine di permettere la condivisione controllata dei parametri clinici sottoposti a telemonitoraggio e raccolti tramite i dispositivi medici collegati a interfacce di normalizzazione del dato.

In questo capitolo vengono approfonditi i requisiti, funzionali e non, che sono stati individuati al fine di creare una piattaforma che possa da un lato rispettare gli standard vigenti in materia e dall'altro risultare fruibile, efficiente ed efficace così da garantire una buona esperienza agli utenti che si interfacceranno con essa.

3.1 Requisiti funzionali

I requisiti funzionali definiscono cosa deve fare il sistema per soddisfare le esigenze o le aspettative dell'utente. In questa specifica circostanza, tali requisiti, sono stati redatti in seguito a un'analisi del contesto sanitario nazionale e delle esigenze delle aziende che operano al suo interno. Di seguito vengono riportati i risultati dell'analisi.

3.1.1 Gestione del consenso al trattamento dei dati

I dati personali degli assistiti, che vengono trattati all'interno della piattaforma che si sta progettando possono essere di diversa natura:

- *Dati personali comuni*[1] come cognome, nome, sesso, data di nascita, luogo di nascita, provincia di nascita, indirizzo e luogo di residenza, provincia di residenza; essi possono essere raccolti in occasione della presa in carico dell'assistito da parte di una struttura sanitaria facente parte della rete di telemedicina distribuita.
- *Dati personali particolari* come i dati relativi alla salute e, se strettamente necessario, i dati idonei a rilevare la vita sessuale, l'origine etnica o razziale, le convinzioni religiose, filosofiche o di altro genere, nonché dati genetici e dati biometrici.
- Ulteriori dati forniti volontariamente dall'assistito o da un caregiver.

È necessario che gli assistiti rilascino il consenso al trattamento dei propri dati e alla relativa condivisione qualora si ritenesse necessario.

Si giunge così alla definizione del requisito per cui la piattaforma deve essere in grado di gestire i dati per cui è stato rilasciato il consenso da parte dell'assistito e per il solo periodo in cui il consenso è effettivamente vigente, il tutto sottostando alle norme che regolano tali attività.

Senza rilasciare il consenso al trattamento dei propri dati, quindi, il paziente non potrà accedere ai servizi di telemedicina offerti dalla piattaforma in oggetto.

[1] I dati personali sono categorizzati in "comuni" e "particolari" dal sistema di protezione stabilito dal Reg. UE 679/2016 (noto anche come RGPD o GDPR).

3.1.2 Indicizzazione condivisa degli eventi di telemedicina

Una delle esigenze evidenziata dall'analisi dei requisiti sta nella necessità di avere a disposizione dei vari professionisti, che operano nell'ambito della telemedicina, un registro condiviso in cui andare a indicizzare gli eventi che vengono erogati dalle varie strutture sanitarie per ogni paziente. In questo modo è possibile avere un quadro completo della situazione di ogni assistito e, di conseguenza, calibrare al meglio le eventuali azioni da intraprendere.

Questo registro deve poter essere alimentato e consultato da tutte le strutture secondo i permessi associati a ognuna di esse.

Degli esempi di eventi da indicizzare sono i seguenti:

- Avvenuta esecuzione di una visita in modalità telematica;
- Inizio di un percorso di monitoraggio;
- Inizio di un percorso di teleriabilitazione.

Come si evince dagli esempi appena riportati, all'interno del registro non è necessario riportare gli esiti delle visite, i valori registrati durante il processo di telemonitoraggio o altri dati specifici; difatti, indicizzando anche queste nozioni o i relativi referti, il tutto risulterebbe meno fruibile. Nel caso in cui si vogliano approfondire le informazioni rilevate dal registro occorrerà recuperare ulteriori dati direttamente dalle strutture sanitarie che li detengono.

3.1.3 Acquisire dati dai vari sistemi di monitoraggio

I sistemi di monitoraggio utilizzati dagli assistiti, sotto prescrizione del medico curante, sono soggetti a una forte eterogeneità che si può riscontrare in diversi aspetti legati ai dispositivi stessi:

- Tipologia di parametri rilevati;
- Unità di misura (che possono variare anche per la stessa valutazione a seconda del sensore);
- Sensibilità e precisione degli strumenti;
- Frequenza con cui vengono effettuate le rilevazioni;
- Frequenza, modalità e standard di trasmissione dei dati.

Partendo dall'assunto che non è possibile creare un sistema che sia in grado di operare con tutti i dispositivi presenti sul mercato, soprattutto per garantire una

soglia di qualità minima di affidabilità degli strumenti utilizzati, è comunque necessario raccogliere i dati da sensori e dispositivi di monitoraggio di diversa natura. Di conseguenza, è altresì indispensabile che il sistema di monitoraggio, presente all'interno della piattaforma di telemedicina che si sta implementando, sia in grado di comunicare con i suddetti dispositivi. Come indicato nella sezione 3.2.1, i dati dovranno essere inviati al repository che li conterrà attraverso le modalità indicate dallo standard FHIR, si viene quindi a creare la necessità di implementare un modulo che normalizzi i dati raccolti e li allinei alla rappresentazione normata da HL7 all'interno dello standard appena citato.

3.1.4 Possibilità per l'assistito di immettere dati nel sistema

Non tutte le informazioni di cui è necessario tener traccia sono rilevabili attraverso dei sensori: per alcune di esse è necessario che l'assistito immetta i relativi dati nel sistema attraverso un applicativo.

È necessario tenere in considerazione, inoltre, che non tutti i pazienti possono trovarsi nella condizione di compiere tale operazione; in questi casi deve essere prevista la possibilità per cui un caregiver possa svolgere tale funzione per l'assistito stesso.

Degli esempi di queste informazioni possono essere:

- L'avvenuta assunzione o meno di farmaci prescritti;
- La quantità di acqua ingerita dal paziente;
- Il cibo assunto dall'assistito.

3.1.5 Accesso flessibile da parte del medico alla piattaforma

Una necessità diffusa nei medici è quella di poter aver accesso ai dati sanitari dei propri assistiti con sempre maggiore flessibilità. Questo implica che un dottore possa monitorare le condizioni dei pazienti ogni qual volta lo ritenga opportuno, indipendentemente da dove si trovi in un dato momento e, quindi, indipendentemente dal fatto che sia nel suo ufficio dinanzi alla postazione su cui è istallato il gestionale della struttura sanitaria per cui lavora o meno.

Anche grazie alla telemedicina, i dati sanitari degli assistiti e le rispettive cartelle cliniche elettroniche sono costantemente aggiornati e accessibili online in qualsiasi momento; questo fa sì che il medico avrà bisogno solamente di un dispositivo do-

tato di connessione internet (ad esempio il proprio smartphone) per poter essere a conoscenza della situazione dei propri pazienti, quando lo ritiene più opportuno.

3.1.6 Notificare al medico eventuali eventi critici

Per facilitare il monitoraggio e la fruizione dei dati sanitari dei propri assistiti, considerando che essi possono essere di un numero elevato per un singolo medico, risulta necessario un sistema che notifichi al dottore dei casi che debbano essere posti alla sua attenzione; degli esempi di aventi da notificare possono essere:

- La misurazione di un parametro fuori soglia durante un processo di telemonitoraggio;
- La pubblicazione di un referto di una televisita che il proprio assistito ha effettuato presso un'altra struttura sanitaria in seguito alla prescrizione del medico stesso.

Con l'implementazione del sistema di notifiche il medico non dovrà preoccuparsi di controllare periodicamente le condizioni di tutti i suoi assistiti, si potrà limitare a consultare i dati e le informazioni relativi alle notifiche inoltrategli dal sistema.

Le condizioni e i casi clinici dei pazienti hanno, di certo, dei livelli di criticità diversi; motivo per cui il dottore curante deve avere la possibilità di indicare al sistema quest'informazione così che le notifiche possano giungergli con modi differenti, selezionando la modalità più appropriata sulla base di questo dato. Volendo fornire un esempio, i livelli di criticità, indicabili per ogni singolo assistito, potrebbero essere di 4 tipi:

- Nessuna criticità;
- Criticità bassa;
- Criticità media;
- Criticità alta.

3.2 Requisiti non funzionali

I requisiti non funzionali rappresentano le proprietà e le caratteristiche relative ad sistema, come caratteristiche del processo di sviluppo e standard da adottare. Essi non riguardano direttamente le specifiche funzioni erogate dal sistema, ma possono

sia definire i vincoli ai quali il sistema deve sottostare, sia riferirsi a caratteristiche che si desidera il sistema presenti.

3.2.1 Trasmissione dei dati sanitari secondo lo standard FHIR

Come già espresso nel capitolo 2, lo standard per la trasmissione di dati sanitari indicato per il nuovo Fascicolo Sanitario Elettronico (FSE 2.0) e la Piattaforma Nazionale di Telemedicina è il Fast Healthcare Interoperability Resource (FHIR).

Ribadendo il problema per cui nell'ambito sanitario la condivisione dei dati rappresenta da sempre una forte criticità, la possibilità di reperire tutte le informazioni di un paziente, di una determinata situazione critica risulta di fondamentale importanza. Troppo spesso, ancora oggi, nonostante gli svariati processi di digitalizzazione avviati, i dati sanitari sono distribuiti in diversi siti con formati eterogenei. È necessario, quindi adottare uno standard che sia comune a tutte le soluzioni di digitalizzazione delle attività sanitarie.

Il prototipo di cui si stanno analizzando i requisiti deve inserirsi in un'ottica di interoperabilità con altre infrastrutture di telemedicina (concetto approfondito nella sezione 3.2.3); ciò avvalora ulteriormente la scelta di allinearsi allo standard di cui sopra, con il fine di ridurre al minimo i problemi di integrazione che potrebbero sorgere interfacciandosi con altre realtà.

Ulteriore motivazione che ha portato alla definizione di questo requisito è la necessità di creare una soluzione che sia conforme ai dettami legislativi che regolano la telemedicina, dettami che prevedono, per l'appunto, l'utilizzo dello standard FHIR per la trasmissione di dati di natura sanitaria.

3.2.2 Adattabilità del sistema rispetto a soluzioni preesistenti

Il processo di digitalizzazione della sanità italiana è stato ormai intrapreso da diversi anni e, sebbene si stia procedendo in maniera eterogenea all'interno delle varie regioni che formano la penisola, si può affermare che nella maggioranza delle strutture sanitarie, pubbliche e private, siano già presenti strumenti informatici come software gestionali o database per il salvataggio dei dati relativi agli assistiti.

La piattaforma di telemedicina che si sta sviluppando, deve quindi tenere conto della forte eterogeneità con cui sono implementate le varie soluzioni all'interno delle strutture che decideranno di aderirvi. È di fondamentale importanza, quindi, riuscire ad attuare un processo di astrazione rispetto alle singole implementazioni.

Si considerino come caso di esempio i database delle varie strutture sanitarie; essi possono presentare diverse tipologie di eterogeneità:

- *Tecnologica*: il sistema fisico di salvataggio dei dati può essere diverso in ogni singola struttura.
- *Logica*: essa si può dividere ulteriormente in
  - *Eterogeneità di modello*: modelli di rappresentazione dei dati differenti;
  - *Eterogeneità di linguaggio*: differenti linguaggi di interrogazione della base di dati.
- *Concettuale*: esprime diversi modi di rappresentare l'informazione anche se il modello dei dati è lo stesso, ad esempio si possono avere conflitti di omonimia, sinonimia, struttura degli attributi, etc.

Oltre a quanto detto, occorre tenere a mente che i dati rappresentati all'interno della piattaforma devono essere conformi allo standard FHIR; quindi, deve essere previsto un processo di normalizzazione del dato che converta i dati salvati nelle varie strutture in dati fedeli alle direttive rappresentate dalle risorse codificate in FHIR.

3.2.3 Interoperabilità con altri sistemi di telemedicina

Oltre che a dover interoperare con le tecnologie già presenti nelle strutture aderenti alla rete, è necessario tenere in considerazione che, quella che si sta sviluppando, non è l'unica piattaforma di telemedicina presente sul territorio nazionale; come già riportato nella sezione 2.4.2 sono molte le strutture di telemedicina che operano, soprattutto a livello regionale o provinciale. Si consideri inoltre la Piattaforma Nazionale di Telemedicina in piena fase di sviluppo.

È chiaro come la soluzione che si sta implementando debba convivere, collaborare e comunicare con il resto delle tecnologie appena elencate; per far sì che ciò accada è necessario porsi nella condizione per cui la piattaforma sia conforme a tutti gli standard adottati e indicati dalla PNT. La soluzione che si sta descrivendo deve essere quindi basata sui paradigmi e standard definiti da FSE 2.0., in particolare l' utilizzo di standard di interoperabilità semantica dei dati sanitari (HL7-FHIR) degli assistiti prodotti dal Sistema Sanitario Nazionale nelle sue diverse forme (strutture pubbliche, private accreditate, private) al fine della ricomposizione trasversale (tra i

diversi setting assistenziali in cui l'assistito è coinvolto) e longitudinale (nel tempo) della storia socio-sanitaria dell'assistito stesso. [15]

3.2.4 Sicurezza nella gestione e comunicazione dei dati sanitari

I dati sanitari necessitano di protezione e sicurezza sotto un duplice profilo *esterno* e *interno*. Da un lato, infatti, essi necessitano di una tutela rafforzata rispetto ai rischi di accesso indebito, alterazione, manipolazione, connessi ad attacchi cibernetici diretti a sistemi informativi sanitari.

Per altro verso, i dati sanitari necessitano di protezione dalle possibili violazioni *interne*, siano esse dovute a errori umani o a un uso improprio delle strutture informatiche.

È compito del prototipo, oggetto di questo documento di tesi, di far fronte a tali problematiche. Le tecniche necessarie al fine di garantire quanto sopra, sono:

- Allineamento al FSE 2.0 rispetto alle norme riguardanti la gestione e il trattamento dei dati, oltre che alle leggi nazionali vigenti in materia;
- Creazione di un applicativo di interfaccia che possa ridurre al minimo le possibili violazioni dovute a errori o azioni malevole (concetto approfondito nella sezione 3.2.5).

3.2.5 Usabilità

L'usabilità misura il grado di efficacia, efficienza e soddisfazione con cui il prodotto (a mezzo della sua interfaccia) consente a un utente specifico, in un contesto d'uso specifico, di raggiungere obiettivi specifici. Più nel dettaglio per essere usabile un prodotto interattivo dovrà:

- Essere adeguato ai bisogni e alle aspettative di specifici utenti che lo utilizzano in specifici contesti d'uso;
- Risultare facile da capire, da apprendere, da usare ed essere gradevole;
- Consentire di eseguire le specifiche attività lavorative in maniera corretta, veloce e con soddisfazione;
- Generare pochi errori, non critici.

In definitiva, con il termine user experience si fa riferimento alle modalità con cui un soggetto si relaziona ad un prodotto, servizio o sistema. In particolare, assu-

mono importanza le sensazioni, percezioni e reazioni che questa interazione genera nell'individuo.

Riassumendo, per i motivi appena riportati, l'applicativo di interfaccia, in tutte le sue versioni, deve essere in grado di garantire una buona esperienza d'uso agli utenti. [16]

3.2.6 Scalabilità

La scalabilità descrive la capacità di un sistema di adattarsi facilmente a un carico di lavoro maggiore.

Considerando che il numero di strutture sanitarie che potrebbero aderire alla piattaforma di telemedicina è elevato, è necessario attuare delle strategie di ottimizzazione che facciano sì che la rete sia in grado di gestire un numero crescente di nodi.

Inoltre, è bene valutare delle strategie per rendere il più immediato possibile l'inserimento di un nuovo nodo nella rete e di conseguenza l'estensione della piattaforma.

3.2.7 Distribuzione

La piattaforma che si sta sviluppando deve essere di tipo distribuito, ossia deve essere in grado di operare in più strutture sanitarie separate fisicamente e logicamente.

Nel momento in cui si progetta un applicativo distribuito è necessario tenere in considerazione una serie di elementi relativi alle strutture sanitarie che ne faranno parte quali:

- Eterogeneità;
- Autonomia.

Di conseguenza è necessario normare queste caratteristiche così da avere una linea comune a cui tutte i nodi della rete, nel caso in cui vogliano aderirvi, debbano rispettare.

Capitolo 4

Strumenti utilizzati

In questo capitolo vengono descritti gli strumenti software usati per lo sviluppo del prototipo della piattaforma di telemedicina descritta in questo documento di tesi. Di seguito vengono quindi riportati i componenti software, e le librerie che costituiscono la base tecnologica della soluzione adottata per far fronte agli obiettivi descritti nel capitolo 3.

Lo scopo di questa trattazione è quello di marcare la distinzione tra i moduli, le componenti e gli strumenti di terze parti, proprietari od open source, utilizzati nello sviluppo del prototipo e il contributo originale apportato nel progetto, riportato nel capitolo 5.

4.1 Blockchain

La blockchain è una sottofamiglia di tecnologie per la gestione di transazioni basate su un registro strutturato come una catena di blocchi contenenti le transazioni la cui validazione è affidata a un meccanismo di consenso, distribuito su tutti i nodi della rete.

Le principali caratteristiche delle tecnologie blockchain sono:

- *Decentralizzazione*: all'interno di una rete blockchain non vi è un unico soggetto centrale a cui tutti gli altri soggetti fanno riferimento ma un insieme di soggetti che operano al fine di verificare e gestire le transazioni avvenute all'interno della blockchain, garantendo così il concetto di decentralizzazione.
- *Trasparenza*: per via della decentralizzazione dell'autorità si è in grado di garantire a tutti la possibilità di verificare, di "controllare", di disporre di una totale trasparenza sugli atti e sulle decisioni.
- *Immutabilità*: tale caratteristica sta a indicare che i registri di archiviazione delle transazioni sono inalterabili e quindi di minima corruzione.
- *Consenso*: è possibile registrare nuove transazioni all'interno della rete soltanto quando la maggioranza dei partecipanti alla rete dà il proprio consenso.
- *Sicurezza*: la blockchain produce una struttura di dati con qualità intrinseche di sicurezza; si basa, infatti, su principi di crittografia, immutabilità, decentralizzazione e consenso, che garantiscono l'attendibilità delle transazioni.
- *Responsabilità*: risulta di fondamentale importanza identificare le responsabilità delle parti che interagiscono tramite transazioni blockchain eseguite puntualmente o in maniera programmata (smart contract); data l'immutabilità dei registri, la capacità di risalire al responsabile di una transazione è immediata e sicura.
- *Programmabilità*: grazie a tale caratteristica è possibile avviare delle operazioni in base all'input ricevuto. Il tutto è facilitato dalla presenza degli smart contract: codice digitale che offre una serie di garanzie a condizioni predefinite concordate tra le parti, quest'ultime possono stabilire una condizione che può avviare un'azione o una serie di azioni prestabilite.

4.1.1 Tipologie di blockchain

L'obiettivo della classificazione che segue è identificare le differenze tra le varie tipologia di blockchain non solo ad alto livello ma anche a livello di applicazione, così da spiegare le motivazioni delle scelte progettuali effettuate nell'adozione di una tipologia di rete piuttosto che un'altra.

4.1.1.1 Blockchain pubbliche

Le blockchain pubbliche sono reti peer to peer in cui chiunque ha diritto di lettura e scrittura sulla rete e può partecipare al processo di consenso. Esse non richiedono alcuna autorizzazione per poter accedere alla rete, eseguire delle transazioni o partecipare alla verifica e mining di un nuovo blocco. Non vi sono distinzioni tra i nodi: difatti, nessun utente della rete ha privilegi sugli altri, nessuno può controllare le informazioni che vengono memorizzate su di essa, modificarle o eliminarle, e nessuno può alterare il protocollo che determina il funzionamento di questa tecnologia. In questa tipologia di reti ogni nodo è potenzialmente inaffidabile, motivo per cui il meccanismo del consenso viene sviluppato in modo da prevenire eventuali azioni dannose di un singolo nodo che potrebbero compromettere i dati o le transazioni eseguite sulla rete.

4.1.1.2 Blockchain private

Le blockchain private condividono molte caratteristiche con quelle consortium, come l'essere gestite da un'organizzazione ritenuta fidata da tutti i nodi. I permessi di scrittura e lettura sono riservati a certi utenti; difatti, l'idea di base della blockchain privata è che la rete sia composta da un insieme di nodi fidati (trusted), che compongono la governance della rete. Si tratta di reti private e non visibili, che riducono drasticamente il livello di decentralizzazione, sicurezza e immutabilità in cambio di un incremento dello spazio di archiviazione, della velocità di esecuzione e della riduzione dei costi. L'organizzazione proprietaria della rete ha il potere di modificare le regole di funzionamento della blockchain stessa, rifiutando determinate transazioni in base alle regole e alle normative stabilite.

4.1.1.3 Blockchain consortium

Le blockchain consortium sono soggette a un'autorità centrale che determina chi può accedere alla rete, oltre a definire quali sono i ruoli che un utente può ricoprire

all'interno della stessa, definendo anche regole sulla visibilità dei dati registrati. Il processo di consenso è controllato da un insieme preselezionato di nodi. Controllando chi può accedervi e chi può visualizzare i dati registrati il livello di sicurezza è modulabile sulla base delle esigenze dell'organizzazione. In generale, le blockchain per cui è necessaria un'autorizzazione da parte dell'autorità centrale per farne parte sono inoltre più performanti, veloci, scalabili oltre che meno costose di quelle pubbliche (che non prevedono autorizzazioni per aderire alla rete). I motivi di questa maggiore efficienza sono i seguenti:

- hanno una dimensione e diffusione minore;
- le transazioni vengono verificate da un limitato numero di utenti rispetto alle blockchain pubbliche.

Questa tipologia di blockchain, per via delle sue caratteristiche intrinseche, può essere considerata come "parzialmente decentrata" e, quindi, un ibrido tra le soluzioni private e quelle pubbliche.

Le caratteristiche appena descritte rendono questa tipologia la più adatta a essere implementata e utilizzata nella piattaforma di telemedicina descritta in questo documento di tesi.

4.1.2 Motivazioni dell'adozione delle blockchain

L'Italia, alla fine del 2018, ha sottoscritto la dichiarazione sullo sviluppo della blockchain nell'ambito del MED7, il gruppo costituito da sette Paesi del Sud Europa (Italia, Spagna, Francia, Malta, Cipro, Grecia e Portogallo). Nella dichiarazione si riconosce che la blockchain – e in generale le tecnologie basate su registri distribuiti – possono essere determinanti nello sviluppo di questi paesi, tanto da rendere necessaria la creazione di un coordinamento – anche tecnico – tra i paesi, per sperimentare l'utilizzo di questa e di altre tecnologie emergenti (ad esempio 5G, Internet of Things, Intelligenza Artificiale).

Sotto il profilo medico-sanitario, la sintesi della strategia nazionale blockchain rileva una generale lentezza del sistema sanitario a recepire le nuove tecnologie e modificare i propri schemi organizzativi, tanto da rendere necessario lo studio di adeguate strategie di implementazione e di transizione, anche attraverso incentivi per gli operatori già insediati. La strategia nazionale blockchain, in ambito sanitario, detta alcune linee guida, riassumibili come segue:

- L'uso della tecnologia deve rispettare i limiti etici e giuridici nazionali e comunitari di settore, se esistenti, soprattutto per quanto concerne la protezione dei dati personali, la modalità di condivisione e riutilizzo dei dati clinici.
- La progettazione tecnica delle piattaforme dovrà prestare particolare attenzione alle fasi di gestione dell'identità, autenticazione e autorizzazione dei soggetti coinvolti, poiché da ciò dipenderà l'affidabilità dell'infrastruttura.
- Obiettivo primario dei sistemi blockchain medico-sanitari dovrà essere l'interoperabilità, la sicurezza e la privacy, e tale obiettivo potrà essere raggiunto anche tramite l'intervento di strutture di coordinamento e standardizzazione nel settore, sia nazionali che europee.
- Lo sviluppo della tecnologia blockchain dovrà comunque tenere conto, nel design tecnologico, dell'importanza del fattore umano, che in ambito sanitario ha un impatto sicuramente importante dal punto di vista dell'empatia con il paziente; per questo, le procedure automatizzate dovranno essere valutate sul piano antropologico e psicologico, le informazioni verso pazienti e familiari sulle nuove tecnologie dovranno essere presentate in maniera efficace, perché la mancata comprensione dei benefici da parte dei pazienti potrebbe comportare dei costi importanti.

Ad oggi, ogni ambulatorio, studio medico, ospedale, clinica privata, utilizza il suo software gestionale creando documenti sanitari sui pazienti non collegati tra loro, difficilmente aggiornabili proprio per la disomogeneità degli strumenti utilizzati. Se invece il sistema sanitario procedesse verso una standardizzazione delle applicazioni software in maniera centralizzata, i dati sanitari di ogni paziente sarebbero completi e precisi, indipendentemente da quale sia la struttura sanitaria di provenienza. Ogni medico potrebbe sapere, ad esempio, con assoluta precisione quali sono i farmaci in uso del paziente, le allergie, i precedenti clinici, garantendo una prestazione sanitaria più efficiente e di qualità.

In un Paese come l'Italia dove l'efficienza del servizio sanitario varia in base alla regione o alla provincia di residenza, con ospedali altamente tecnologici e altri meno sviluppati sotto questo aspetto, è indispensabile riuscire a sviluppare una sanità digitale che sia il più efficace ed efficiente possibile.

L'implementazione della registrazione di tutti i dati sanitari, dalla semplice "ricetta medica" alla più complicata cartella clinica, su piattaforma centralizzata migliorerebbe di molto la vita di medici e operatori sanitari che potrebbero accedere

a tutte le informazioni riguardanti il loro paziente, aggiornate al minuto, e avere a disposizione tutti i dati necessari per svolgere al meglio il loro lavoro. [17]

Implementando tale tipo di soluzione, risulterebbe necessario una forte attenzione per gli aspetti di sicurezza e tracciabilità dei dati. La tecnologia blockchain potrebbe entrare in gioco proprio per far fronte a tali necessità.

In definitiva, quindi, le blockchain applicate al settore della sanità permettono a ospedali, contribuenti e altre strutture sanitarie di condividere l'accesso ai loro network senza compromettere la sicurezza e l'integrità dei dati.

4.1.3 Quorum

Quorum è una piattaforma blockchain privata creata come soft fork[1] di Ethereum. Mantenendo gran parte delle funzionalità di Ethereum, Quorum ha aggiunto delle funzionalità utili alla gestione di una blockchain di tipo consortium.

4.1.3.1 Solidity

Essendo basato su Ethereum, anche Quorum, utilizza Solidity per lo sviluppo di smart contract. Solidity è un linguaggio di programmazione progettato specificamente per scrivere smart contract sulla blockchain di Ethereum.

4.1.3.2 GoQuorum Wizard

Per lo sviluppo del prototipo della rete blockchain utilizzata nella piattaforma di telemedicina è stato utilizzato GoQuorum Wizard. Quest'ultimo è uno strumento da riga di comando che consente di configurare una rete di nodi di tipo consortium in locale sul proprio dispositivo. Esso si basa su due elementi fondamentali:

- essendo scritto in Javascript è progettato per essere eseguito come modulo NPM globale dalla riga di comando, è necessario, quindi il modulo Node.js (approfondito nella sezione 4.3);
- per generare le risorse e i volumi su cui si troveranno i nodi, al fine di creare la rete, viene utilizzato Docker (approfondito nella sezione 4.2).

[1] Con il termine fork si fa riferimento al processo che, durante la fase di sviluppo di un software, porta alla creazione di un programma completamente nuovo che viene sviluppato a partire da un codice esistente. Nel dettaglio, è un aggiornamento del software compatibile con le versioni precedenti del sistema di partenza.

4.2 Docker

Docker è una piattaforma open source che consente agli sviluppatori di creare, implementare, eseguire, aggiornare e gestire container: componenti eseguibili standardizzati che combinano il codice sorgente delle applicazioni con le librerie e le dipendenze del sistema operativo necessarie per eseguire tale codice in qualsiasi ambiente. La tecnologia dei container offre tutte le funzionalità e i vantaggi delle macchine virtuali(VM), tra cui isolamento delle applicazioni, scalabilità, oltre ad vere un costo notevolmente inferiore rispetto alle VM: i container non comportano il payload di un'intera istanza del sistema operativo, includono solo i processi e le dipendenze necessari per eseguire il codice. [18]

4.3 Node.js

Node.js è un ambiente integrato di sviluppo ed esecuzione di codice Javascript, che permette di utilizzare tale linguaggio anche al di fuori di un browser lato client.

Node può essere impiegato in molti ambiti tra cui la realizzazione di componenti server da far interagire con app mobile e web app, IoT (Internet of Things) e domotica, applicazioni desktop, gestione del sistema operativo.

Tale framework all'interno del progetto di tesi è stato utilizzato per creare l'app di interfaccia da riga di comando (simulazione di un applicativo desktop) e il middleware di interfacciamento verso il model della blockchain.

4.3.1 NPM

NPM è il sistema di pacchetti di Node.js. Esso è dotato di una utility da riga di comando pronta all'uso, attraverso la quale è possibile gestire le versioni dei pacchetti utilizzati nel progetto, rivedere le dipendenze e anche impostare script personalizzati. Quando una nuova applicazione viene inizializzata, NPM crea automaticamente un file *package.json* che contiene tutti i pacchetti utilizzati nel progetto con le relative versioni. Basterà condividere questo file per indicare quali sono le dipendenze, con le relative versioni, necessarie all'esecuzione del sistema che si sta sviluppando. [19]

4.4 GitLab

GitLab è uno strumento web basato su due cardini fondamentali: controllo delle versioni e Git, sistema distribuito che permette il controllo delle versioni a tutti gli

utenti associati a un dato repository. Esso rappresenta una versione a pagamento del più comune GitHub; nel dettaglio, ogni utente ha un suo spazio utilizzabile per lo sviluppo delle proprie applicazioni. Per il progetto in esame in questo documento è stato utilizzato lo spazio Gitlab associato all'utenza di NBS s.r.l., accessibile tramite intranet aziendale.

L'uso di questo tool permette a più utenti di lavorare in modo coordinato sulla stessa base di codice, pur sviluppando in modo indipendente. GitHub aiuta, quindi, ad archiviare e gestire il proprio codice e a tracciare e controllare le modifiche. Per capire esattamente cos'è GitHub, è opportuno richiamare i due principi a esso collegati, già introdotti in precedenza:

- *Controllo versioni*: Il controllo delle versioni permette di tracciare e gestire le modifiche al codice di un progetto software con facilità. Man mano che un progetto software cresce, il controllo delle versioni diventa essenziale. Nel dettaglio permette di lavorare in sicurezza attraverso il branching (ramificazione) e il merging (fusione). Con il branching è possibile duplicare parte del codice sorgente (chiamato repository). Si procede, quindi, con l'apportare in modo sicuro modifiche a quella parte del codice senza influenzare il resto del progetto. Una volta che la sezione di codice modificata funziona correttamente, la si può fondere nel codice sorgente principale e rendere le modifiche ufficiali. Tutte queste modifiche vengono poi monitorate e, se necessario, possono essere riportate a una versione precedente.
- *Git*: Git è un sistema di controllo versioni distribuito, il che significa che l'intero codice e la relativa cronologia sono disponibili a ogni membro del team di sviluppo che ha accesso al repository, il che permette di creare facilmente ramificazioni e fusioni. [20]

All'interno del progetto descritto in questo elaborato di tesi, questo strumento è stato utilizzato per mantenere un versioning dell'intero codice alla base di tutte le componenti che costituiscono il prototipo della piattaforma di telemedicina sviluppato.

4.5 HAPI FHIR

Per la realizzazione del server FHIR è stata utilizzata la libreria HAPI FHIR; essa è un'implementazione della specifica HL7 FHIR (approfondita nella sezione 2.2) per

Java (linguaggio di programmazione descritto nella sezione 4.5.4).

Nello specifico, il codice di implementazione del server fa riferimento alla versione della libreria che implementa la release R4 dello standard HL7 FHIR.

4.5.1 Tipologie di server

HAPI FHIR fornisce diversi framework per la creazione di server FHIR:

- *HAPI FHIR Simple Plain Server*: fornisce un'implementazione di un server FHIR rispetto a un database arbitrario fornito dall'utente.
- *Server HAPI FHIR JPA*: fornisce un'implementazione completa di un server FHIR rispetto a un database relazionale. A differenza del Plain Server, il server JPA fornisce il proprio schema di database e gestisce tutta la logica di archiviazione e recupero senza che sia richiesta alcuna codifica da parte dell'utente che sta implementando il server. [21]

4.5.1.1 Simple Plain Server - Facade

Il framework utilizzato per l'implementazione del server nel progetto di tesi definisce un Simple Plain Server.

In questa modalità, il codice che gestisce l'archiviazione delle risorse e la logica di recupero delle stesse, fornito dalla libreria, si occupa di:

- Elaborazione HTTP;
- Semantica FHIR REST.

Questo modulo definisce un meccanismo in grado di posizionare un livello FHIR comune su una serie di origini dati esistenti. Questo fa si che le strutture sanitarie, già in possesso di database popolati con i dati dei loro assistiti, non debbano migrare la loro base di dati in un nuovo schema imposto dall'infrastruttura.

Una facade, infatti, è costituita da un oggetto che permette l'accesso a sottosistemi che espongono interfacce complesse e molto diverse tra loro, attraverso un'unica interfaccia nettamente più semplice.

4.5.2 Risorse e provider

L'elemento costitutivo di base in FHIR è una risorsa. Tutto il contenuto scambiabile è definito attraverso tale elemento; inoltre, tutte le tipologie di risorse condividono il seguente insieme di caratteristiche:

- Un modo comune per definirle e rappresentarle;
- Un insieme comune di metadati;
- Una parte leggibile dall'uomo.

Le risorse hanno una vasta gamma di usi, dal puro contenuto clinico come piani di cura e referti diagnostici alla pura infrastruttura come

- Message Header Anlisys: analisi dell'intestazione di un messaggio, essa racchiude dei metadati riguardanti il messaggio stesso;
- Capability Statements: la dichiarazione delle capacità è una parte fondamentale del quadro generale di conformità in FHIR. Viene utilizzata come dichiarazione delle caratteristiche del software o di un insieme di regole a cui allineare l'applicativo

[22]

Esse sono rappresentate attraverso dei documenti che possono essere di due formati: JSON e XML. Nello specifico all'interno del prototipo sono state utilizzate risorse codificate in JSON.

Vi è la possibilità di utilizzare le risorse già mappate da HL7 oppure di definire, seguendo le linee guida imposte dalla libreria, delle risorse personalizzate sulla base delle esigenze riscontrate nell'applicazione.

Nel progetto sono state utilizzate due risorse definite dallo standard:

- Paziente;
- Osservazione.

Per garantire una forte stabilità del software sviluppato, le risorse appena indicate sono state selezionate tra la lista di risorse che hanno superato il processo di approvazione per gli stardard ANSI/ISA-95[2]. La rappresentazione delle risorse all'interno della libreria, quindi, è da considerarsi come stabile, rappresentando così un punto fermo nelle regole di compatibilità tra versioni FHIR. Sebbene siano possibili cambiamenti a queste sezioni di libreria, essi dovrebbero essere poco frequenti e strettamente limitati.

In generale, in HAPI FHIR, ogni tipo di risorsa definito dallo standard, creato da HL7, ha una classe corrispondente, che contiene un numero di getter e setter per le proprietà di base a essa associate.

[2]ANSI/ISA-95 è uno standard internazionale dall'International Society of Automation per lo sviluppo di un'interfaccia automatizzata tra sistemi aziendali e sistemi di controllo.

Inoltre, a ogni risorsa è associato un Provider: classe che ne implementa le operazioni principali. Tra queste operazioni si hanno:

- Creazione;
- Lettura puntuale di una risorsa tramite ID;
- Ricerca di una risorsa, con la possibilità di implementare diversi criteri di ricerca basati su determinati campi della risorsa stessa;
- Aggiornamento e gestione delle versioni;
- Cancellazione;
- Ottenimento della classe di risorsa gestita dal provider. [21]

Anche grazie all'utilizzo dei provider, non si ha direttamente accesso alla stringa che costituisce la chiamata al server o alla composizione della stringa di risposta. Entrambe devono essere gestite interfacciandosi a esse con gli appositi metodi messi a disposizione dalla classe della risorsa e dal provider.

4.5.3 Interceptor

La versione 3.8.0 di HAPI FHIR ha introdotto un nuovo framework di intercettazione utilizzato nell'intera libreria.

Le classi di Interceptor possono *collegarsi* a vari punti della catena di elaborazione sia lato client che lato server. Queste classi si basano su metodi *Hook*; essi sono dei metodi singoli (riconoscibili in quanto marcati con l'annotazione @Hook) che vengono richiamati in risposta a un'azione specifica che si svolge nel framework HAPI FHIR.

Grazie a questa tipologia di classi è possibile gestire tutte le operazioni che non riguardano direttamente le risorse, come, ad esempio, i meccanismi di autenticazione e autorizzazione verso il server.

I *pointcut* (punti di aggancio) a cui si possono applicare i metodi Hook all'interno di un Client sono riportati in figura 4, mentre quelli relativi a un generico Server sono mostrati in figura 5. [21]

4.5.4 Java

Java è un linguaggio multi-piattaforma e orientato agli oggetti. Si tratta di un linguaggio di programmazione veloce, sicuro e affidabile utilizzato in diversi ambiti

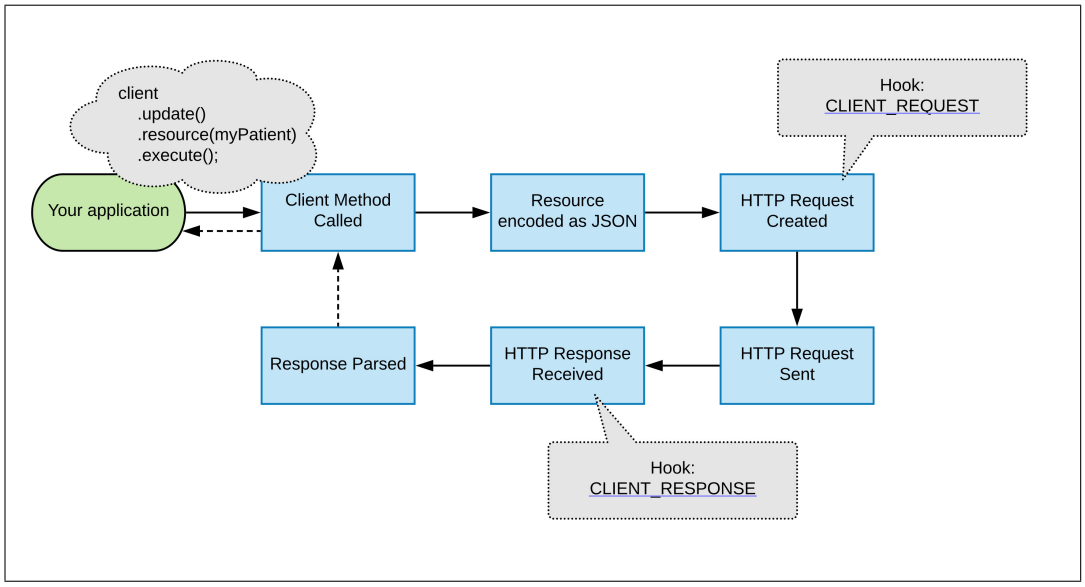

Figura 4: Pointcut in un client HAPI FHIR

come le applicazioni per dispositivi mobili, i software aziendali, le applicazioni per big data o le tecnologie lato server.

4.5.4.1 Java Virtual Machines (JVM) e Compilatore Just-in-Time

Per poter illustrare il processo di traduzione implementato da Java, si introducono di seguito i traduttori. Essi sono software che traducono un programma scritto in un dato linguaggio di programmazione in codice eseguibile da un calcolatore elettronico; è possibile caratterizzare i traduttori in due categorie principali:

- *Compilatori*: Realizzano solo la fase di traduzione del codice sorgente in codice eseguibile (o meglio codice oggetto); l'input dell'esecutore a valle del compilatore è l'intero codice.
- *Interpreti*: Costituiscono un sistema di traduzione che oltre a tradurre esegue anche il programma; ogni istruzione viene prima tradotta e poi eseguita.

Il programma Java combina entrambi i metodi indicati sopra mediante l'uso di una macchina virtuale Java. Qualsiasi file Java viene prima compilato in bytecode. Il bytecode Java può essere eseguito solo nella JVM. Essa interpreta quindi il bytecode per eseguirlo sulla piattaforma hardware sottostante. Pertanto, se l'applicazione è in esecuzione su un computer Windows, JVM la interpreterà per tale sistema operativo. Ma se è in esecuzione su una piattaforma open-source come Linux, JVM la interpreta per questo sistema operativo. [23]

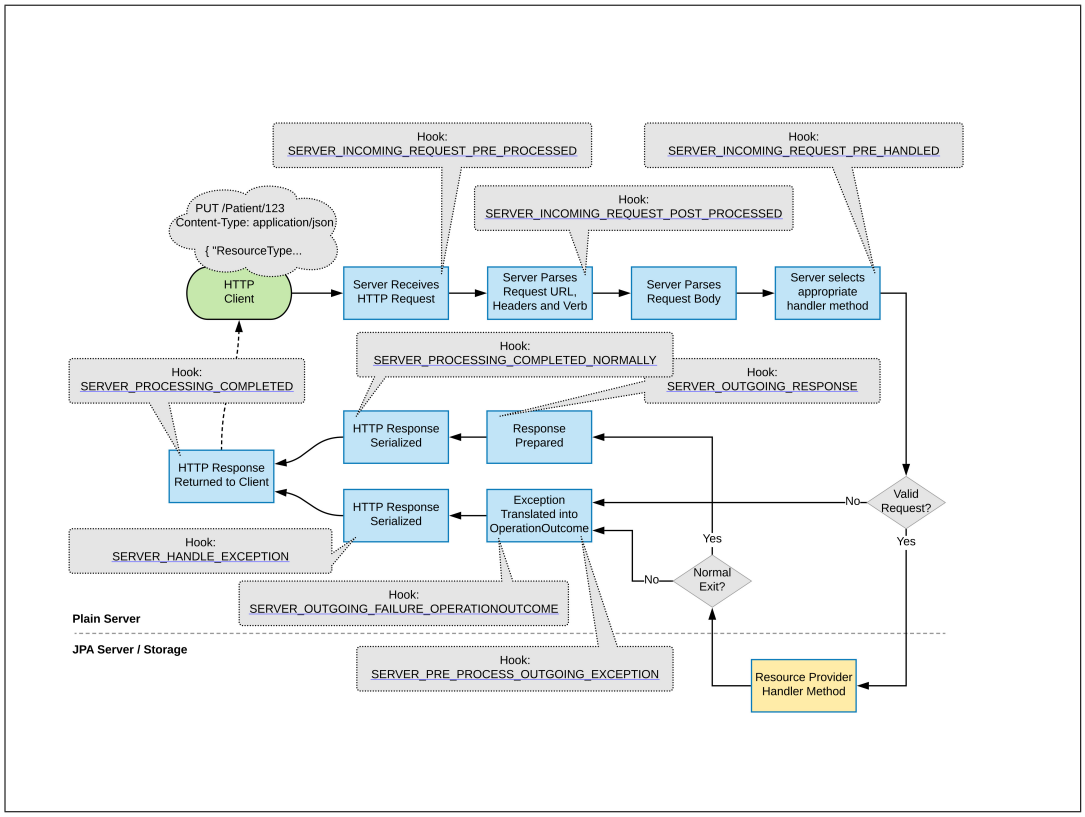

Figura 5: Pointcut in un server HAPI FHIR

Inoltre, l'interprete fa riferimento a un compilatore Just-in-Time, ogni volta che si ha un nuovo metodo il compilatore compila la relativa sezione di codice e si riattiva il codice precompilato ogni qual volta risulti necessario senza dover attuare il processo di compilazione nuovamente.

4.6 Ide

Un IDE (ambiente di sviluppo integrato), è un software progettato per la realizzazione di applicazioni che unisce strumenti di sviluppo comuni in un'unica interfaccia utente. È costituito da diverse componenti:

- Editor del codice sorgente: un editor di testo che agevola la scrittura di codice software grazie a utili funzionalità come il completamento automatico specifico del linguaggio, l'evidenziazione della sintassi con suggerimenti visivi e l'individuazione di errori durante la scrittura.
- Automazione della build locale.

- Debugger: un programma per testare altri programmi in grado di visualizzare graficamente la posizione di un bug nel codice originale. [24]

Durante l'implementazione del prototipo della piattaforma di telemedicina sono stati utilizzati i seguenti IDE: Visual Studio Code (sezione 4.6.1) e Android Studio (sezione 4.6.2).

4.6.1 Visual Studio Code

Visual Studio Code (VSC) è un editor cross-platform compatibile con Windows, Linux e macOS che permette di evidenziare la sintassi di ciascun linguaggio di programmazione, integra il supporto per il debugging, controllo Git, IntelliSense (il completamento automatico delle istruzioni), possibilità di mantenere aperti più file affiancandone il contenuto in più schede.

Punto di forza di VSC sono le estensioni grazie alle quali è possibile ampliare notevolmente le funzionalità del programma; tra quelle utilizzate nel progetto, oltre a quelle necessarie per la scrittura nei vari linguaggi di programmazione utilizzati, vi è *Docker*, estensione che implementa la gestione dei volumi Docker, e *Git*, estensione che facilita la gestione delle versioni. [25]

L'ide descritto in questa sezione è stato utilizzato per lo sviluppo della blockchain (sezione 5.2), del server FHIR (sezione 5.3) e dell'app di interfaccia (sezione 5.6).

4.6.2 Android Studio

Android Studio è un ambiente di sviluppo integrato adibito per la creazione di applicazioni Android disponibile su licenza Apache 2.0, basata su linguaggio Java. Il programma è del tutto compatibile con dispositivi Windows, Linux e macOS. [26]

Tale ide è stato utilizzato per lo sviluppo dell'applicativo mobile descritta nella sezione 5.7.

4.7 Ionic

Per lo sviluppo delle due applicazioni, descritte nella sezione 5.7, è stato utilizzato Ionic: un framework HTML5 open source, usato per scrivere applicazioni mobile ibride con tecnologie web come HTML, JavaScript, CSS e SASS. Con Ionic si possono creare applicazioni web che possono essere riportate su ogni piattaforma o dispositivo a partire da un'unica base di codice.

Questo framework offre una vasta libreria front-end di componenti per l'interfaccia grafica, ottimizzati per mobile e compatibili con qualsiasi framework JavaScript, come Angular, React e Vue. Nello svilupo degli applicativi, presentati nella sezione 5.7, è stato utilizzato il framework Angular (approfondito nella sezione 4.7.2).

Ionic presenta inoltre una interfaccia da riga di comando (Ionic CLI) funzionale all'attività di creazione, testing e distribuzione delle app.

La versione di Ionic utilizzata, sfrutta come sistema di runtime Capacitor (approfondito nella sezione 4.7.1). [27]

4.7.1 Capacitor

Capacitor è un runtime di app multipiattaforma che consente di creare web app eseguite in modo nativo su iOS, Android, Electron e sul web, usando la tecnologia Progressive Web App (PWA).

Il runtime Capacitor supporta Swift e Objective-C su iOS, e Java e Kotlin su Android, e viene fornito senza alcun set specifico di controlli per l'interfaccia utente. Condividendo i vantaggi multipiattaforma con Cordova[3], Capacitor possiede però un approccio più moderno allo sviluppo di app, e sfrutta le più recenti API web. [28]

4.7.2 Angular

Angular è uno dei più utilizzati framework javascript (sviluppato da Google) per la realizzazione di interfacce utente.

Tra le altre funzionalità, esso implementa il *two-way data binding*: associazione bidirezionale utilizzata come *listener* di eventi e per aggiornare simultaneamente i valori tra i componenti padre e figlio. Nel dettaglio, Angular crea per ogni elemento dell'interfaccia che utilizza la direttiva *ng-model*, un *watch*, ossia una funzione che "osserva" i cambiamenti della variabile su cui è definita.

Attraverso il processo *Digest Loop*, infine, vengono esaminati tutti i watch con lo scopo di individuare dei cambiamenti dall'ultima esecuzione ed eventualmente aggiornare il DOM[4]. [29]

[3]Cordova è un framework di sviluppo mobile per la programmazione multipiattaforma.

[4]DOM: Document Object Model, rappresentazione di una pagina HTML e interfaccia di programmazione che permette di relazionarsi in maniera univoca con il documento.

4.8 Express

Express.js è un framework open-source Node.js per la programmazione di applicazioni web e mobile. Esso è stato utilizzato nella creazione del web service di interfacciamento verso la blockchain da parte delle varie istanze dell'app mobile.

Il framework Express consente di creare API di routing e di impostare middleware per rispondere alle richieste HTTP in modo semplice e lineare. [30]

4.9 DBeaver

DBeaver è un software che funge da strumento per la gestione e il monitoraggio di database destinato a sviluppatori e amministratori di database.

In questo progetto DBeaver è stato utilizzato per gestire connessioni multiple alle varie istanze di database create per il prototipo di rete di telemedicina.

DBeaver supporta tutti i database più popolari; tra essi vi è anche PostgreSQL, utilizzato per l'implementazione dei database utilizzati nel progetto e approfondito nella sezione 4.9.1. [31]

4.9.1 PostgreSQL

PostgreSQL, anche noto come Postgres e utilizzato per implementare le istanze di database utilizzate nel progetto di tesi, è un sistema di database relazionale a oggetti (ORDBMS), open source e gratuito. Le principali caratteristiche di Postgres sono affidabilità, integrità dei dati, funzionalità ed estensibilità.

Per l'interrogazione e l'interazione con i dati, Postgres utilizza un linguaggio sottoinsieme di SQL. [32]

4.10 Firebase

Firebase è una piattaforma serverless per lo sviluppo di applicazioni mobili e web. Supportata da Google, essa fornire una suite di strumenti per scrivere, analizzare e mantenere applicazioni cross-platform; offrendo funzionalità come analisi, database, messaggistica e segnalazione di arresti anomali per la gestione di applicazioni web, iOS e Android. [33]

All'interno del progetto in esame, questo strumento è stato utilizzato per gestire le notifiche da inviare all'applicativo mobile dedicato al livello di utenza dei professionisti sanitari.

Capitolo 5

Sviluppo del prototipo

Al fine di soddisfare i requisiti descritti nel capitolo 3, è stato progettato e implementato un prototipo di una piattaforma di telemedicina distribuita che si basa su due tecnologie: una rete blockchain di tipo consortium e un'istanza di server FHIR da allocare per ogni struttura sanitaria facente parte della rete.

Nel capitolo corrente viene riportata la progettazione dell'applicativo nella sua interezza, come esso opera, le diverse modalità di interfacciamento e alcune porzioni di codice ritenute significative per la soluzione dei problemi affrontati.

Si precede quindi con una descrizione del prototipo a un livello di astrazione tale per cui si possa comprendere il workflow complessivo, per poi scendere nel dettaglio descrivendo gli elementi costituenti dell'intera architettura.

5.1 Architettura

La piattaforma di telemedicina presentata in questo documento di tesi, è costituita da una rete di nodi, ognuno dei quali è composto da quattro componenti cardine:

- Nodo blockchain;
- Server FHIR;
- Database di tipo relazionale;
- App di interfaccia da riga di comando (l'interfaccia viene indicata in blu nella figura 6).

L'intera trattazione riportata in questa sezione è da riferirsi all'architettura osservabile in figura 6, all'interno della quale sono visibili alcune delle interazioni fondamentali (riguardanti tutti i nodi e non solo quelli a cui gli archi fanno capo) come:

- *Token*: Rilascio di token di autorizzazione da parte del nodo della blockchain;
- *Richiesta autorizzata con token*: Richiesta inviata al server FHIR ponendo nell'header della stessa il token di autorizzazione di cui sopra;
- *Recupero dati*: Ottenimento delle risorse dal database della struttura sanitaria attraverso un'opportuna query SQL.

Tali operazioni sono approfondite di seguito.

La rete Blockchain Consortium funge da "cabina di regia" dell'intera piattaforma; essa, difatti, è necessaria per:

- Indicizzare gli eventi di telemedicina, che si verificano all'interno di una delle strutture sanitarie connesse, per i soli pazienti che hanno rilasciato il consenso al trattamento dei propri dati; nel momento in cui il consenso dovesse venire rimosso semplicemente si smetterà di indicizzare gli eventi dell'assistito in questione all'interno della rete.
- Tenere traccia di tutte le strutture sanitarie, private e non, associate alla piattaforma; questa scelta progettuale è stata presa al fine di rendere il più immediato possibile l'inserimento di una nuova struttura nel sistema. Non sarà quindi necessario notificare l'ingresso alle n strutture già presenti ma basterà aggiornare il registro della blockchain, a cui ciascun nodo ha accesso.
- Rilasciare token di autorizzazione per la richiesta di dati sanitari verso i server

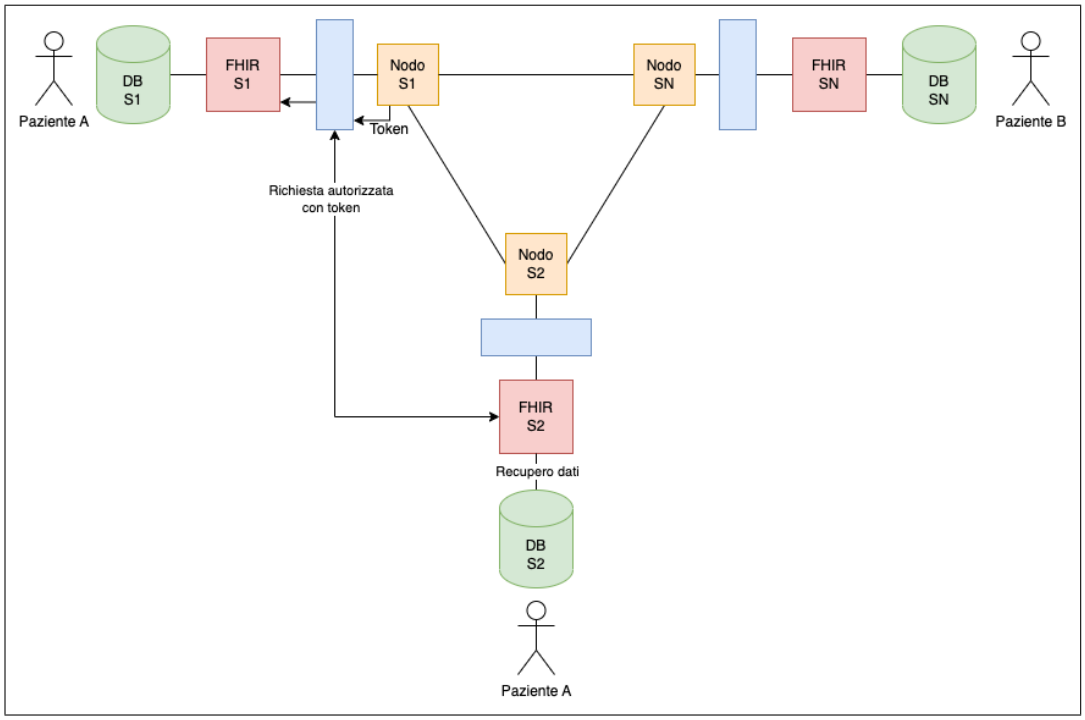

Figura 6: Architettura del prototipo della piattaforma di telemedicina

FHIR delle strutture sanitarie che li detengono. Nell'indicizzazione degli eventi, difatti, è riportata anche l'ubicazione del dato sanitario associato all'evento stesso.

Al netto della funzione di hash applicata ai dati dalla blockchain, all'interno del registro i dati relativi ai pazienti non sono salvati in chiaro (come ad esempio con il codice fiscale dell'assistito) ma con il relativo identificativo interno della struttura a cui il paziente si è rivolto e un codice univoco per la struttura stessa. Ciò per avere un livello di sicurezza maggiore, difatti, se anche qualcuno riuscisse ad accedere alla rete, riuscendo quindi a superare l'autorità centrale che ne detiene i privilegi di gestione, non riuscirebbe a comprendere le informazioni contenute nei suoi registri. La chiave di lettura, ossia l'associazione tra codice fiscale del paziente e l'identificativo interno viene inviata dalla struttura sanitaria di riferimento ai soli professionisti autorizzati a consultare i dati dell'assistito; questo invio avviene off-chain attraverso una richiesta al server FHIR.

Le varie istanze di server FHIR sono necessarie per la gestione e l'invio dei dati sanitari degli assistiti. Ognuna di esse si basa su un database relazionale all'interno del quale vengono salvati i soli dati riguardanti eventi di natura medica. L'intero meccanismo di gestione dei token rimane all'interno del server e non usufruisce del

database per il salvataggio delle autorizzazioni.

Nel prototipo implementato, per semplicità si è scelto di creare un'istanza di database per ogni struttura ma, verosimilmente, in un contesto reale lo si può considerare come una vista[1] sulla base di dati già in possesso della struttura sanitaria che vuole aderire alla piattaforma di telemedicina. Ciò perché si ritiene di fondamentale importanza la capacità del sistema di essere scalabile rispetto alle soluzioni già esistenti, senza dover stravolgere l'organizzazione aziendale delle strutture sanitarie.

L'applicativo da riga di comando è necessario per interfacciare gli utenti alle tre componenti riportate sopra; grazie a delle chiamate verso il server FHIR e gli smart contract nella blockchain, permette di coordinare le operazioni sull'intero sistema.

Per poter gestire al meglio le funzionalità riguardanti il Telemonitoraggio, sono state sviluppate due applicazioni mobile: una per il livello di utenza degli assisti (la cui architettura è visibile in figura 25) e una per i medici (si faccia riferimento alla figura 33 per osservarne l'architettura).

I pazienti potranno inserire nell'interfaccia grafica dell'applicazione delle informazioni riguardanti il loro percorso di telemonitoraggio; questi dati, uniti a quelli ottenuti dall'app attraverso l'interfacciamento con il servizio restAPI di Google Fit e con il resto dei sensori a disposizione dell'utente, vengono inviati al server FHIR della struttura sanitaria di riferimento andando a popolare l'evento di telemonitoraggio associato al paziente.

L'utenza dei professionisti sanitari potrà tenere monitorati i propri pazienti andando a indicare per ognuno di essi un livello di priorità. Sulla base del livello espresso verranno inviate al medico delle notifiche ogni qual volta viene registrata nel server FHIR una osservazione con un valore fuori soglia.

Come per l'interfaccia da riga di comando, anche le app mobile avranno la necessità di comunicare con la blockchain. Per evitare però che svariate istanze di applicazioni si interfaccino direttamente con la rete, è stato creato un webservice che espone delle restAPI; esse possono essere richiamate degli applicativi per effettuare delle operazioni verso gli smart contract salvati nei nodi della blockchain.

Nelle sezioni che seguono si scende nel dettaglio implementativo delle varie componenti appena descritte.

[1] Le viste sono comunemente utilizzate per mostrare i dati di un database con una struttura diversa da quella che hanno effettivamente all'interno della base dati.

5.2 Blockchain

In questa sezione viene approfondita la blockchain sviluppata per tale progetto di tesi con lo scopo di avere un registro di indicizzazione condiviso, sicuro, accessibile a cui tutte le strutture sanitarie possono far capo, oltre che per la gestione dell'autorizzazione all'accesso delle risorse FHIR.

5.2.1 Smart Contract

Nel dettaglio, sono stati implementati tre smart contract che gestiscono rispettivamente i dati relativi alle strutture sanitarie, le autorizzazioni e gli eventi da indicizzare.

Tutti i contract ereditano da *ERC721*, classe che implementa lo standard ERC-721 per i token non fungibili.

5.2.1.1 Smart Contract per la gestione delle strutture sanitarie

La struttura del token gestito in questo contratto è riportata nella tabella 1.

Tabella 1: Token MedicalStruct

Nome Attributo	**Tipo**	**Descrizione**
key	string	Nome della struttura sanitaria (univoco)
chainAddress	string	Indirizzo del nodo della blockchain della struttura sanitaria
serverAddress	string	Endpoint del server FHIR della struttura sanitaria

I token vengono indicizzati sulla base dei vari attributi. La scelta delle strutture in cui salvare i dati è stata presa in funzione delle necessità riscontrate durante l'implementazione dei metodi.

- *getMedicalStructByKey*: token indicizzato sulla base del valore di chiave (nome della struttura).
- *getMedicalStructByAddress*: token indicizzato sulla base dell'indirizzo del nodo blockchain.

- *existMedicalStruct*: struttura che associa a ciascuna chiave registrata un valore booleano, utile a verificare l'esistenza della chiave e l'eventuale deprecazione del token.

5.2.1.2 Smart Contract per la gestione delle autorizzazioni

La struttura del token gestito in questo contratto è riportata nella tabella 2.

Tabella 2: Token Auth

Nome Attributo	**Tipo**	**Descrizione**
key	string	Chiave del token composta dai campi patientStruct, keyPatient, doctorCode
patientStruct	address	Indirizzo del nodo della blockchain della struttura sanitaria a cui si è rivolto il paziente
keyPatient	string	Identificativo univoco del paziente all'interno della struttura sanitaria
doctorCode	string	Codice fiscale del dottore protagonista dell'autorizzazione
accepted	bool	Valore booleano che indica se la collaborazione è stata accetta da parte del medico

I mapping per l'indicizzazione dei token sono:

- *getAuthByKey*: Indicizza i token per la rispettiva chiave;
- *getAuthsByDoctorCode*: Struttura dati che indicizza una lista di autorizzazioni per il codice fiscale del medico a cui le autorizzazioni si riferiscono;
- *existAuth*: Valore booleano che, data la chiave, indica se esiste l'autorizzazione (in quanto le autorizzazioni possono essere rimosse);
- *existAuthForDoc*: Valore booleano che indica se esistono autorizzazioni per un dato medico.

5.2.1.3 Smart Contract per la gestione degli eventi

La struttura del token gestito in questo contratto è riportata nella tabella 3.

Tabella 3: Token Event

Nome Attributo	**Tipo**	**Descrizione**
key	string	Identificativo dell'evento definito a seguito dell'inserimento dello stesso nel server FHIR
structPatientKey	string	Chiave dell'assistito composta dal codice del paziente e dall'indirizzo della struttura sanitaria di riferimento
medicalStruct	address	Indirizzo del nodo della blockchain della struttura sanitaria a cui si è rivolto il paziente
patientCode	string	Identificativo univoco del paziente all'interno della struttura sanitaria
eventCode	string	Codice che indica la tipologia di evento
date	string	Data dell'evento

I mapping per l'indicizzazione dei token sono:

- *getEventByKey*: Indicizza i token per la rispettiva chiave;
- *getEventsByPatientCode*: Struttura dati che indicizza una lista di eventi per l'identificativo del paziente a cui sono associati;
- *existEvent*: Valore booleano che, data la chiave, indica se esiste l'evento;
- *existEventsPatient*: Valore booleano che indica se esistono eventi registrati per un dato assistito.

5.2.2 Model

Al fine di astrarre i metodi dei tre smart contract appena presentati, è stato creato un model che funge da interfaccia verso tali metodi.

Quest'ultimo viene invocato da tutte le componenti che devono interfacciarsi con la blockchain: il webservice (approfondito nella sezione 5.5) e l'app di interfaccia da riga di comando (descritta nella sezione 5.6).

Risulta di fondamentale importanza che il model invochi l'istanza corretta di contratto: quella che detiene le informazioni della struttura sanitaria a cui si vuole accedere. Per garantire ciò è stata utilizzata una soluzione che implementa il pattern Singleton, pattern utilizzato al fine di impedire che da una classe possa essere creato più di un oggetto.

Si osservi il codice riportato in figura 7 per comprendere la trattazione che segue. A partire dall'abi e dall'indirizzo di ogni contratto, viene generata un'istanza per ogni nodo della blockchain. Il termine abi sta per *Abstract Binary Interface*, ossia la rappresentazione binaria dell'interfaccia di uno smart contract, all'interno della quale vengono elencate e descritte le variabili e i diversi metodi del contract.

Invocando la funzione *getInstanceContract* e passando come parametri il nome del contratto e l'indirizzo del nodo della struttura sanitaria si ottiene l'istanza di contratto desiderata.

5.3 Server FHIR

In questa sezione è descritto il server FHIR; si analizzano, di seguito, la gestione delle risorse, delle richieste ed, infine, gli *interceptor* creati per la gestione dell'autenticazione mediante token e del sistema di notifiche.

5.3.1 Risorse gestite dal server

Come già indicato, al fine di gestire la comunicazione di dati sanitari all'interno della piattaforma di telemedicina, è stato implementato un server FHIR. Tali dati sono gestiti classificandoli in cinque risorse FHIR, descritte nei paragrafi che seguono.

5.3.1.1 Paziente

Classe standard dalla libreria, necessaria a mappare le informazioni relative ai singoli pazienti. NElla tabella 4 sono riportati i soli attributi della classe utilizzati all'interno del progetto.

```
const abi_cEvent = compiler.compile("../Contracts/Event/ContractEvent.sol")[0];  // PATH RELATIVI ALLA DIRECTORY DI LANCIO
const abi_cAuth = compiler.compile("../Contracts/Auth/ContractAuth.sol")[0];
const abi_cMedicalStruct = compiler.compile("../Contracts/MedicalStruct/ContractMedicalStruct.sol")[0];

const eventContractAddress_cf = JSON.parse(fs.readFileSync("../Contracts/Event/address.json"))[0];
const authContractAddress_cf = JSON.parse(fs.readFileSync("../Contracts/Auth/address.json"))[0];
const medicalStructContractAddress_cf = JSON.parse(fs.readFileSync("../Contracts/MedicalStruct/address.json"))[0];

const eventStruct1 = new web3.eth.Contract(abi_cEvent, eventContractAddress_cf);
const eventStruct2 = new web3_2.eth.Contract(abi_cEvent, eventContractAddress_cf);
const eventStruct3 = new web3_3.eth.Contract(abi_cEvent, eventContractAddress_cf);

const authStruct1 = new web3.eth.Contract(abi_cAuth, authContractAddress_cf);
const authStruct2 = new web3_2.eth.Contract(abi_cAuth, authContractAddress_cf);
const authStruct3 = new web3_3.eth.Contract(abi_cAuth, authContractAddress_cf);

const medicalStructStruct1 = new web3.eth.Contract(abi_cMedicalStruct, medicalStructContractAddress_cf);
const medicalStructStruct2 = new web3_2.eth.Contract(abi_cMedicalStruct, medicalStructContractAddress_cf);
const medicalStructStruct3 = new web3_3.eth.Contract(abi_cMedicalStruct, medicalStructContractAddress_cf);

const ENUMContractType = {
    Event: "Event",
    Auth: "Auth",
    MedicalStruct: "MedicalStruct"
}

function getInstanceEventContract(contractAddress, type) {
    var address = JSON.parse(fs.readFileSync("../Assets/wallets.json"));
    var index = contractAddress.concat(type);
    switch(index){
        case address[0].concat(ENUMContractType.Event): return eventStruct1;
        case address[1].concat(ENUMContractType.Event): return eventStruct2;
        case address[2].concat(ENUMContractType.Event): return eventStruct3;
        case address[0].concat(ENUMContractType.Auth): return authStruct1;
        case address[1].concat(ENUMContractType.Auth): return authStruct2;
        case address[2].concat(ENUMContractType.Auth): return authStruct3;
        case address[0].concat(ENUMContractType.MedicalStruct): return medicalStructStruct1;
        case address[1].concat(ENUMContractType.MedicalStruct): return medicalStructStruct2;
        case address[2].concat(ENUMContractType.MedicalStruct): return medicalStructStruct3;
    }
}
```

Figura 7: Codice Java per l'estrazione di un'istanza di contratto

5.3.1.2 Osservazione

Classe standard dalla libreria, utilizzata per tenere traccia di tutte le informazioni legate ai rilevamenti di natura sanitaria effettuati sui pazienti; ogni osservazione contiene al suo interno un campo che indica il paziente a cui si riferisce.

Con questa risorsa si tiene traccia sia degli eventi indicizzati nella blockchain, sia delle osservazioni ad essi associate.

Anche in questo caso, all'interno della tabella 5 sono visibili gli attributi della classe utilizzati nel prototipo.

5.3.1.3 Token di autorizzazione

Classe creata per questo progetto, utile alla gestione e alla verifica dei token di autorizzazione posti nell'header di ogni richiesta. Una descrizione degli attributi della classe è visibile nella tabella 6.

Tabella 4: Risorsa FHIR Patient

Nome	**Tipo**	**Descrizione**
id	IdType	Id generato applicando una funzione di hash al codice fiscale e al timestamp associato alla generazione della risorsa
name	HumanName	Contiene al suo interno le risorse Family e Given (rispettivamente cognome e nome)
identifier	Identifier	Codice fiscale
birthDay	DateType	Data di nascita
address	Address	Indirizzo di residenza
gender	AdministrativeGender	Genere selezionato tra una classe Enum dello standard HL7
version	Version	Contiente un valore incrementale che indica la versione degli aggiornamenti sulla risorsa e la data dell'ultima modifica

5.3.1.4 Informazioni di cross

Classe necessaria per effettuare un mapping delle informazioni di un dato paziente tra le varie strutture sanitarie che lo hanno in carica; anche questa classe e il relativo provider sono stati creati appositamente per questo progetto di tesi. Consultare la tabella 7 per analizzare gli attributi che compongono la classe.

5.3.1.5 Livello di priorità

Questa risorsa serve a gestire i livelli di priorità che ogni medico assegna a un determinato paziente rivoltosi alla struttura sanitaria a cui il server FHIR fa riferimento. Questa classe, come le due precedenti, è stata implementata specificatamente per questo progetto. Nella tabella 8 sono presenti gli attributi che descrivono tale risorsa.

Tabella 5: Risorsa FHIR Observation

Nome	**Tipo**	**Descrizione**
id	IdType	Id generato applicando una funzione di hash al codice fiscale e al timestamp associato alla generazione della risorsa
code	CodeableConcept	Contiente al suo intenro le risorse Family e Given (rispettivamente cognome e nome)
issued	InstantType	Codice fiscale
partOf	Reference	Data di nascita
subject	Reference	Indirizzo di residenza
reference Range	ObservationReference RangeComponent	Genere selezionato tra una classe Enum dello standard HL7
value Quantity	Quantity	Valore dell'osservazione a cui viene associata una specifica unità di misura
version	Integer	Valore incrementale che indica la versione degli aggiornamenti sulla risorsa

5.3.2 Richieste gestite dal server

Tutte le richieste gestite dal server sono riportate e descritte all'interno della tabella 9. Per tutte le richieste valgono le seguenti osservazioni:

- I parametri di tipo *IdentifierParam* vengono passati al server ponendoli direttamente nella rotta, essi non saranno specificati nella colonna "Parametri" ma sono indicati nella colonna "Rotta" di seguito al pattern "/:" (riferimento alla tabella 9);
- Nella colonna "Parametri" sono riportati solo i parametri definiti di seguito
  - *RequiredParam* parametro fornito in maniera standard, posto in seguito al carattere "?" nella rotta;
  - *ResourceParam* parametro fornito nel body indicante una risorsa FHIR (nella tabella 9 sono indicati con il nome della classe).

Tabella 6: Risorsa FHIR TokenAuth

Nome	**Tipo**	**Descrizione**
id	IdType	Identificativo univoco del token (attualmente rappresentato dal token stesso dato che non possono essere salavate due autorizzazioni identiche)
token	StringType	Contiene la stringa del token

Tabella 7: Risorsa FHIR Cross

Nome	**Tipo**	**Descrizione**
id	IdType	risultato di una funzione di hash applicata a idPatient e addressStruct
idPatient	StringType	Identificativo interno del paziente per la struttura corrispondente a addressStruct
address Struct	StringType	Indirizzo del nodo blockchain della struttura
fiscalCode	StringType	Codice fiscale del paziente

Tabella 9: Richieste gestite dal server FHIR

Metodo	**Rotta**	**Parametri**	**Descrizione**
POST	/Patient	Patient	Crea una nuova risorsa Patient
GET	/Patient/	Non previsti	Restituisce tutte le risorse Patient
GET	/Patient/:id	Non previsti	Restituisce la risorsa Patient corrispondente all'id fornito
GET	/Patient	cf	Restituisce tutte le risorse Patient con il codice fiscale fornito
GET	/Patient	family	Restituisce tutte le risorse Patient concordi con il nome fornito

Metodo	**Rotta**	**Parametri**	**Descrizione**
PUT	/Patient/:id	Patient	Esegue un update della risorsa Patient corrispondente all'id fornito dei soli campi presenti nel *body*
POST	/Observation	Observation	Crea una nuova risorsa Observation
GET	/Observation/:id	Non previsti	Restituisce la risorsa Observation corrispondente all'id fornito
GET	/Observation	subject	Restituisce le risorse Observation che fanno capo al paziente indicato in subject
GET	/Observation	part-of	Restituisce le risorse Observation che fanno capo all'evento indicato in part-of
GET	/Observation	part-of begin-date end-date	Restituisce le risorse Observation che fanno capo all'evento indicato in part-of registrate all'interno dell'intervallo temporale fornito
PUT	/Observation/:id	Observation	Esegue un update della risorsa Observation corrispondente all'id fornito dei soli campi presenti nel *body*
POST	/TokenAuth	TokenAuth	Crea una nuova risorsa TokenAuth
GET	/TokenAuth	Non previsti	Restituisce tutte le risorse TokenAuth
DELETE	/TokenAuth/:id	Non previsti	Elimina la risorsa TokenAuth corrispondente all'id fornito
POST	/Cross	Cross	Crea una nuova risorsa Cross
GET	/Cross	id-assistito address	Restituisce la risorsa Cross corrispondente ai parametri indicati
POST	/PriorityLevel	PriorityLevel	Crea una nuova risorsa PriorityLevel

Metodo	**Rotta**	**Parametri**	**Descrizione**
GET	/PriorityLevel	id-assistito codice-fiscale-dottore	Restituisce la risorsa PriorityLevel corrispondente ai parametri indicati
PUT	/PriorityLevel	PriorityLevel	Esegue un update della risorsa PriorityLevel corrispondente all'id presente nella risorsa fornita nel *body*

5.3.3 Interceptor

Come già riportato nella sezione 4.5.3, il metodo ottimale all'interno della libreria HAPI FHIR per implementare delle operazioni che non riguardino direttamente le risorse gestite dal server è quello di utilizzare degli interceptor. Nello specifico, nel progetto descritto in questo documento di tesi, sono state utilizzate tre classi di interceptor (di cui due sviluppate appositamente per questo caso di studio).

5.3.3.1 Gestione del sistema di sicurezza CORS

Per la gestione della policy CORS è stato utilizzato l'interceptor messo a disposizione dalle libreria adottata, andando a consentire l'acceso al servizio a tutte le sorgenti mediante il carattere speciale "*".

Chiaramente, consentire l'accesso ad ogni sorgente in un'applicazione reale è una scelta progettuale carente in termini di sicurezza; considerando però che l'intero prototipo della piattaforma di telemedicina è accessibile solo all'interno della rete aziendale di NBS s.r.l. tale criticità viene a cadere.

5.3.3.2 Meccanismo di autenticazione medinate token

Il sistema id telemedicina sviluppato basa l'autenticazione sul rilascio di token. Essi sono emessi dalla blockchain nel momento in cui un medico o un professionista sanitario richiede di accedere a delle risorse contenute in un'istanza del server FHIR.

Nel momento in cui la struttura sanitaria invia la richiesta di collaborazione ad un determinato professionista genera il token, che viene salvato e gestito da uno smartcontract apposito; oltre a ciò la struttura salva tale token anche all'interno del proprio server FHIR.

Tabella 8: Risorsa FHIR PriorityLevel

Nome	**Tipo**	**Descrizione**
id	IdType	risultato di una funzione di hash applicata a idPatient e fiscalCodeDoctor
idPatient	StringType	Identificativo interno del paziente per la struttura relativa all'istanza del server FHIR
fiscalCode Doctor	StringType	Codice fiscale del dottore
priorityLevel	StringType	Livello di priorità che il medico assegna al percorso di telemonitoraggio del paziente

In questo modo, nel momento in cui giunge al server una richiesta, attraverso l'interceptor si è in grado di controllare se il token presente nell'header della stessa sia valido o meno, confrontandolo con quelli forniti dalla struttura sanitaria di riferimento.

I token utilizzati possono essere di due tipologie:

- per interfacciarsi con il server della propria struttura basta fornire il token privato della struttura, non salvato nella blockchain;
- per inviare una richiesta autenticata ad un altro server è necessario fornire un token rilasciato dalla blockchain che contiene al suo interno le seguenti informazioni
  - indirizzo del nodo blockchain a cui fa riferimento il server;
  - identificativo interno del paziente;
  - codice fiscale del medico che ha inviato la richiesta.

Nel caso in cui il token inviato sia appartenente alla seconda tipolgia viene controllato che i dati inseriti nel token siano conformi alla richiesta inoltrata.

Il diagramma che illustra lo use case dell'interceptor è riportato in figura 8.

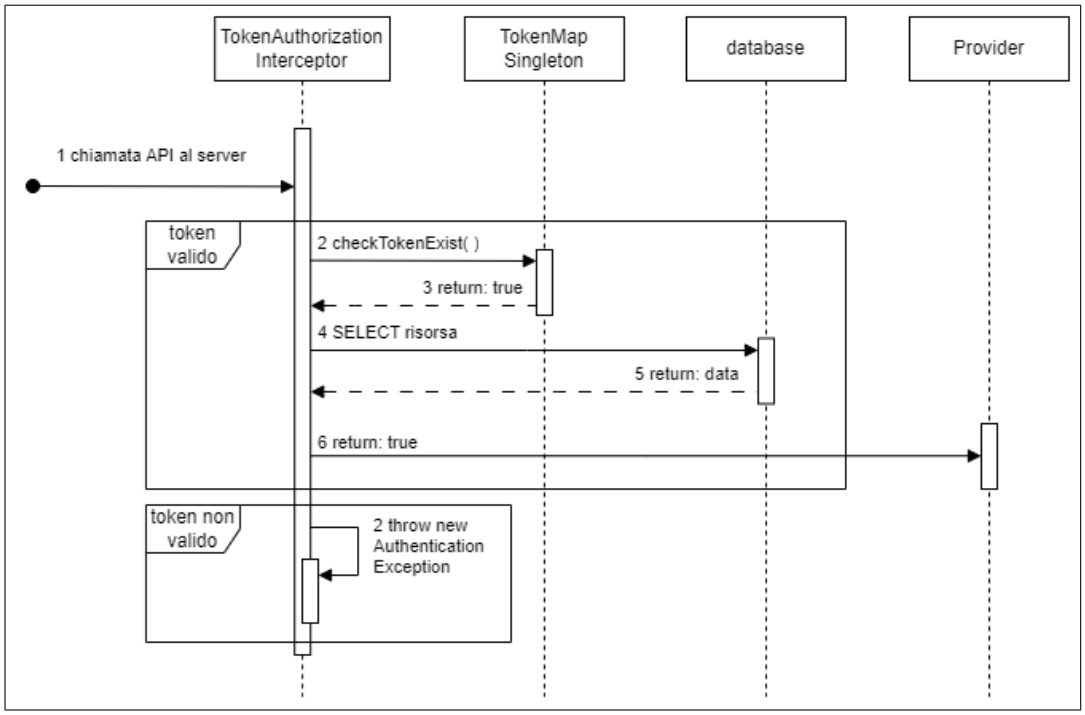

Figura 8: Use case dell'interceptor TokenAuthorizationInterceptor

5.3.3.3 Gestione dell'invio delle notifiche

Come definito all'interno della sezione 5.4, nel database viene tenuta traccia del livello di priorità che ogni medico assegna al percorso di telemonitoraggio di un determinato paziente. Insieme a questa informazione si salva anche il valore del token identificativo del dispositivo in cui il medico ha eseguito il login sull'app di telemonitoraggio.

Nel momento in cui giunge una richiesta di inserimento di una nuova osservazione al server FHIR, l'interceptor dedicato controlla se il relativo valore è fuori soglia. Nel caso in cui si verifica questo evento, si recuperano dal db tutti i medici che hanno espresso un livello di priorità per il paziente e il relativo identificativo del dispositivo. Ad ognuno di essi, mediante un'app Firebase sviluppata per questo progetto, viene inviata una notifica push tramite una richiesta *http* eseguita mediante la classe java *OkHttpClient*.

Il diagramma che illustra lo use case dell'interceptor è riportato in figura 9.

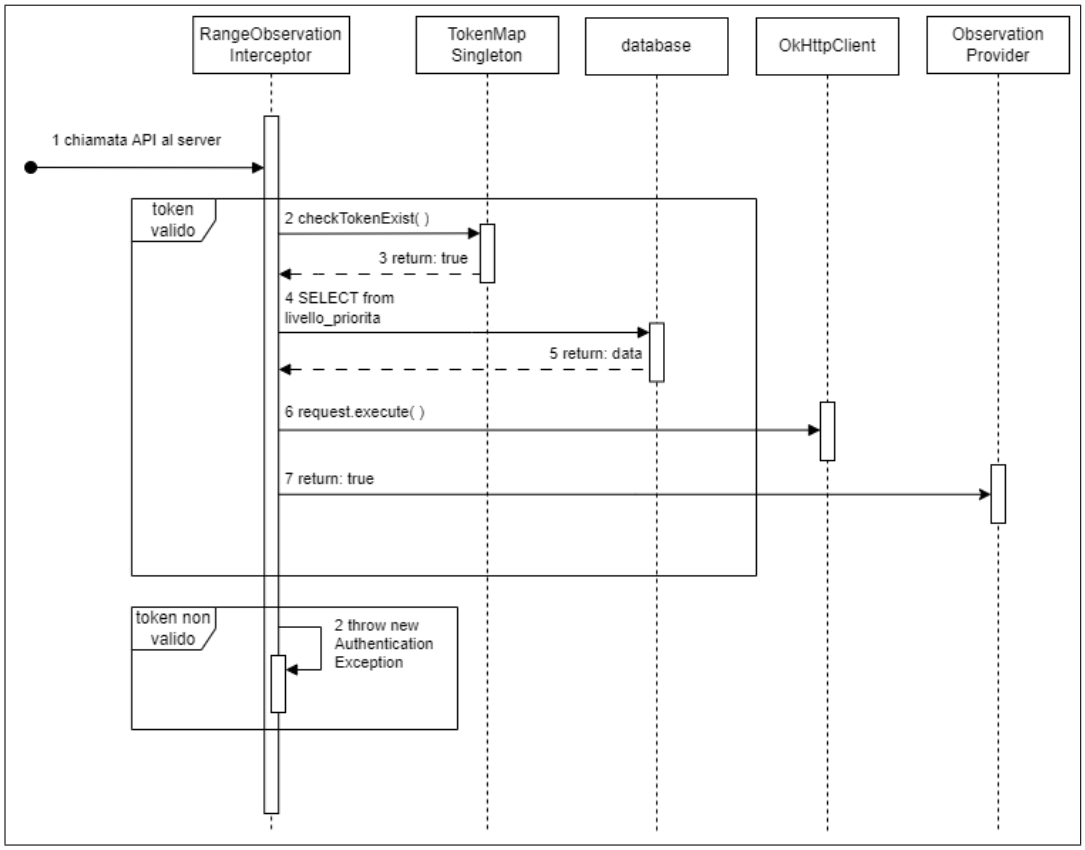

Figura 9: Use case dell'interceptor RangeObservationIntercetor

5.4 Database

I requisiti che hanno portato alla realizzazione e alla successiva implementazione delle varie istanze del database sono stati già espressi all'interno della sezione 3.

Analizzando la progettazione del database, in figura 10 è visibile lo schema logico. Si procede quindi con l'analisi della struttura delle singole tabelle; le tabelle costituenti il database sono cinque.

- *assistito*: questa tabella serve a tener traccia delle risorse Patient gestite dal server FHIR;
- *osservazione_master*: in questa tabella vengono salvati gli eventi gestiti da FHIR come risorsa Observation e indicizzati nel registro blockchain;
- *osservazione_details*: tale tabella serve a salvare le Observatione gestite nel solo server FHIR che vanno a definire gli eventi di cui sopra;
- *cross_assistito*: questa tabella è necessaria per salvare i dati che permettono di leggere correttamente le informazioni presenti all'interno della blockchain;

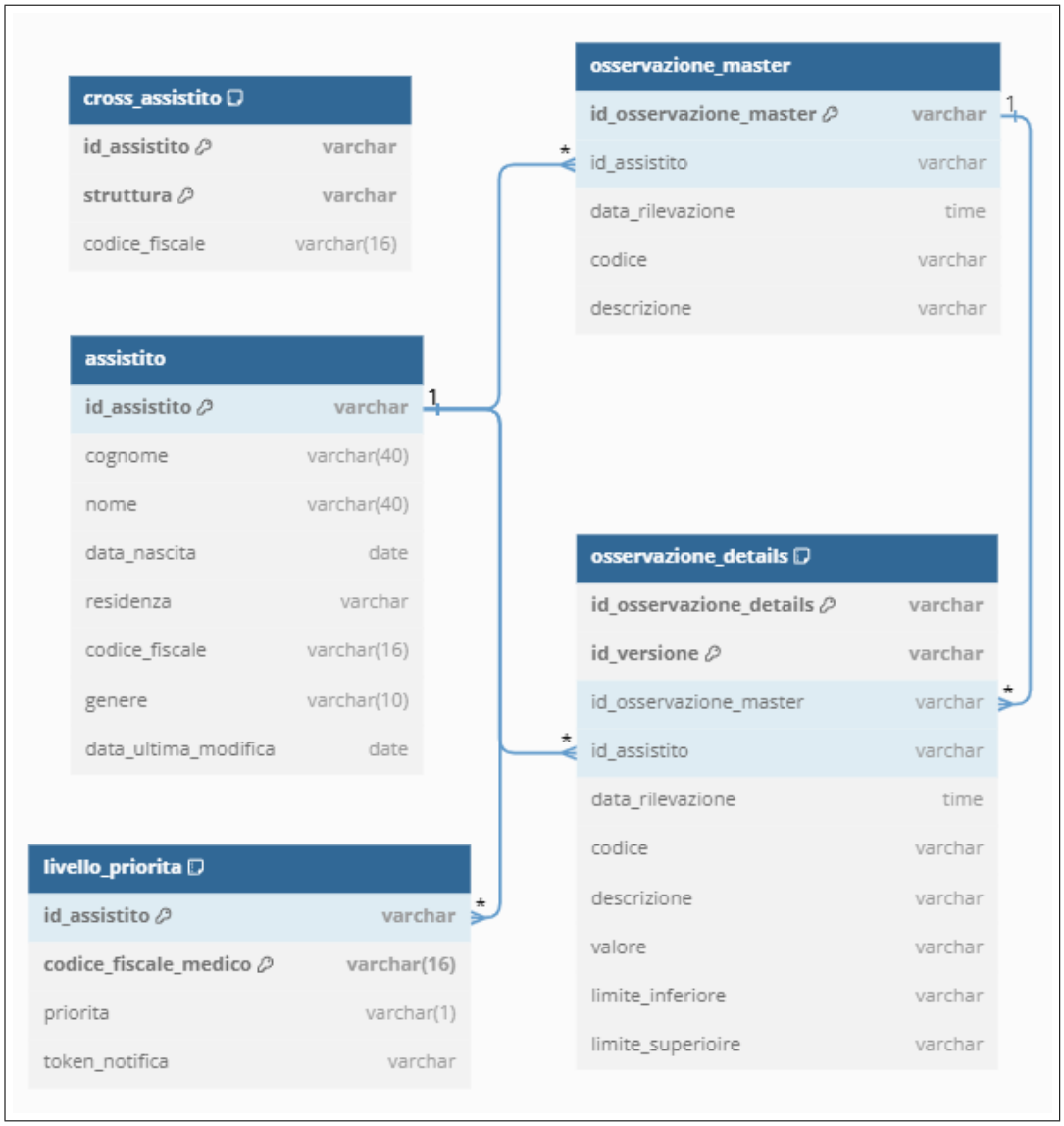

Figura 10: Schema logico del database

- *livello_priorità*: all'interno di questa tabella vengono memorizzati i livelli di priorità indicati dai medici per i pazienti della struttura oltre al token univoco del dispositivo a cui mandare eventuali notifiche.

Non si entra ulteriormente nei dettagli dei campi delle tabelle perché si ritiene che i loro nomi siano esplicativi di ciò che ogni singolo campo memorizza al suo interno e del relativo dominio.

5.4.1 Interfacciamento lato server FHIR

Ogni server FHIR dispone di un database in cui salvare permanentemente le risorse gestite, che simula la base di dati della struttura sanitaria. Per quanto riguarda la connessione e le chiamate verso il database sono state implementate delle soluzioni che permettono di rendere questo collegamento il più modulare e scalabile possibile, in vista di dover integrare questa soluzione in un contesto reale.

5.4.1.1 Connessione al database

All'interno del server FHIR è stato creato un file di configurazione, denominato *DBConfig.properties*, con i seguenti dati:

- *JDBCConnectionURL*: endpoint del database;
- *JDBCDriver*: driver da utilizzare per la connessione;
- *UserName*: nome utente necessario alla connessione;
- *Password*: password necessaria per la validazione del nome utente;
- *Instance*: istanza del server in cui è presente il database;
- *DefaultUserTimeout*: timeout in millisecondi dopo il quale considerare chiusa la connessione verso il database se non si riceve risposta.

Basterà quindi cambiare tale file per poter connettere il server ad un diverso database.

5.4.1.2 Chiamate verso il database

Tutte le tipologie di chiamate (SELECT, INSERT, UPDATE) sono parametrizzate ed indipendenti dalla struttura della tabella. Nel dettaglio, prima di ogni chiamata si effettua una SELECT per ottenere i campi della tabella indicata; ciò avviene mediante il codice riportato in figura 11.

A seguito di ciò, ad ogni campo ottenuto si associa il dato fornito nei parametri della chiamata al metodo. In figura 12 è riportato il codice che mostra la chiamata INSERT nel DB.

5.5 Web service restAPI

Per evitare che tutte le istanze dell'applicativo mobile si interfaccino direttamente con i nodi della blockchain, perdendo in parte il senso di avere una rete di tipo

```
private static String getTableFields(String tableName) {
    String fields = "";
    String query = null;
    try {
        query = "SELECT column_name AS COLUMN_NAME, "
                    + "CASE "
                    +" when DATA_TYPE = 'CHAR' then 'trim(' || COLUMN_NAME || ') as ' || COLUMN_NAME "
                    +" when DATA_TYPE = 'DATE' then 'coalesce(nullif(( to_char(' || COLUMN_NAME || ' , "
                    +"''yyyy/mm/dd hh24:mi:ss'')),''''),null) as ' || COLUMN_NAME "
                    +    " else COLUMN_NAME end as COLUMN_SELECT"
                    + " FROM "
                    + "information_schema.columns  "
                    + " WHERE  table_name = '"+tableName+"' order by COLUMN_NAME asc ";

        Statement stmtSelect = conn.createStatement(
            ResultSet.TYPE_SCROLL_INSENSITIVE,
            ResultSet.CONCUR_UPDATABLE
            );
        ResultSet rsSelect = stmtSelect.executeQuery(query);
        while(rsSelect.next()) {
            fields += rsSelect.getString("COLUMN_SELECT");

            if(!rsSelect.isLast()) {
                fields += ", ";
            }
        }
    } catch( Exception e) {
        System.out.println(e);
    }
    return fields;
}
```

Figura 11: Codice Java per ottenere i parametri della tabella

consortium, è stato implementato un web service restAPI che faccia da middlware tra le applicazioni e la rete blockchain.

Quindi, il vantaggio che si ottiene da questa modalità di gestione delle chiamate sta nell'evitare di esporre la rete blockchain. Per simulare ciò nel prototipo i tre nodi costituenti la rete sono stati generati in altrettanti volumi docker; solo il webservice ha accesso a tali volumi, mentre esso è esposto sull'intera rete aziendale della NBS s.r.l. a cui i dispositivi su cui sono istallati gli applicativi si connettono.

Il web service, come già enunciato nella sezione 5.2.2 fa riferimento al model implementato per la chain e ne invoca i metodi, così da avere un livello di astrazione maggiore rispetto all'implementazione degli smart contract.

Scendendo nel dettaglio, le chiamate gestite dal server sono quelle riportate nella tabella 10.

5.6 Interfaccia da riga di comando

L'interfaccia descritta in questa sezione, all'interno del prototipo, va a simulare un applicativo desktop a cui gli operatori sanitari possono interfacciarsi al fine di

```
public static boolean insert(TableName tableName, String[] valoriInsert){
    String tableNameString = tableName.getName();
    String fields = "";
    Statement stmt = null;
    int rowcount = 0;
    String query = "";
    String valoriQuery = "";
    int index = 0;

    for (String value : valoriInsert) {
        valoriQuery = valoriQuery.concat(value);

        if (index++ != valoriInsert.length -1){
            valoriQuery = valoriQuery.concat(", ");
        }
    }

    if(!connectDb()) return false;
    fields = getTableFields(tableNameString);

    query = "INSERT INTO "+ tableNameString + " ("
    + fields + ") VALUES (" + valoriQuery + ");";
    System.out.println(query);
    try {
        stmt = conn.createStatement(
            ResultSet.TYPE_SCROLL_INSENSITIVE,
            ResultSet.CONCUR_UPDATABLE
            );
        rowcount = stmt.executeUpdate(query);
        conn.close();
        if(rowcount == 0) return false;
    } catch (SQLException e) {
        e.printStackTrace();
    }
    System.out.printf("Success - %d - rows affected.\n",rowcount);
    //mostra il numero di righe interessate dalla INSERT
    return true;
}
```

Figura 12: Codice Java per la insert di risorse nel DB

interagire con il backend descritto nelle sezioni precedenti di questo capitolo.

Anch'essa, come il web service (sezione 5.5), fa riferimento ai metodi del model (sezione 5.2.2) per invocare le funzionalità della blockchain; per gestire le risorse FHIR invece fa capo alle richieste esposte dai server predisposti visibili nella tabella 9.

5.6.1 Workflow e casi d'uso

Si procede con una spiegazione dettagliata dei workflow possibili all'interno dell'interfaccia, illustrando e descrivendo i relativi schemi *use case UML* e, qualora lo si

Tabella 10: Richieste gestite dal web service

Metodo	**Rotta**	**Parametri**	**Descrizione**
GET	/Auth	cf_medico	Permette di ottenere tutte le autorizzazioni associate al medico per cui si è fornito il codice fiscale.
POST	/Auth	key	Attraverso questa query è possibile modificare l'autorizzazione, di cui si fornisce la chiave, andando a cambiare il campo che specifica se l'autorizzazione è accettata o meno dal medico; nel dettaglio eseguendo questa chiamata si procede con l'accettazione della collaborazione.
GET	/Struct	address	Eseguendo questa chiamata si ottengono i dati relativi alla struttura sanitaria salvati inchain: indirizzo del nodo ed endpoint del server FHIR.
GET	/Event	id_patient address	Permette di ottenere tutti gli eventi del paziente per cui si è fornito l'indicativo, registrati nella struttura corrispondente al parametro address.

ritenesse opportuno, le schermate che mostrano l'aspetto dell'interfaccia, nei passi più salienti della trattazione.

5.6.1.1 Meccanismo di login

All'avvio dell'applicativo sarà necessario effettuare il login come struttura sanitaria. Questo meccanismo simula, in sostanza, la possibilità di avere più istanze di app, una per ogni nodo della rete di telemedicina.

A seguito della selezione, attraverso una chiamata allo smart contract "ContractMedicalStruct", si ottengono una serie di dati legati all'istanza dell'interfaccia selezionata e, di conseguenza, al nodo relativo, quali:

- endpoint del server FHIR;

- indirizzo del nodo della blockchain.

A questo punto sarà possibile procedere selezionando un assistito su cui effettuare delle operazioni, oppure accedere all'aera riservata di un medico o, più in generale, di un professionista sanitario.

Il workflow appena descritto è visibile nella schema riportato in figura 13.

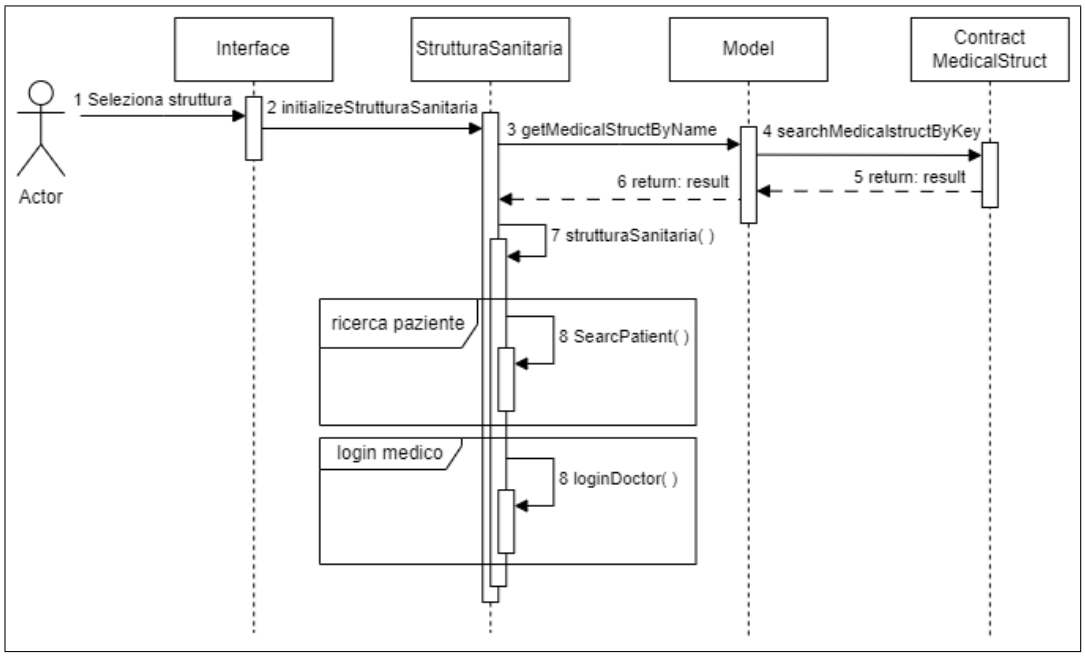

Figura 13: Use case meccanismo di login

5.6.1.2 Ricerca paziente e relative operazioni

Volendo registrare le informazioni relative ad un determinato paziente è dapprima necessario effettuare una ricerca di quest'ultimo per nominativo, inserendo nell'interfaccia nome e cognome dell'assistito.

Come è visibile in figura 14, è possibile anche inserire parte del nominativo per ottenere dal sistema tutte le corrispondenze rispetto agli assistiti registrati nella base di dati della struttura. Questo perché la condizione nella *SELECT*, riportata in figura 15, si basa sull'operatore *LIKE = %<nominativo>%*.

Nel caso in cui il paziente che si sta ricercando non sia tra quelli individuati nella richiesta al database, o comunque il risultato ottenuto nella ricerca sia vuoto, è possibile inserire una nuova risorsa di tipo Patient effettuando una chiamata al proprio server FHIR.

Una volta selezionato il paziente sarà visibile nell'interfaccia il menù delle operazioni disponibili per l'assistito indicato; nello specifico esse sono:

- Registrare una nuova osservazione;
- Autorizzare un nuovo medico a visualizzare le informazioni del paziente;
- Rimuovere un'autorizzazione concessa in precedenza.

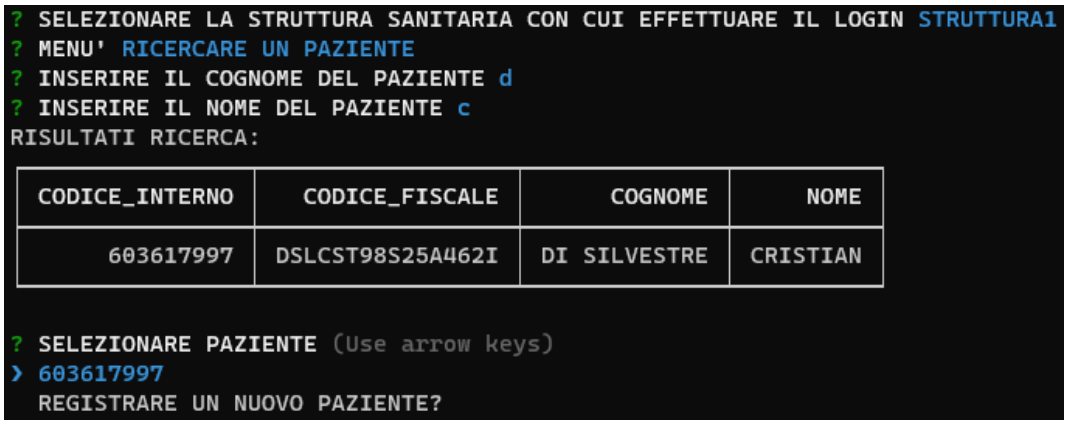

Figura 14: Schermata dell'interfaccia: funzionalità selezione paziente

5.6.1.3 Ricerca e inserimento evento

Prima di inserire una nuova osservazione, è necessario indicare l'evento a cui va associata; vengono, quindi, visualizzati tutti gli eventi indicizzati nella blockchain per il paziente selezionato dalla struttura sanitaria corrente, con la possibilità di selezionarne uno tra essi o di aggiungerne di nuovi.

L'aspetto dell'interfaccia rispetto alla funzionalità appena descritta è visibile in figura 16. Al fine dell'inserimento devono essere indicati codice, descrizione e data di avvio dell'evento. Una volta inserito o selezionato l'evento sarà possibile inserire la nuova osservazione.

Valutando nel complesso la ricerca e l'eventuale inserimento di un nuovo evento, in figura 17, è visibile l'intero workfolow. Come già indicato, questo tipo di risorse vengono indicizzate nella blockchain e salvate nel database della struttura attraverso una chiamata al server FHIR.

5.6.1.4 Inserimento osservazione

Per inserire una nuova osservazione è necessario valorizzare i seguenti campi:

- codice;

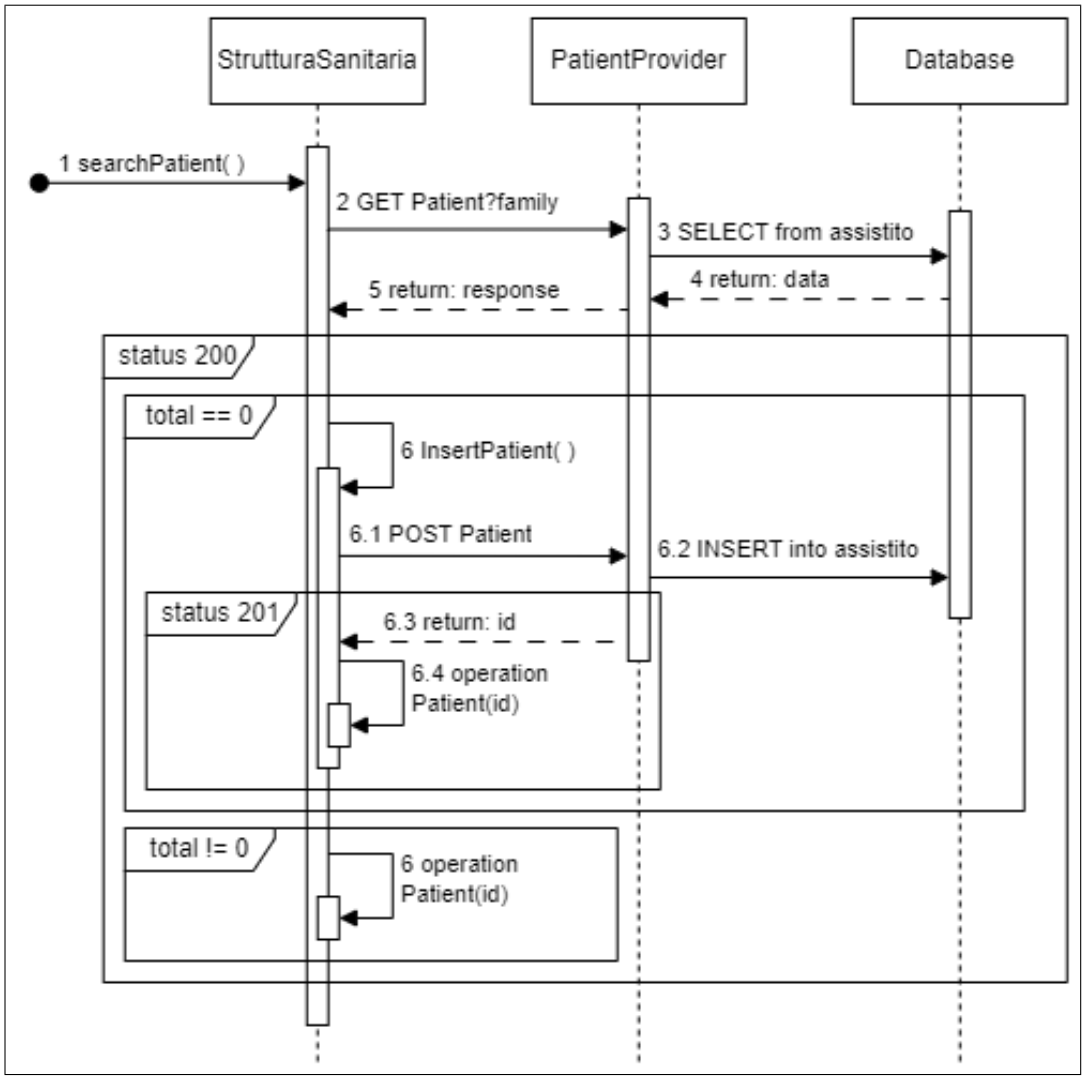

Figura 15: Use case ricerca paziente e relative operazioni

- descrizione;
- data di esecuzione;
- valore dell'osservazione;
- valore minimo accettabile per l'osservazione;
- valore massimo accettabile per l'osservazione.

L'identificativo dell'evento e del paziente a cui l'osservazione fa capo viene inserito automaticamente dall'interfaccia sulla base delle selezioni fatte fino al momento dell'inserimento dell'osservazione e descritte nelle sezioni precedenti.

Lo use case di questa funzionalità è visualizzabile in figura 18.

```
? SELEZIONARE UN'OPERAZIONE: REGISTRARE UNA NUOVA OSSERVAZIONE PER IL PAZIENTE SELEZIONATO

NESSUN EVENTO REGISTRATO PER IL CODICE PAZIENTE INSERITO
? INSERIRE UN NUOVO EVENTO Yes
? INSERIRE CODICE EVENTO 01
? INSERIRE VERSIONE ESPLICITA DESCRIZIONE EVENTO PERCORSO DI TELEMONITORAGGIO
? INSERISCI LA DATA 2021-09-11T09:30:10+01:00
?
NOME: DI SILVESTRE CRISTIAN
CODICE EVENTO: 01
DESCRIZIONE EVENTO: PERCORSO DI TELEMONITORAGGIO
DATA: 2021-09-11T09:30:10+01:00

CONFERMARE L'INSERIMENTO? Yes

TRANSAZIONE ESEGUITA
EVENTO INSERITO CORRETTAMENTE
?
INSERIRE UNA NUOVA OSSERVAZIONE PER QUESTO EVENTO (Y/n)
```

Figura 16: Schermata dell'interfaccia: funzionalità inserimento evento

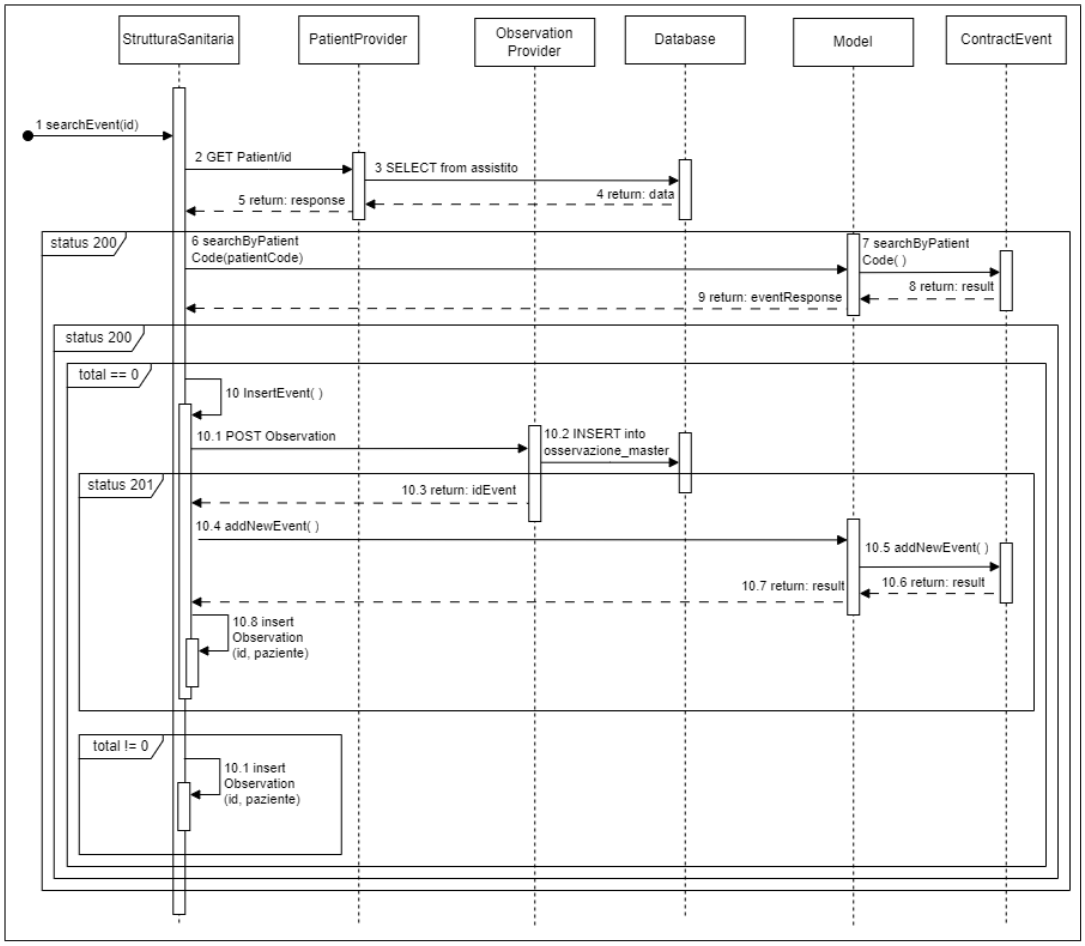

Figura 17: Use case inserimento evento

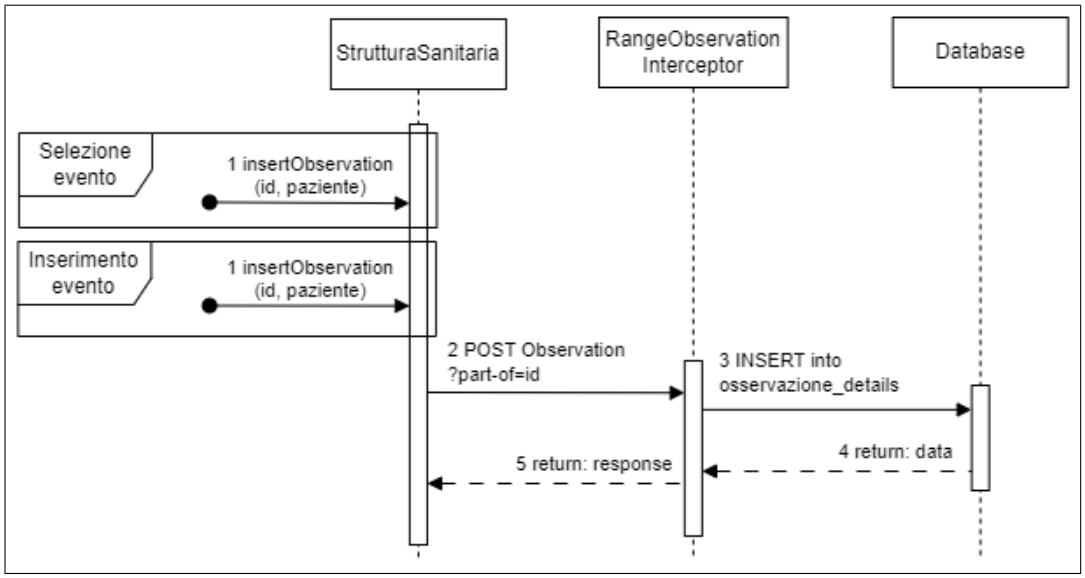

Figura 18: Use case inserimento osservazione

5.6.1.5 Inserimento autorizzazione

Altra operazione possibile è autorizzare un medico a visualizzare le informazioni di un assistito.

Per poter accedere ai dati del paziente il medico dovrà collaborare con la struttura sanitaria a cui l'assistito si è rivolto, al fine di seguire il percorso medico del paziente. Formalmente, quindi, si inoltra una richiesta di collaborazione al professionista sanitario, collaborazione che quest'ultimo potrà decidere di accettare o meno.

Il prototipo gestisce questa funzionalità con i passi riportati di seguito e schematizzati in figura 19:

- si immette nel proprio server FHIR un token di autorizzazione, necessario alla validazione del token fornito dalla bockchain al medico;
- si inizializza una nuova autorizzazione nella blockchain indicando il valore *false* per l'attributo *accepted*;
- si immette una nuova risorsa *PriorityLevel* nel proprio server FHIR inizializzando la priorità a "0" e il token del dispositivo a cui inviare le eventuali notifiche a "null"[2].

[2]valore che indica la mancata valorizzazione di un campo nel database

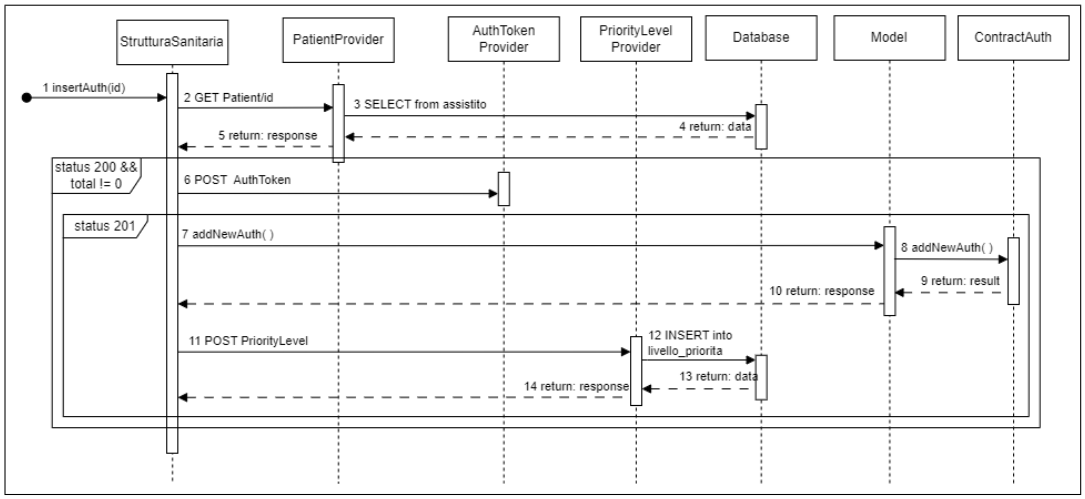

Figura 19: Use case inserimento autorizzazione

5.6.1.6 Rimozione autorizzazione

Ultima operazione disponibile su un determinato assistito consiste nel rimuovere a un medico l'autorizzazione a visualizzarne le informazioni sanitarie.

Come visibile in figura 20, viene invocata un'operazione di *DELETE* sul server FHIR e rimosso il token dallo smart contract relativo nella blockchain.

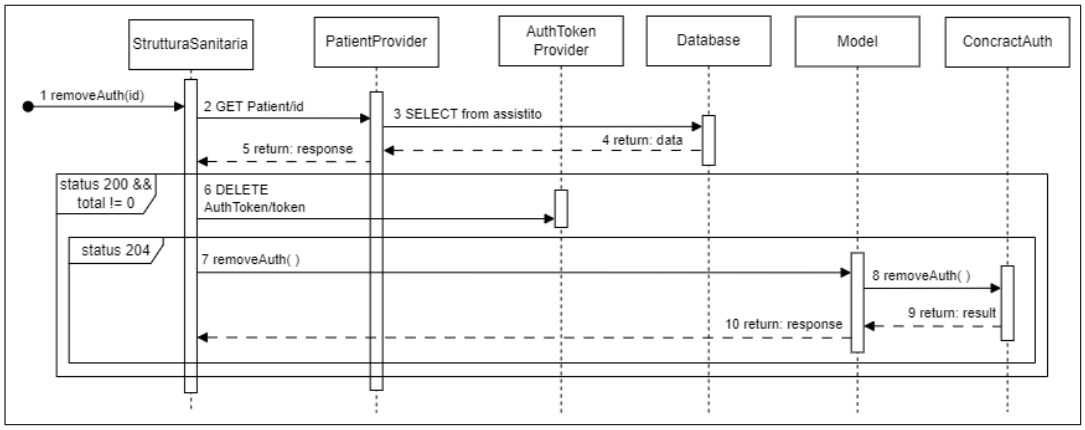

Figura 20: Use case rimozione autorizzazione

5.6.1.7 Login medico e accettazione collaborazione

Oltre a poter effettuare delle operazioni relative agli assistiti gestiti dalla struttura sanitaria, l'interfaccia offre anche la possibilità di accedere all'area riservata di un medico o di un professionista sanitario.

Ogni struttura riconosce le credenziali dei soli medici che operano al suo interno; il professionista sanitario potrà quindi accedere alla sua area riservata nelle sole strutture presso cui lavora.

All'interno della proprio area riservata, un medico può visualizzare tutte le collaborazioni (relative ad altrettante autorizzazioni concesse da altre strutture sanitarie) che siano accettate o ancora da accettare.

Considerando che le autorizzazioni, all'interno della blockchain, vengono indicizzate salvando il codice fiscale del medico, quest'ultimo può visualizzare tutte le collaborazioni a esso richieste eseguendo l'accesso presso ogni struttura sanitaria di cui fa parte; un professionista sanitario, ad esempio, può operare in un ospedale pubblico e in più cliniche private.

Come mostrato in figura 21, nel momento in cui si accetta una collaborazione, attraverso una lettura sul registro della blockchain si ottengono i dati relativi alla struttura sanitaria che ha inoltrato la richiesta di collaborazione, tra cui l'endpoint del relativo server FHIR. Server a cui viene inviata una richiesta per ottenere le informazioni di cross necessarie ad associare il codice fiscale dell'assistito al codice interno con cui esso è indicizzato nello smart contract. Queste informazioni vengono poi salvate all'interno del database della struttura presso la quale il medico ha eseguito l'accesso.

Una volta accetta, l'autorizzazione viene mostrata tra le collaborazioni confermate.

5.6.1.8 Visualizzazione autorizzazioni e relative osservazioni

Nella propria area riservata il medico può consultare la lista di tutti gli assistiti per cui gli è stata concessa l'autorizzazione, necessaria ad accedere alle relative osservazioni sanitarie.

Selezionando un evento, indicizzato nella blockchain, quest'ultima rilascia un token con il quale l'applicativo di interfaccia invia due richieste di tipo GET al server FHIR della struttura sanitaria, che detiene i dati:

- la prima ottenere i dati relativi all'evento;
- la seconda per ottenere le osservazioni a esso associate.

Come nei casi precedenti, prima di questa operazione, è necessario ottenere l'endpoint del server a cui inoltrare le richieste invocando i metodi dell'apposito smart contract salvato nella blockchain.

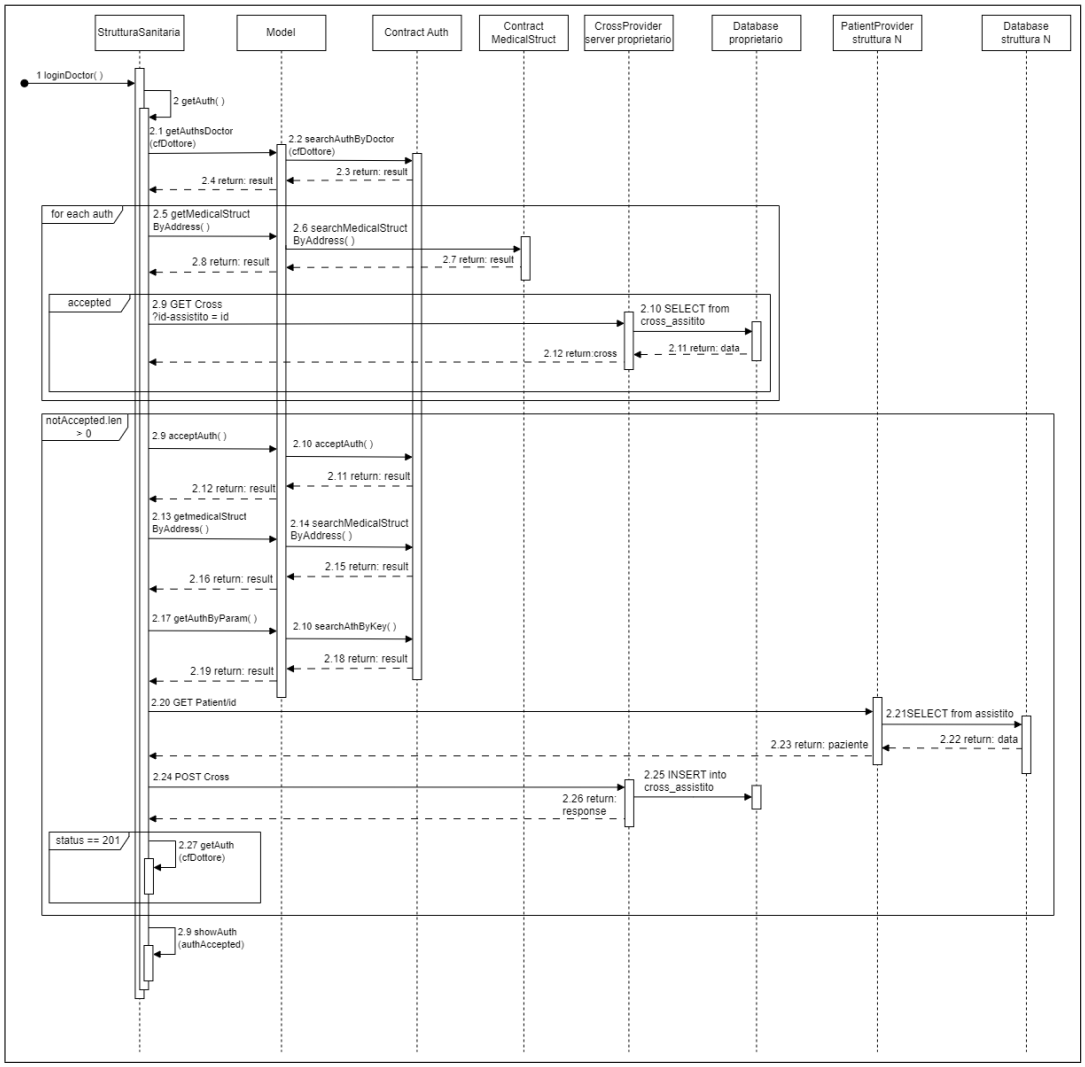

Figura 21: Use case login medico e accettazione collaborazione

5.7 Applicativi mobile

La prima scelta progettuale da discutere, affrontando la trattazione degli applicativi mobile, è quella che ha portato allo sviluppo di due app separate anziché di un singolo applicativo che permettesse l'accesso a diversi livelli di utenza. Come già riportato più volte all'interno di questo documento di tesi, garantire una buona modularità per l'intero prototipo è stato da subito uno dei requisiti fondamentali che ha guidato molte delle scelte intraprese in fase di progettazione. Avere due app distinte permette, difatti, di mantenere separati moduli software e funzionalità che hanno poco in comune, sia in termini implementativi che semantici, oltre che di alleggerire il peso dell'applicativo stesso aumentandone l'efficienza. Infine, in un'ottica di implementazione su vasta scala del prototipo realizzato, è verosimile che

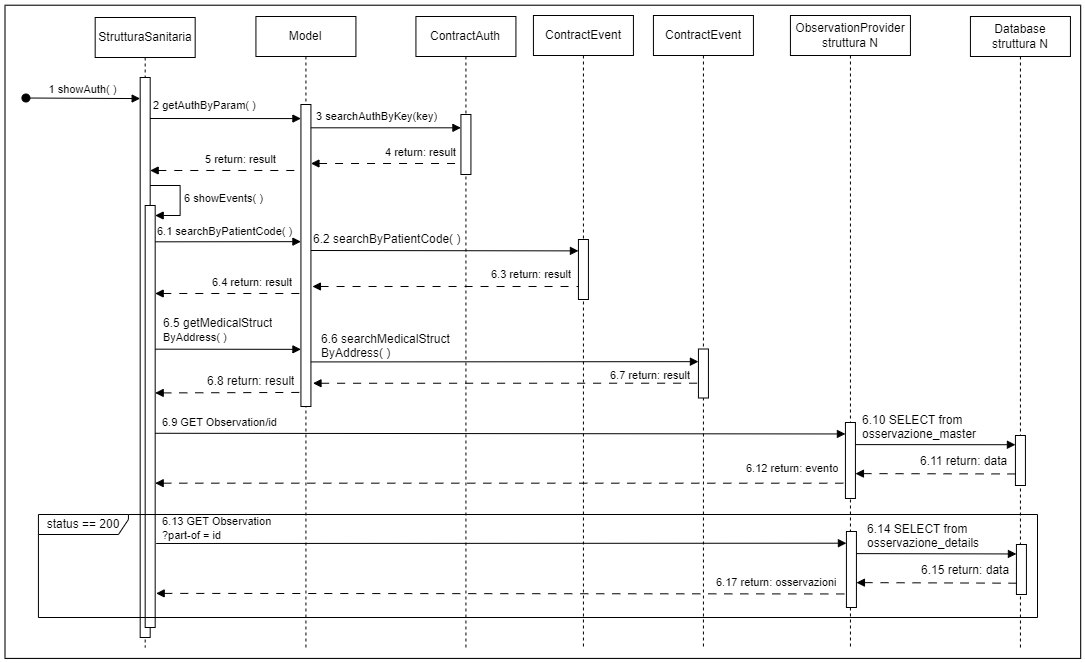

Figura 22: Use case visualizzazione autorizzazioni e relative osservazioni

non tutte le strutture sanitarie aderenti alla rete adottino entrambe le applicazioni realizzate; con questa scelta, quindi, si garantisce anche una buona flessibilità del sistema per adattarsi alle esigenze dei vari utenti.

Si procede con l'analisi dei due applicativi implementati, descrivendone le funzioni, mostrandone l'interfaccia grafica e scendendo nel dettaglio degli use case che li caratterizzano.

5.7.1 Gestione visualizzazione informazioni

Angular, framework utilizzato per lo sviluppo delle pagine degli applicativi mobile, offre una serie di direttive utili per la formattazione condizionale delle pagine HTML. Nel dettaglio le direttive strutturali modificano la struttura del DOM aggiungendo o rimuovendo elementi sulla base di condizioni legate a variabili definite lato Typescript.

- La direttiva ngIf aggiunge un blocco di elementi in base alla valutazione di una espressione booleana.
- La direttiva ngSwitch consente di generare un blocco di elementi da un insieme di possibili blocchi in base al valore di un'espressione.

- La direttiva ngFor consente di generare un elenco di blocchi di elementi basati sul contenuto di una lista.

Nella generazione dei DOM delle pagine presenti in entrambe le applicazioni sono state usate le direttive appena riportate. Di seguito sono presenti due esempi di utilizzo che riportano rispettivamente:

- la definizione delle card per la visualizzazione delle osservazioni legate ad un evento di telemonitoraggio nell'app dedicata al livello di utenza dei medici (figura 23);
- la definizione delle card per la registrazione di osservazioni nell'app dedicata agli assistiti (figura 24).

Nei codici a cui si è appena fatto riferimento, come prefisso delle direttive è riportato un "*". Questo carattere rappresenta un'abbreviazione sintattica per indicare che il markup sotto il controllo della direttiva è un template da utilizzare per la generazione degli elementi del DOM. [34]

I vantaggi ottenuti dall'utilizzo di tali direttive sono diversi:

- maggiore leggibilità del codice;
- maggiore manutenibilità dello stesso;
- maggiore modularità nella gestione dei dati.

5.7.2 App Paziente

Come già introdotto nella sezione 5.1, l'app per gli assistiti ricopre diverse funzionalità (osservare la figura 25 per comprendere meglio la descrizione che segue):

- Recupero e normalizzazione dei dati provenienti dai vari sensori presenti nel luogo in cui l'assistito sta effettuando il percorso di telemonitoraggio, oltre che l'ottenimento delle informazioni registrate da altre applicazioni presenti sul dispositivo, attraverso i servizi di Google Fit, fungendo da *ponte di normalizzazione*. Con il termine normalizzazione di dati si fa riferimento al processo di organizzazione e strutturazione dei dati raccolti in risorse FHIR di tipo Observation.
- Login dei pazienti attraverso delle credenziali; per questa funzionalità è stato utilizzato un modulo software aziendale della NBS s.r.l. che si interfaccia con

```
<ion-card *ngFor="let observation of observations" class="custom-card">
  <ion-card-header>
    <ion-card-title><div class="description">
      {{ observation.title }}
    </div>
    <div class="value" >
      {{ observation.value }}
    </div></ion-card-title>
  </ion-card-header>
  <ion-card-content>
    {{ observation.description }}
  </ion-card-content>
</ion-card>
-------------------------------------------------------------------------------------
this.observations = [];
this.httpClientService.getObservation(url, headers).subscribe((res) => {
  if(res.total<=0) { this.toastService.presentToast(
    "top",
    "Nessuna osservazione registrata per la data selezionata!"
    )}
  else {
    this.observations = res.entry.map((element) => {
      let title: any = this.osservazioniService
      .getTitleCardFromCode(element.resource.code.coding[0].code);
      return {
        title: title,
        description: element.resource.code.coding[0].display,
        value: element.resource.valueQuantity.value
      };
    })
  }
});
```

Figura 23: Codice HTML e Typescript utilizzo direttiva ngFor Angular

un database di test con delle credenziali di prova. In un contesto reale questo database rappresenta una sezione della base di dati della struttura sanitaria.

- Registrazione di tutti i dati raccolti nel server FHIR della struttura sanitaria che ha richiesto l'avvio del percorso di telemonitoraggio.

Scendendo nel dettaglio dell'invio dei dati al server, è necessario l'identificativo del paziente e il codice univoco dell'evento di telemedicina a cui aggiungere le osservazioni registrate dall'app. Il primo viene rilevato in fase di login mentre per il secondo sarà necessario che l'operatore della struttura sanitaria, dopo aver indicizzato nella blockchain l'evento, inserisca il codice nel'applicativo.

```
<ion-card-content  [ngSwitch]="card.type">
  {{ card.description }}
  <br>
  <ion-input *ngSwitchCase="'input'" label="Valore"
    labelPlacement="floating" [(ngModel)]="card.value"
    (ionInput)="changeInput(card)" type="number" fill="solid"
    helperText="Inserire un valore numerico">
  </ion-input>
  <ion-checkbox *ngSwitchCase="'check'" (ionChange)="checkBox(card)"
    labelPlacement="end" checked={{card.value}}>Medicinale assunto correttamente
  </ion-checkbox>
  <div *ngSwitchCase="'increment'">
    <ion-text>{{card.value}}</ion-text>
    <br>
    <ion-button color="primary" (click)="decrement(card)">
      <ion-icon name="remove-circle-outline"></ion-icon>
    </ion-button>
    <ion-button color="primary" (click)="increment(card)">
      <ion-icon name="add-circle-outline"></ion-icon>
    </ion-button>
  </div>
  <br>
  <div id="ended">
    <ion-button color="danger" (click)="deleteCard(card)">
      <ion-icon name="trash-outline"></ion-icon>
    </ion-button>
  </div>
</ion-card-content>
```

Figura 24: Codice HTML utilizzo direttiva ngSwitch Angular

Questa scelta è stata intrapresa al fine di ridurre al minimo le interazioni con la blockchain e far si, quindi, che solo il personale sanitario autorizzato ne abbia il diritto.

5.7.2.1 Registrazione e invio delle osservazioni al server FHIR

In seguito all'accesso il paziente viene indirizzato nella pagina principale (di cui è visibile una schermata in figura 26) da cui è possibile attivare diverse operazioni:

- sincronizzare i dati tramite i servizi offerti da Google Fit;
- aggiungere osservazioni e di conseguenza inviare i dati al server FHIR.

Il processo di selezione dell'operazione è gestito dai moduli di routing dell'app e della Tab, visibile in basso nella schermata (figura 26). Attraverso la definizione di path associati ai percorsi prefissati, percorribili dall'utente per navigare all'interno dell'applicativo, e al caricamento di *pagine figlio* all'interno della schermata è possi-

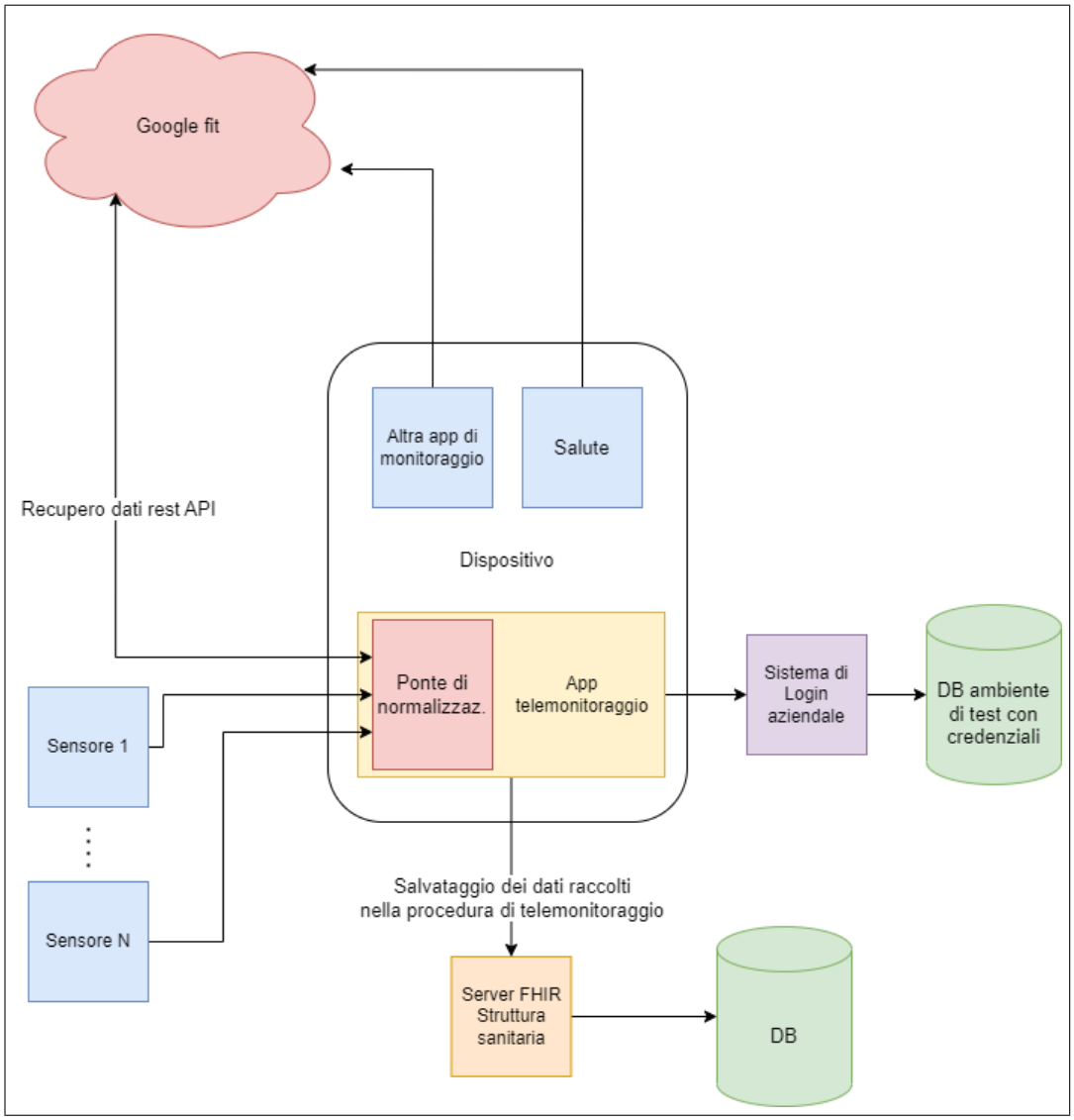

Figura 25: Architettura e interazioni dell'applicativo mobile per i pazienti

bile gestire l'intera navigazione. Lo use case dell'apertura dell'app è visibile in figura 27.

Le osservazioni registrabili manualmente dall'utente sono di tre tipologie:

- inserimento di un testo (come nel caso della scheda *Saturazione* in cui è richiesto l'inserimento di un valore numerico);
- conferma di un evento (come nella scheda *Compressa* in cui si registra l'avvenuta assunzione di un medicinale);
- incremento di un valore (come nel caso della scheda *Acqua* in cui si registra la quantità di acqua assunta durante il giorno).

Figura 26: Aspetto dell'area riservata dell'applicativo mobile per i pazienti

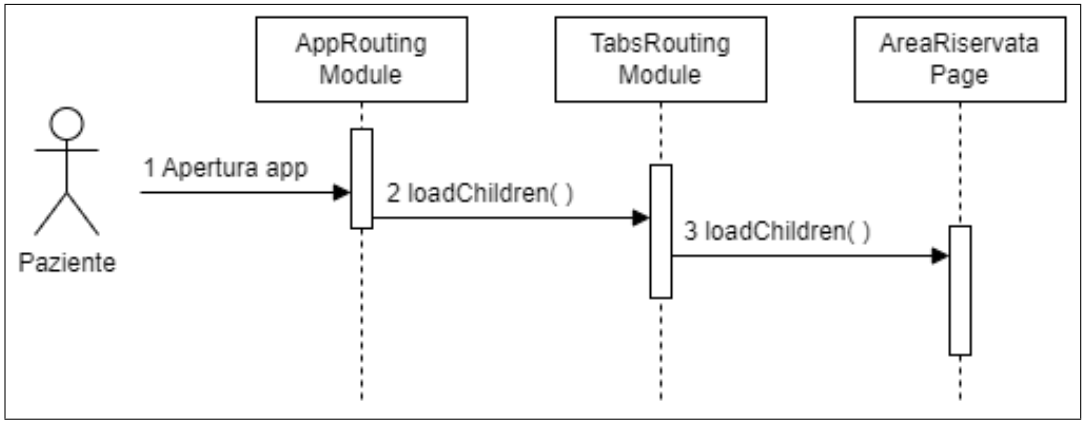

Figura 27: Use case apertura applicativo paziente

Per aggiungere una nuova osservazione è necessario cliccare sul pulsante "AGGIUNGI OSSERVAZIONE" (figura 26) e selezionare la scheda di interesse, eventualmente ricercandola attraverso la barra di ricerca (figura 28).

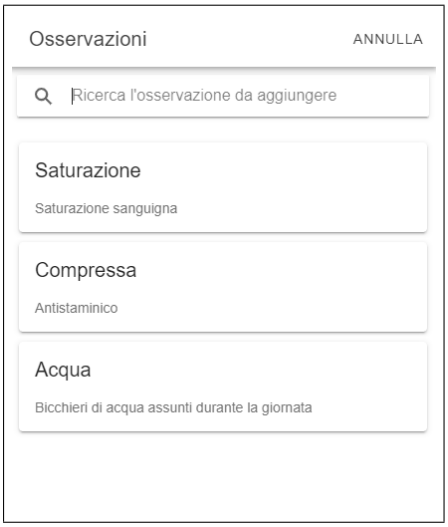

Figura 28: Inserimento nuova osservazione nell'applicativo mobile per i pazienti

Cliccando poi sul pulsante "SALVA I DATI" (figura 26) avviene l'invio delle informazioni relative alle osservazioni inserite. L'invio è condizionato da un processo di sanificazione degli input; volendo fare un esempio legato alle schede visibili in figura 26 non è possibile inviare il dato *Saturazione* senza aver popolato il campo di inserimento testuale.

Come nel caso dell'interfaccia da riga di comando, le osservazioni non vengono salavate nella blockchain ma solo all'interno del server FHIR. Il processo dettagliato è mostrato in figura 29.

5.7.2.2 Sincronizzazione con Google Fit

Al fine di poter sincronizzare l'applicativo a Google Fit, è necessario utilizzare il servizio di autenticazione OAuth2; occorre quindi:

- creare un'app nella console di Google e associare a essa delle credenziali per ogni piattaforma su cui si intende istallare l'applicativo di telemonitoraggio (web, Android, iOS);
- inserire tali credenziali in un file .options presente nell'app mobile;
- creare una schermata per permettere all'utente di dare il consenso al trattamento dei dati (mostrata in seguito al primo login);
- eseguire, prima delle chiamate ai servizi Google, l'autenticazione con l'account Google dell'utente (attraverso un reindirizzamento verso la pagina web di login di Google).

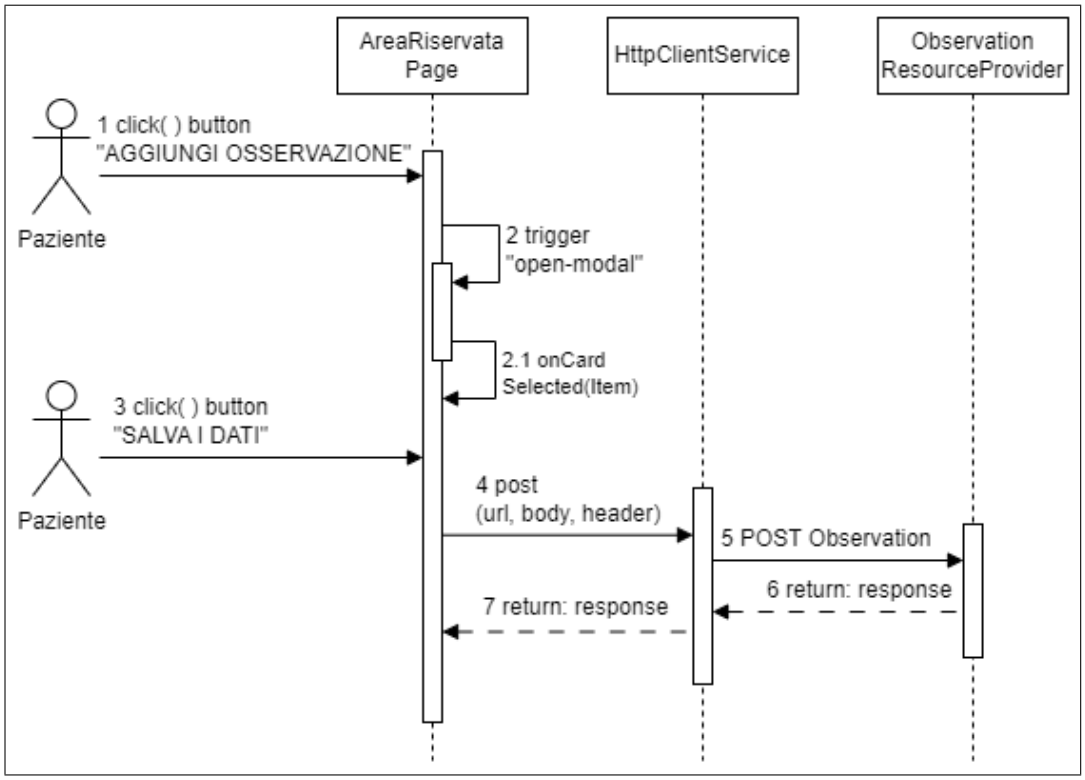

Figura 29: Use case inserimento e salvataggio osservazione nell'applicativo paziente

Cliccando sul pulsante "GOOGLE FIT", presente nella barra degli strumenti della schermata in figura 26, l'utente potrà eseguire l'accesso al suo account Google; in seguito, verranno scaricati i dati presenti in Google Fit per poi essere convertiti in risorse FHIR e inviati al relativo server. Lo use case del processo appena descritto è mostrato in figura 30.

5.7.2.3 Storico

Un'altra possibilità offerta all'utente è quella di visualizzare lo storico delle osservazioni a esso associate. Cliccando sul pulsate "Storico" nella *tab di navigazione* si giunge alla schermata visibile in figura 31. Qui è presente un calendario e selezionando una data, attraverso una richiesta al server FHIR vengono mostrate tutte le osservazioni registrate durante la giornata indicata.

Lo use case possibile all'interno di questa pagina è visibile in figura 32.

5.7.2.4 Utilizzo di Capacitor Preferences

Capacitor è il runtime multipiattaforma di Ionic che semplifica la creazione di applicazioni mobile che vengono eseguite in modo nativo su iOS, Android e altri disposi-

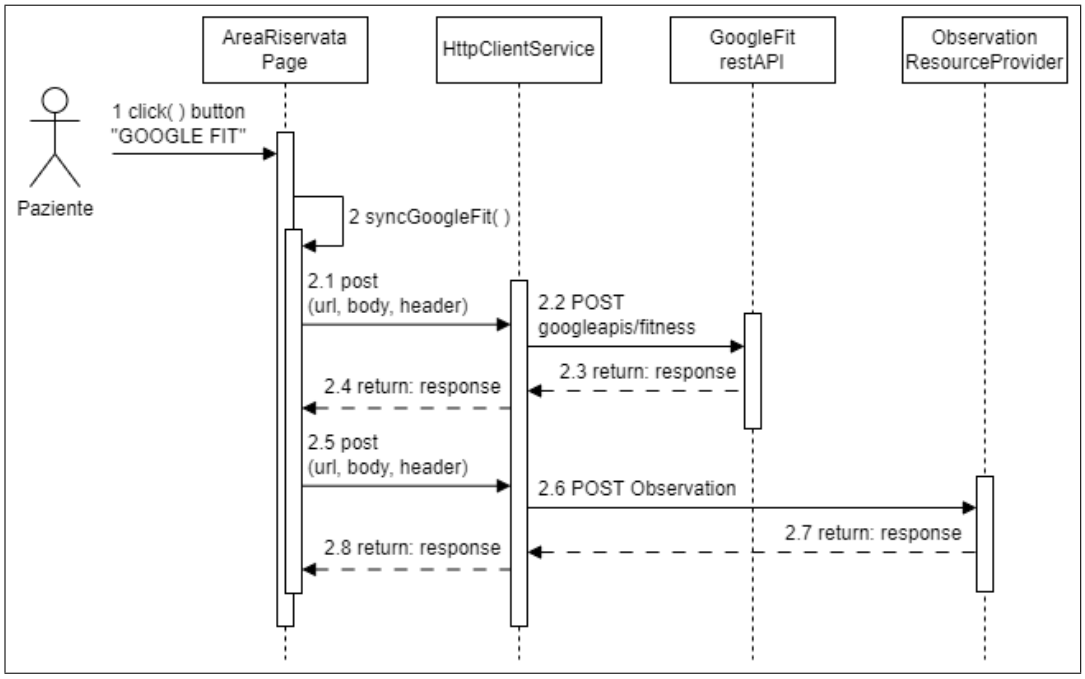

Figura 30: Use case sincronizzazione con servizi Google Fit nell'applicativo paziente

tivi. Sono disponibili diversi plugin che permettono di astrarre funzionalità native dei vari sistemi operativi, uno tra essi è Capacitor Preferences.

Quest'ultimo fornisce un archivio persistente chiave/valore per dati leggeri. All'interno di questo archivio vengono salvati tutti i dati e le informazioni che descrivono le interazioni dell'utente con l'applicazione.

Nel dettaglio le informazioni di cui si tiene traccia sono:

- le card delle osservazioni selezionate e non ancora inviate al server FHIR;
- i valori inseriti nelle card.

5.7.3 App Medico

L'applicazione sviluppata per il livello di utenza dei medici e dei professionisti sanitari ha lo scopo di garantire all'utente una forte flessibilità nel monitoraggio dei propri assistiti. Le funzionalità implementate al suo interno per garantire ciò sono le seguenti:

- Interfacciamento con la blockchain per ottenere le autorizzazioni concesse al medico e gli eventi, indicizzati nei registri, relativi agli assistiti protagonisti delle autorizzazioni, oltre che per ottenere i token di autorizzazione. Le chia-

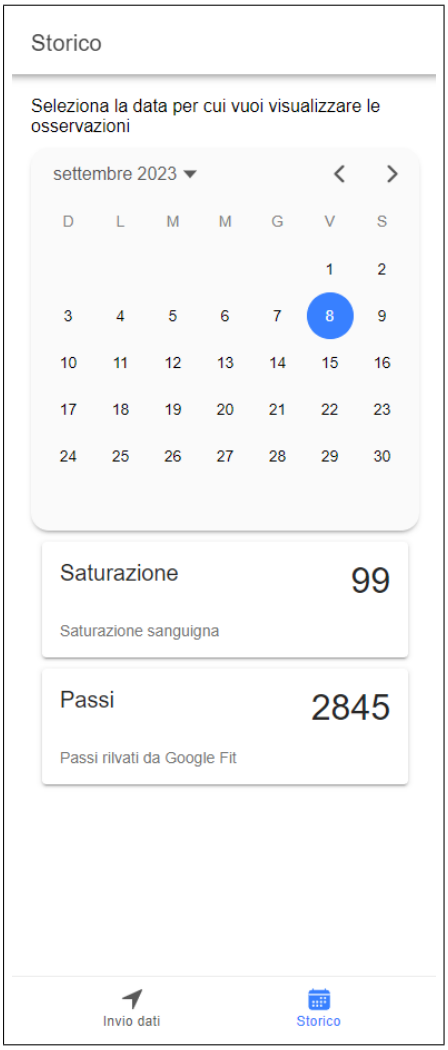

Figura 31: Aspetto dello storico dell'applicativo mobile per i pazienti

mate, come già indicato, sono gestite dal webservice (meccanismo approfondito nella sezione 5.7.3.1).

- Interfacciamento verso il server FHIR per ottenere le osservazioni riferite ad un determinato evento.
- Ricezione di notifiche basate sul livello di priorità assegnato a ciascun paziente, in caso di registrazione di osservazioni anomale.

Come per l'applicazione mobile con livello di utenza dedicato agli assistiti, anche per questa app il login è gestito attraverso un modulo software aziendale proprietario di NBS s.r.l..

Uno schema dell'architettura dell'applicativo è riportato in figura 33.

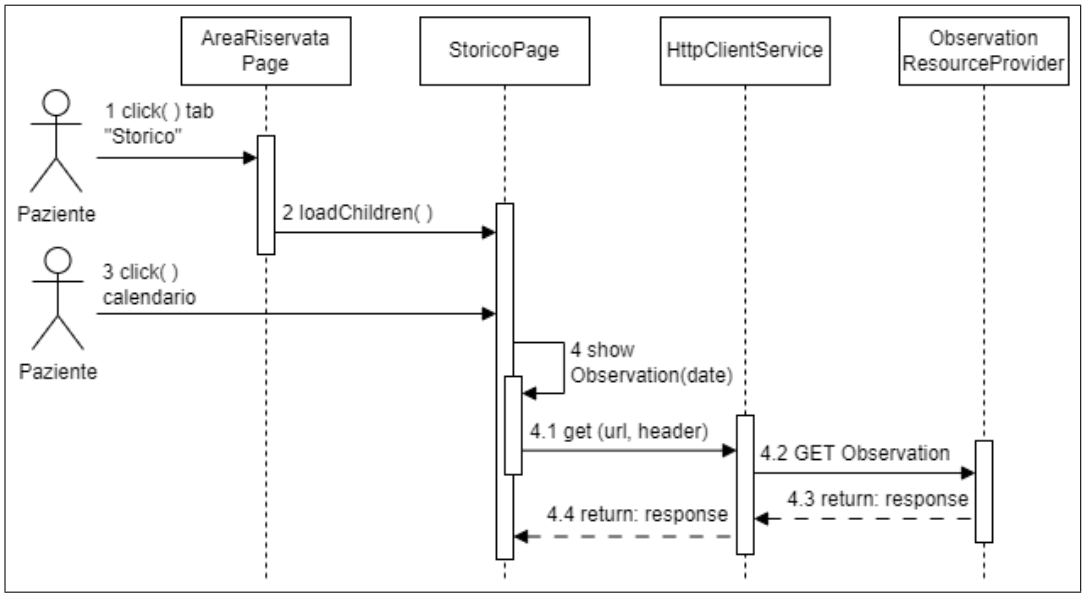

Figura 32: Use case interazioni con lo storico nell'applicativo paziente

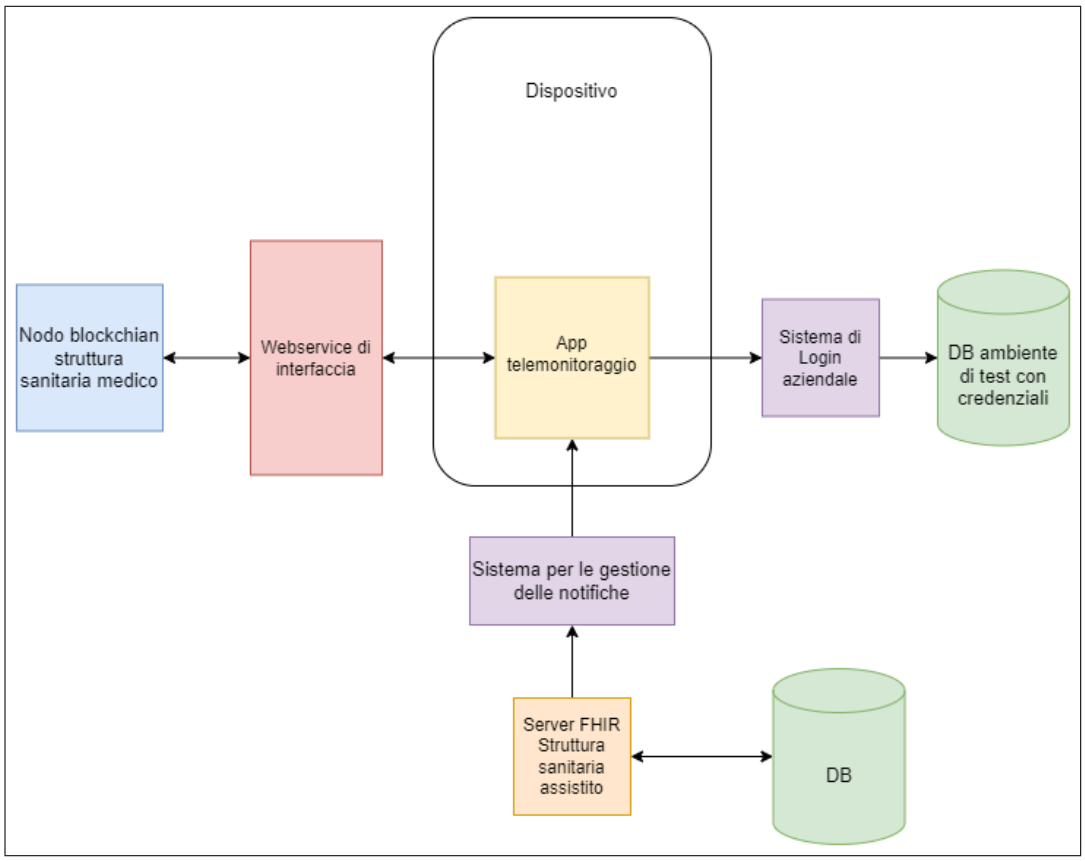

Figura 33: Architettura e interazioni dell'applicativo mobile per i medici

5.7.3.1 Service implementati nell'applicazione

Per la gestione dei vari servizi presenti all'interno dell'app mobile, sono stati implementati una serie di service: moduli software destinati al controllo e all'esecuzione di una particolare funzionalità. La scelta di creare questi moduli è guidata, ancora una volta, da un'ottica di modularità: nel caso in cui si decida di voler cambiare il metodo in cui si implementa un dato servizio basterà modificare il codice presente all'interno del relativo modulo.

Toast service Necessario alla creazione e alla presentazione all'utente di messaggi di tipo toast: piccola area che appare in una zona ben definita del display contenente un messaggio, la cui visibilità dura qualche secondo. Invocando il metodo per stampare a schermo un toast è necessario fornire il testo da visualizzare e la posizione in cui visualizzarlo ("top", "middle" o "bottom").

Notification service Per la gestione del sistema di notifiche all'interno dell'applicativo in esame viene utilizzato un service apposito avente le funzionalità di:

- inizializzare il profilo dell'utente legato al dispositivo a cui inviare le notifiche, generando il relativo token univoco;
- gestire il token identificativo del dispositivo a cui inviare le eventuali notifiche attraverso il pattern *Singleton* esponendo un metodo di get pubblico invocabile dagli altri moduli dell'applicativo che hanno necessità del token;
- gestire gli eventi relativi al meccanismo di notifiche: arrivo e visualizzazione di una nuova notifica, click sul banner e relativa apertura della notifica, logout dal profilo utente con relativa dissociazione del token.

HTTP client service Per la gestione delle chiamate HTTP ai vari servizi è stato sviluppato un codice che si rifà al pattern *Facade*. Esso indica un oggetto che permette, attraverso un'interfaccia più semplice, l'accesso semplice a sottosistemi che espongono interfacce complesse e molto diverse tra loro, nonché a blocchi di codice complessi.

Il codice in questione è visibile in figura 34. Si espone un metodo per ogni possibile chiamata; ad ogni metodo occorre fornire come parametri l'url della richiesta, l'eventuale body e l'intestazione qualora dovesse servire. Questi dati vengono indicati sempre nello stesso formato, sarà poi il service a convertirli nel formato richiesto

dai singoli servizi invocati: server FHIR e web service di interfacciamento verso la blockchain.

```
export class HttpClientService {

  constructor(private httpClient: HttpClient) { }

  public getFromBlockchain(url: string){
    return this.httpClient.get<any>(url);
  }

  public postToBlockchain(url: string){
    return this.httpClient.post(url, {});
  }

  public getFromFHIR(url: string, headerParams: string[]){

    let headers = new HttpHeaders();
    headerParams.forEach((param: string) => {
      headers = headers.set(param.split(': ')[0].toString(), param.split(': ')[1].toString());
    });

    return this.httpClient.get<any>(url, { headers });
  }

  public postToFHIR(url: string, body: string, headerParams: string[]){

    let headers = new HttpHeaders();
    headerParams.forEach((param: string) => {
      headers = headers.set(param.split(': ')[0].toString(), param.split(': ')[1].toString());
    });

    return this.httpClient.post(url, body, {headers});
  }

  public putToFHIR(url: string, body: string, headerParams: string[]){

    let headers = new HttpHeaders();
    headerParams.forEach((param: string) => {
      headers = headers.set(param.split(': ')[0].toString(), param.split(': ')[1].toString());
    });

    return this.httpClient.put<any>(url, body, {headers});
  }

}
```

Figura 34: Codice Typescript implementazione servizio HttpClientService

5.7.3.2 Visualizzazione assistiti e livello di priorità

Una volta aperta l'applicazione, il professionista sanitario viene reindirizzato, attraverso un meccanismo basato sui moduli di routing dell'app e della tab di navigazione, nella pagina "Assistiti" visibile in figura 35. In questa pagina vengono visualizza-

ti tutti gli assistiti per cui il medico ha accettato una richiesta di collaborazione, pervenuta da altre strutture sanitarie.

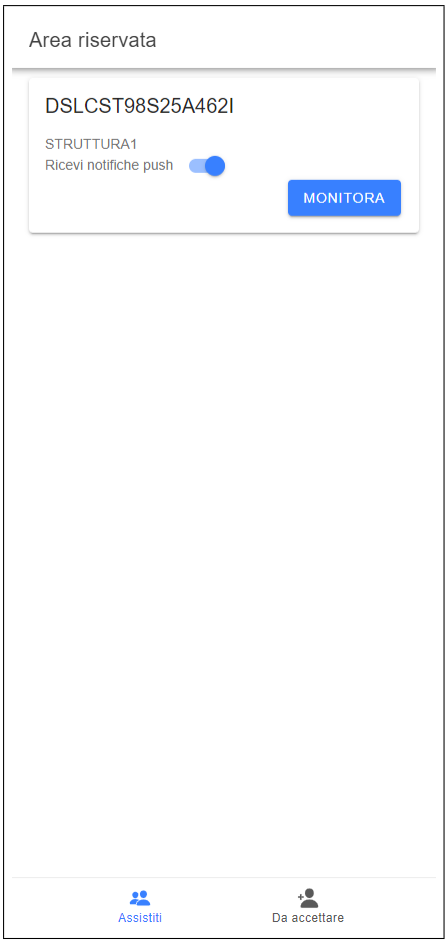

Figura 35: Aspetto della homepage dell'applicativo mobile per i medici

Le autorizzazioni vengono ottenute attraverso una richesta al web service di interfacciamento con la blockchain. Lo use case dettagliato generato all'apertura dell'applicativo mobile appena descritto è visibile in figura 36.

Altro processo invocato all'avvio dell'applicazione, di cui è visibile lo use case in figura 37, permette di inizializzare il servizio di notifiche, attraverso i seguenti passi:

- Controllo della piattaforma in cui è stata lanciata l'app; per l'avvio del processo è necessario che l'applicativo non sia stato avviato come web app.
- Ottenimento del token univoco associato al dispositivo in cui è stata avviata l'app.
- Ottenimento di tutte le autorizzazioni associate al medico (accettate o non).
- Per ogni autorizzazione

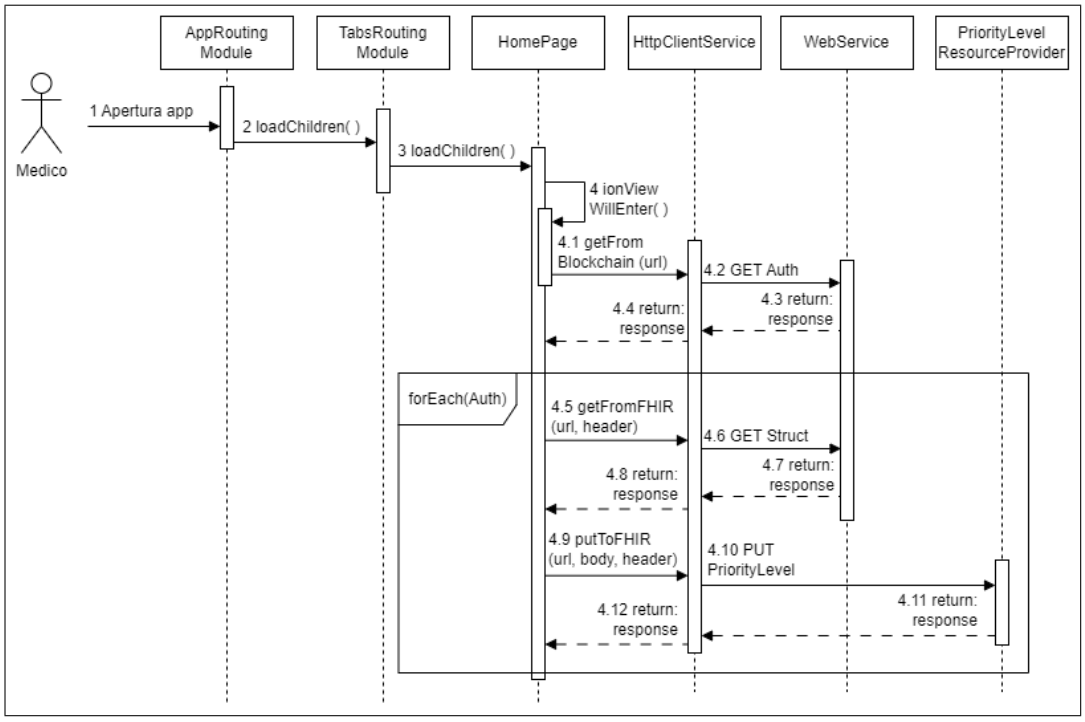

Figura 36: Use case apertura applicativo medico

- Recupero dell'end point del relativo server FHIR attraverso una chiamata alla blockchain.
- Invio di una chiamata POST al server contenente il token del dispositivo in modo da poter ricevere le notifiche nel caso in cui il server registri un'osservazione anomala legata al paziente protagonista dell'autorizzazione.

Un'ulteriore possibilità offerta al medico, all'interno della pagina "Assistiti" (si faccia riferimento alla figura 35), è quella di attivare o meno le notifiche per un determinato paziente cliccando sul *toggle* "Ricevi notifiche push".

All'interno del prototipo il livello di priorità esprimibile si esaurisce in due stati: attivato - non attivato. In una situazione reale è possibile prevedere più livelli di priorità differenziando la tipologie di notifiche inoltrate al medico.

All'apertura dell'app vengono settati i toggle associati ad ogni assistito ottenendo l'informazione del server FHIR della struttura sanitaria a cui si è rivolto il paziente.

Nel caso in cui il medico non avesse effettuato il login al dispositivo in cui è istallata l'app, perderebbe la notifica; una possibile miglioria potrebbe consistere

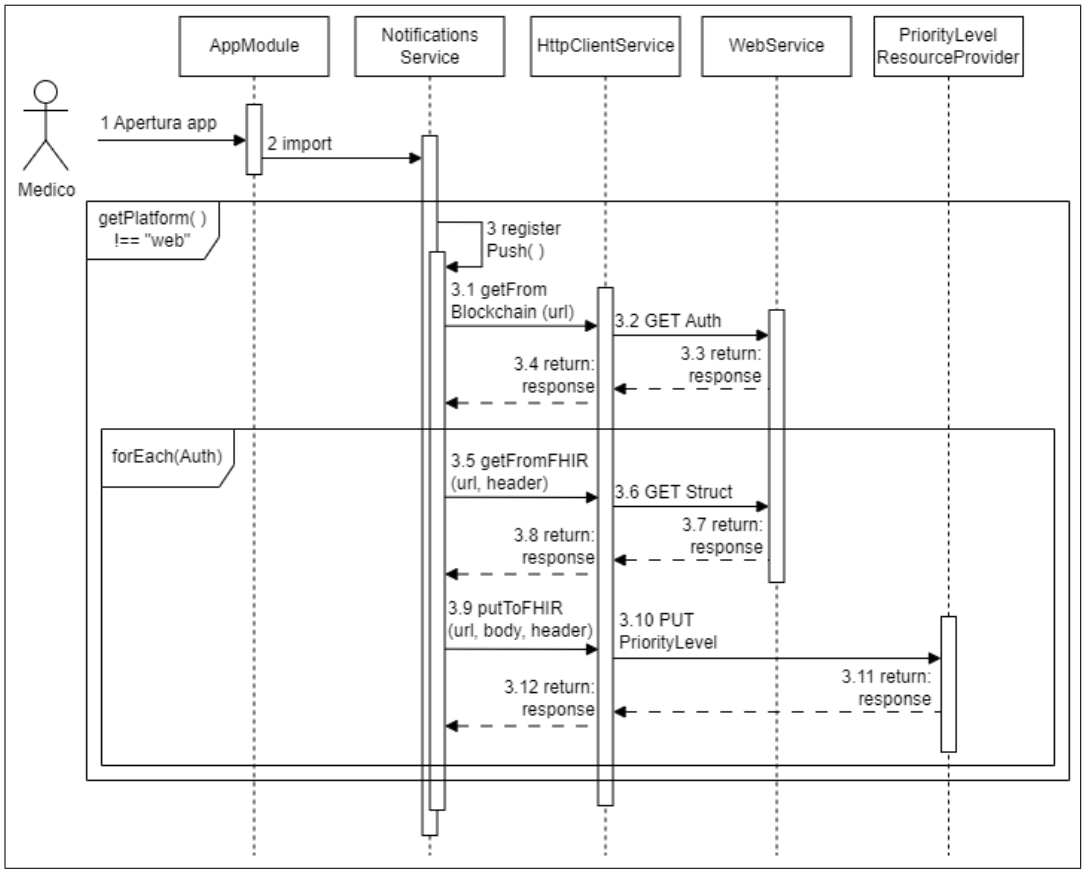

Figura 37: Use case inizializzazione notifiche applicativo medico

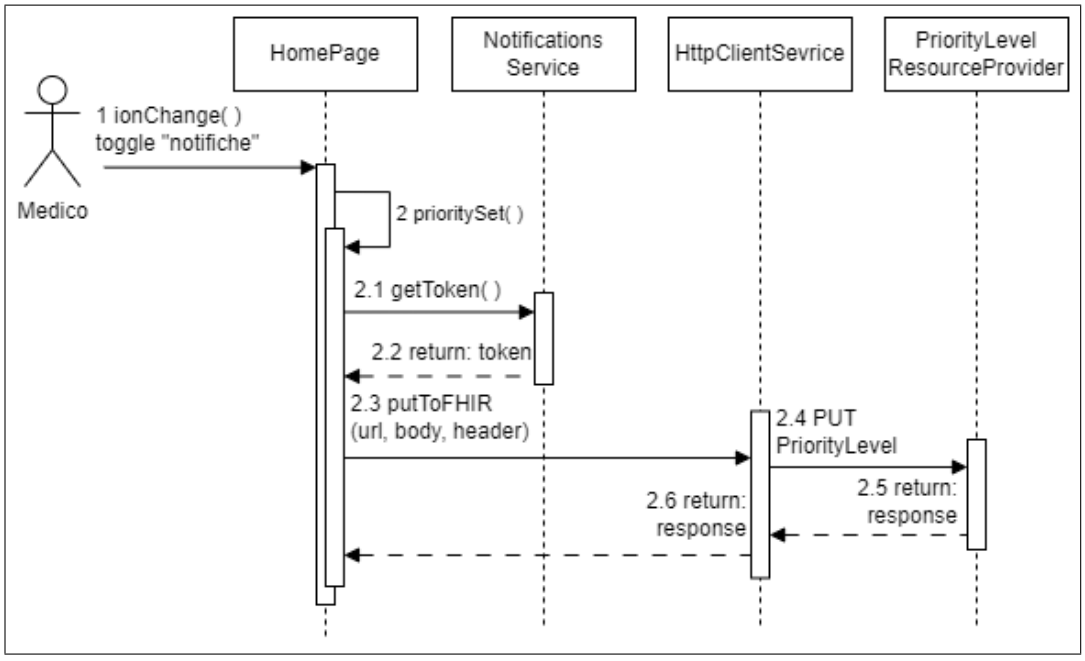

Figura 38: Use case impostazione livello di priorità applicativo medico

nel creare una tabella nel database in cui salvare tutte le notifiche con annesso il destinatario e lo stato (riportante un avvenuto invio o meno). Nel momento in cui il medico dovesse accedere ad un dispositivo riceverebbe in tal modo tutte le notifiche registrate nel periodo in cui non era registrato.

5.7.3.3 Monitoraggio assistiti

Cliccando sul pulsante "MONITORA" all'interno della pagina "Assistiti" visibile in figura 35, si giunge nella pagina di monitoraggio eventi (figura 39). In questa pagina sono visibili tutti gli eventi associati al paziente a cui il medico ha accesso.

Per ognuno di essi vengono visualizzate a schermo le seguenti informazioni:

- titolo dell'evento;
- data di esecuzione dell'evento.

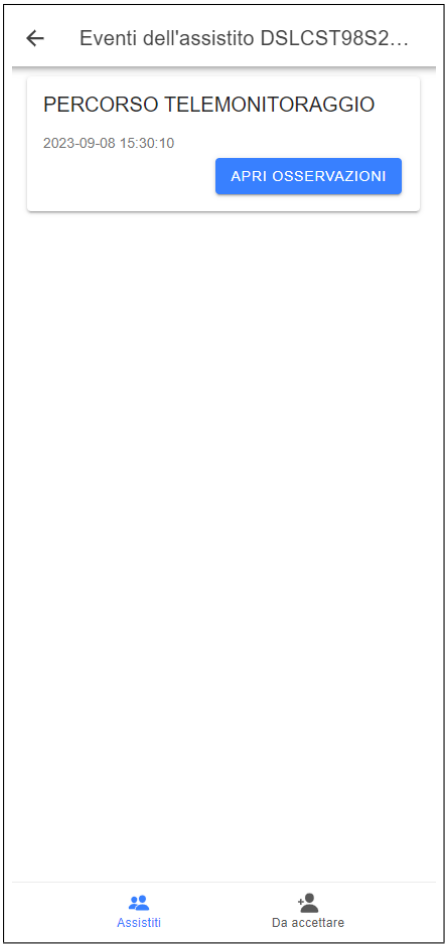

Figura 39: Visualizzare eventi di telemedicina nell'applicativo mobile per i medici

Cliccando sul pulsante "APRI OSSERVAZIONI" (figura 39) è possibile poi visualizzare tutte le osservazioni legate all'evento, attraverso la pagina mostrata in figura 40.

Le osservazioni nel'applicativo prototipale vengono visualizzate come una lista di card; in una soluzione più elaborata potrebbero essere organizzate per tipologia di osservazione, per tracciati grafici di andamento dei vari parametri o potrebbe essere fornita la possibilità di indicare un periodo temporale entro il quale visualizzare le informazioni.

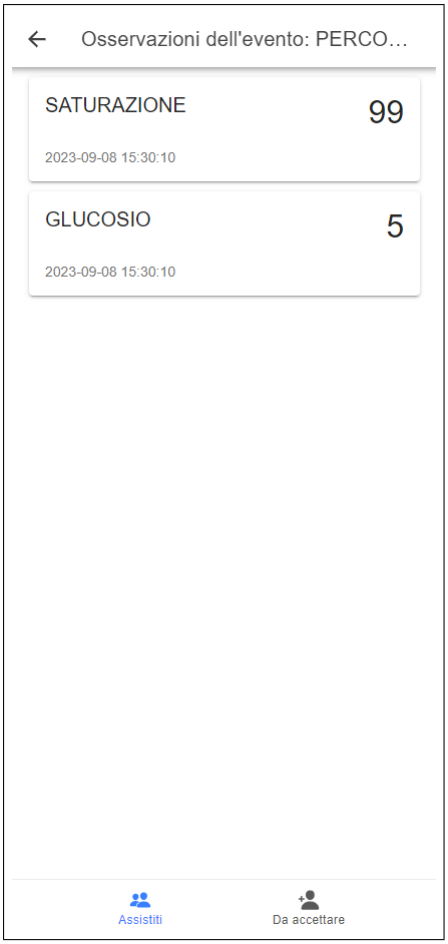

Figura 40: Visualizzare osservazioni nell'applicativo mobile per i medici

L'intero use case del processo di monitoraggio è visibile in figura 41.

5.7.3.4 Accettazione collaborazioni

L'ultima funzionalità offerta dall'applicativo permette al medico di accettare le richieste di collaborazione pervenute da altre strutture sanitarie.

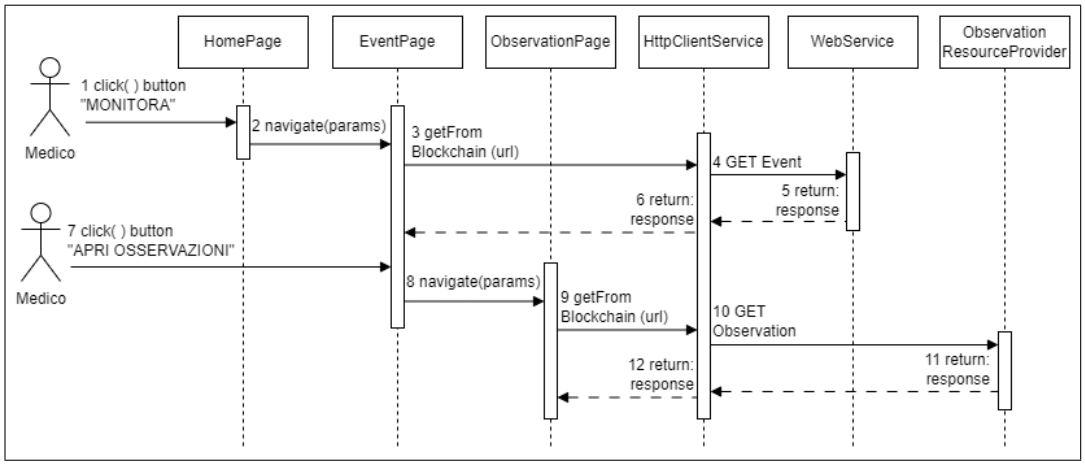

Figura 41: Use case monitoraggio assistito applicativo medico

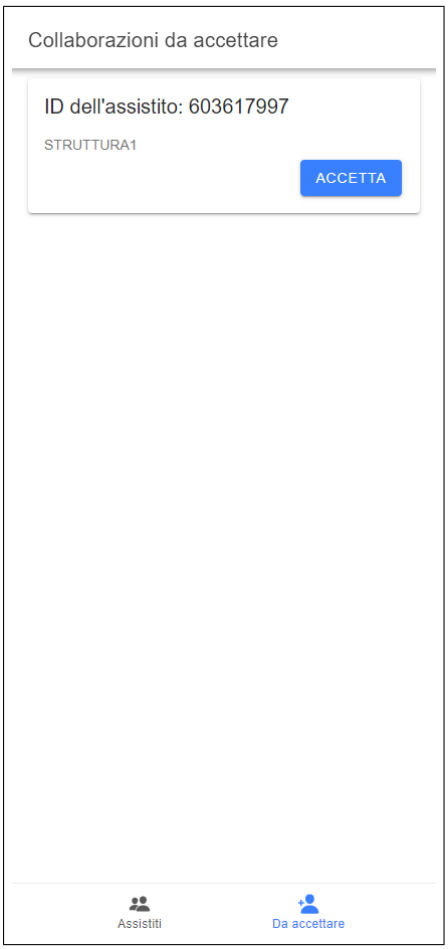

Figura 42: Accettare collaborazioni nell'applicativo mobile per i medici

Cliccando sul pulsante "Da accettare" presente nella tab di navigazione si giunge alla pagina che gestisce tale funzionalità (figura 42). Al suo interno vengono mostra-

te, nel caso fossero presenti, tutte le richieste di collaborazione non ancora accettate; si noti che, a differenza della schermata relativa alla pagina "Assistiti" vista in precedenza (figura 35), non è presente il codice fiscale dell'assistito nella relativa card: questo dato viene ottenuto al momento dell'accettazione della richiesta attraverso una chiamata autorizzata al server FHIR della struttura da cui ha avuto origine la collaborazione.

Il pulsante "ACCETTA", presente in ogni card associata ad un singolo assistito, dà la possibilità di confermare la collaborazione. Una volta accettata correttamente, la card non viene più visualizzata.

Lo use case completo della gestione delle autorizzazioni da accettare, dalla visualizzazione alla conferma, è visibile in figura 43. Esso consta di diversi passi:

- chiamata verso il webservice per modificare lo stato dell'autorizzazione della blockchain in "Accepted";
- chiamata al server FHIR della struttura che ha inviato la richiesta di collaborazione per ottenere le informazioni di cross necessarie a visualizzare correttamente i dati presenti nel registro della blockchain.

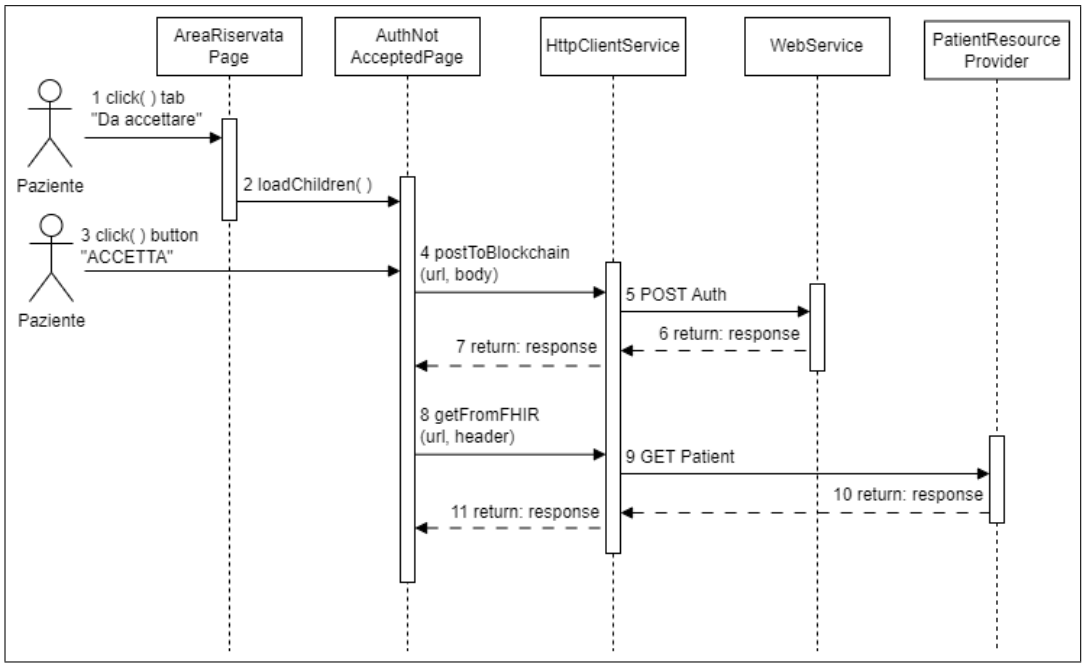

Figura 43: Use case accettazione collaborazione applicativo medico

5.8 Fase di test

In questa sezione è descritta la fase di test delle applicazioni sviluppate; essa ha come scopo quello di controllare ed aumentare la qualità del software e verificarne che il comportamento sia conforme con le linee guida definite durante la fase di analisi dei requisiti.

Sono stati eseguiti diversi test funzionali; oggetto di test sono state sia le singole componenti che il sistema nella sua interezza. Ognuno di essi è giunto a buon fine ottenendo il risultato aspettato in ciascun caso testato.

In tabella 11 sono visibili i test case di maggiore interesse e, per ciascuno di essi, il risultato atteso.

Tabella 11: Test case di maggiore interesse

Oggetto	**Test**	**Risultato atteso**
Blockchain	Gestione indirizzi smart contract	Corretta compilazione degli smart contract e restituzione dell'istanza corretta su richiesta del model
Blockchain	Indicizzazione nuovo evento	Mining del blocco contenente il token relativo all'evento
Blockchain	Indicizzazione di una nuova struttura sanitaria	Possibilità dagli altri nodi di accedere alle informazioni della nuova struttura
Blockchain	Rilascio token di autorizzazione	Rilascio di un token conforme ai soli aventi diritto
Blockchain	Accettazione della collaborazione	Set dell'attributo "accepted" del relativo token al valore "true"
Server FHIR	Validazione delle autorizzazioni	Ottenimento del token e corretta validazione
Server FHIR	Interfacciamento con il DB	Connessione con il database usando i parametri definiti nel file di configurazione
Interfaccia Javascript	Sanificazione degli input	Riconoscimento e segnalazione di input non validi
Web service	Invocazione metodi model	Restituzione degli stessi risultati ottenuti mediante interfaccia Javascript

Oggetto	**Test**	**Risultato atteso**
Intero backend	Accettazione autorizzazione	Conferma dell'accettazione e corretto trasferimento delle informazioni di Cross
Intero backend	Interazione con server FHIR di un'altra struttura	Ottenimento endpoint dalla blockchain e invio richiesta al server
Intero backend	Rimozione autorizzazione	Eliminazione del token dalla blockchain e dal server FHIR
App paziente	Login Google	Corretta autenticazione mediante OAuth 2
App paziente	Sincronizzazione con Google Fit	Ottenimento dei dati dell'utente e invio degli stessi al server FHIR
App medico	Impostare livello di priorità	Corretto invio dei dati al server comprensivi di token identificativo del dispositivo e livello di priorità
Intero sistema	Invio notifiche	Riconoscimeto delle osservazioni anomale e invio della notifica ai soli dispositivi che ne hanno diritto

Capitolo 6

Conclusioni e sviluppi futuri

L'obiettivo del progetto di tesi è stato quello di progettare e implementare un prototipo di una piattaforma di telemedicina distribuita, conforme con gli ultimi standard di interoperabilità e allineato alle direttive europee in termini di Fascicolo Sanitario Elettronico 2.0.

Definito ciò, la finalità ultima prevedeva anche la specializzazione di tale piattaforma per attività di telemonitoraggio attraverso l'implementazione di due applicativi mobile, uno dedicato alla gestione dell'utenza degli assistiti e uno volto all'utenza degli operatori sanitari, nelle specifico di medici.

La progettazione si è basata sui requisiti ottenuti eseguendo un'analisi di mercato, fondata anche sulle conoscenze di NBS s.r.l., azienda presso cui è stato svolto il tirocinio durante il quale è stato realizzato tale lavoro di tesi e presente nel settore della sanità da diversi anni.

Tra tutti i requisiti, quello che ha guidato maggiormente le scelte progettuali intraprese è stato la capacità del sistema di mantenere una forte modularità e interoperabilità verso le soluzioni già esistenti e soprattutto nei confronti della futura Piattaforma Nazionale di Telemedicina, ancora in fase di sviluppo.

Per garantire ciò, oltre a tutti gli altri requisiti, funzionali e non, è stata sviluppata una rete blockchain di tipo consortium costituita da tre nodi, ognuno assegnato ad una struttura sanitaria differente. Per il corretto funzionamento della rete sono stati implementati 3 smart contract che gestiscono, rispettivamente, i dati riguardanti le strutture sanitarie, le autorizzazioni e gli eventi di telemedicina da indicizzare nel registro.

Per la gestione e la trasmissione dei dati sanitari, allineandosi allo standard FHIR di HL7 è stato sviluppato un server FHIR, poi replicato in tutti i nodi della rete e

affiancato da un database relazionale SQL.

Per garantire una corretta interazione dell'utente con il sistema è stata implementata un'interfaccia JavaScript da riga di comando, in un contesto reale associabile ad un applicativo desktop, che permette di compiere tutte le operazioni necessarie alla gestione degli eventi di telemedicina:

- Inserimento e ricerca di un nuovo paziente;
- Inserimento e ricerca di un evento di telemedicina;
- Inserimento di una nuova osservazione sanitaria;
- Visualizzazione dei dati sanitari di un assistito al solo personale autorizzato.

Lato mobile sono state create due applicazioni volte alla gestione del telemonitoraggio, come da specifiche iniziali. Esse permettono una fruizione più comoda e dinamica dei dati dei pazienti per i medici e implementano un ponte di normalizzazione per tutti i dati rilevati dai vari sensori presenti all'interno del contesto domestico presso il quale l'assistito è seguito nel suo percorso di monitoraggio. Entrambe le app si interfacciano con il sistema appena descritto.

Si ritiene quindi che gli obiettivi e le caratteristiche delle applicazioni definite all'inizio del progetto siano stati rispettati e soddisfatti, il tutto tenendo conto dei limiti che un prototipo può presentare. In ottica di un'implementazione reale è possibile individuare una serie di sviluppi per rendere il sistema più puntuale, efficace ed efficiente.

Attualmente, i token di autorizzazione, per semplicità, hanno come oggetto un medico e tutte le osservazioni di un dato paziente presso una struttura sanitaria. Una possibile miglioria sta nello specializzare tali permessi ai dati relativi ad una certa categoria di osservazioni.

Oltre a ciò, l'uso di databse SQL poco si adatta alla tipologia delle risorse FHIR, che possono essere gestite in JSON o XML (JSON per questo progetto). Sebbene la maggior parte delle strutture sanitarie adotta delle basi di dati di tipo relazionale (motivo per cui si è deciso di mappare le informazioni in tal senso), sarebbe interessante prevedere delle basi di dati di tipo NoSQL Document store: in tal modo il sistema ne risulterebbe snellito in quanto i dati ottenuti dalle chiamate non andrebbero convertiti ma posti direttamente all'interno delle relative collection.

Si potrebbero implementare, inoltre, delle funzionalità nell'interfaccia che implementino l'update dei dati; il backend è già predisposto in tal senso.

Un'ulteriore possibilità di crescita si può individuare nel miglioramento della visualizzazione dei dati, mostrando dei grafici sull'andamento delle varie misurazioni e mostrando i valori di picco dei vari trend.

La libreria HAPI FHIR offre la possibilità di strutturare le API con un meccanismo di paging ottenendo le sole porzioni di dati che si intende visualizzare. Considerando che in un contesto reale i dati da visualizzare potrebbero essere di dimensione considerevole, questa implementazione permetterebbe di rendere più agevole il sistema.

Attualmente l'applicativo mobile si interfaccia solo con i servizi di Google Fit al fine di ottenere le misurazioni dei parametri da monitorare; si potrebbero esporre dei connettori verso le varie istanze di device fisici di monitoraggio come i monitor multiparametrici.

Infine, potrebbe essere interessante sviluppare una versione per smartwatch certificati CE come *dispositivi medici software* per entrambi gli applicativi:

- lato assistito si potrebbero sfruttare le rilevazioni che il dispositivo permette di effettuare;
- lato medico si potrebbe raggiungere un nuovo livello di flessibilità nella ricezione delle notifiche e nella visualizzazione delle informazioni dei pazienti.

Bibliografia

[1] URL: https://ilbolive.unipd.it/it/news/telemedicina-spinta-dovuta-pandemia (visitato il 09/2023).

[2] Fabrizio Massimo Ferrara. "Telemedicina. Dagli scenari attuali agli obiettivi del Pnrr, la survey Altems sulle soluzioni sviluppate dalle aziende sanitarie". In: *Quotidiano sanità* (2022).

[3] URL: https://www.agenas.gov.it/comunicazione/primo-piano/2090-piattaforma-telemedicina-fse (visitato il 07/2023).

[4] URL: https://www.salute.gov.it/portale/ehealth/dettaglioContenutiEHealth.jsp?lingua=italiano&id=5491&area=eHealth&menu=fse (visitato il 07/2023).

[5] URL: https://www.healthtech360.it/salute-digitale/cartella-clinica-elettronica-emr-ehr-fse-differenze/ (visitato il 07/2023).

[6] URL: https://www.tibco.com/it/reference-center/what-is-hl7-fhir (visitato il 07/2023).

[7] URL: https://www.salute.gov.it/portale/ehealth/dettaglioContenutiEHealth.jsp?lingua=italiano&id=5524&area=eHealth&menu=telemedicinae (visitato il 07/2023).

[8] Ministero della Salute. *Gazzetta Ufficiale della Repubblica Italiana*. Decreto 29/04/2022, allegato "Linee Guida".

[9] URL: https://github.com/ministero-salute/it-fse-support/tree/main/doc/telemedicina (visitato il 08/2023).

[10] URL: https://digitalhealthitalia.com/pnt-italia-la-nuova-societa-che-realizzera-la-piattaforma-nazionale-di-telemedicina/ (visitato il 08/2023).

[11] Mannaro K - Baralla G - Pinna A - Ibba S. *A Blockchain Approach Applied to a Teledermatology Platform in the Sardinian Region (Italy).* 23/02/2018.

[12] Gupta R - Shukla A - Tanwar S. *AaYusH: A Smart Contract-based Telesurgery System for Healthcare 4.0.* IEEE International Conference on Communications Workshops (ICC Workshops), Dublin, Ireland, 11/07/2020.

[13] *MedicalChain whitepaper 2.1.* Copyright © 2018 Medicalchain.

[14] P. Zhanga - D. C. Schmidta - G. Lenzb - S. T.Rosenbloomc. *FHIRChain: Applying Blockchain to Securely and Scalably Share Clinical Data.* 01/01/2018.

[15] URL: https://www.agendadigitale.eu/sanita/fascicolo-sanitario-elettronico-2-0-cosi-sara-vera-svolta-per-sanita-e-cittadini/ (visitato il 08/2023).

[16] Ente Italiano di Normalizzazione. *UNI EN ISO 9241-971:2022.* 16/06/2022.

[17] URL: https://www.consulcesi.it/news/blockchain-sanita-cartella-clinica (visitato il 07/2023).

[18] URL: https://www.ibm.com/it-it/topics/docker (visitato il 07/2023).

[19] URL: https://kinsta.com/it/knowledgebase/node-js/ (visitato il 07/2023).

[20] URL: https://kinsta.com/it/knowledgebase/cosa-e-github/ (visitato il 07/2023).

[21] *Documentazione HAPI FHIR.* Ver. 6.6.0. 1 Ago. 2023.

[22] URL: https://www.hl7.org/fhir/ (visitato il 07/2023).

[23] URL: https://aws.amazon.com/it/what-is/java/ (visitato il 07/2023).

[24] URL: https://www.redhat.com/it/topics/middleware/what-is-ide (visitato il 08/2023).

[25] URL: https://www.ilsoftware.it/focus/visual-studio-code-cos-e-e-come-funziona_19189/ (visitato il 08/2023).

[26] URL: https://tecnologia.libero.it/android-studio-cose-come-usarlo-47012 (visitato il 08/2023).

[27] URL: https://www.geekandjob.com/wiki/ionic (visitato il 09/2023).

[28] URL: https://www.geekandjob.com/wiki/capacitor (visitato il 09/2023).

[29] URL: https://aulab.it/notizia/306/angular-al-microscopio (visitato il 09/2023).

[30] URL: https://www.geekandjob.com/wiki/express (visitato il 09/2023).

[31] URL: https://blog.desdelinux.net/it/dbeaver-una-excelente-herramienta-para-la-gestion-de-diferentes-db/ (visitato il 09/2023).

[32] URL: https://www.geekandjob.com/wiki/postgresql (visitato il 09/2023).

[33] URL: https://www.geekandjob.com/wiki/firebase (visitato il 09/2023).

[34] URL: https://www.html.it/pag/60329/direttive-in-angular-2/ (visitato il 09/2023).

Ringraziamenti

Mi risulta difficile trovare delle parole che possano esprimere ciò che ho dentro al termine di questo percorso, al termine di questa fase della mia vita che, dopo cinque anni, è giunta alla sua conclusione.

Cinque anni

Potrei dire che il tempo in questo periodo sia volato, che sembra essere stata ieri quella mattina in cui, per la prima volta, salivo sul treno per andare in facoltà a frequentare i precorsi. Potrei dire che sembra essere passato solo qualche giorno dal primo esame, da quel 30 in fisica 1 e anche dalla prima bocciatura. Potrei dirlo, già, ma non è così.

Non è così perché sento sulle spalle tutte le ore di studio, tutte le volte che ho visto l'alba dal finestrino del treno, tutte le giornate in cui entravo in facoltà di giorno e ne uscivo che ormai il sole era calato, tutte le corse in strada per non perdere il pullman, tutti i corsi frequentati, tutte le notti passate sui libri, tutte le penne e gli evidenziatori scaricati.

Di fianco a tutte le fatiche e le delusioni, però, voltandomi indietro, vedo anche le soddisfazioni, le gioie nel prendere una lode, le risate nei corridoi, le persone che in questo percorso mi hanno accompagnato, mi hanno sostenuto, mi hanno incoraggiato; a tutti voi va la mia gratitudine!

Ringrazio il mio relatore, il professor Alessandro Cucchiarelli. Grazie per la sua disponibilità, per la sua dedizione al suo lavoro e per la particolare attenzione che mi ha dedicato in questo percorso.

Ringrazio la NBS, per aver creduto in me dal primo colloquio e per avermi dato l'opportunità di lavorare facendo ciò per cui in questi anni mi sono formato: ciò che mi piace fare. Sento di ringraziare, quindi, i miei tutor aziendali ed oramai colleghi,

per avermi aiutato, spronato e fatto vivere in un ambiente sereno durante il tirocinio.

Ringrazio i miei genitori per non avermi mai fatto mancare il loro amore nonostante tutte le avversità della vita.

Ringrazio i miei nonni, per avermi sempre incoraggiato facendomi sentire il loro ingegnere ancor prima di aver superato il primo esame della triennale.

Ringrazio il resto della mia famiglia per aver fatto il tifo per me e per la vostra presenza discreta ma costante.

Ringrazio la mia ragazza Chiara, per esserci sempre stata e perché ancora oggi dopo 9 anni continui ad accogliere ogni lato di me. Grazie per avermi insegnato a non mollare anche quando è tutto buio intorno e per essere al mio fianco in questa vita per cercare insieme l'interruttore della luce.

Ringrazio la tua famiglia per avermi accolto da subito come se facessi parte da sempre di essa.

Ringrazio mio fratello Giordano per essere stato al mio fianco in questi lunghi anni, dentro e fuori la facoltà. So che per qualsiasi cosa potrò contare su di te ma ricorda che è un gioco a cui si gioca in due.

Ringrazio Mattia, per aver condiviso questi anni di studio con me, rendendo di certo più leggera l'implementazione degli innumerevoli progetti svolti insieme.

Ringrazio tutti i miei amici, pochi ma buoni. Grazie per tutte le risate vissute insieme in questi anni, soprattutto per quelle esagerate, forzate e fatte per mascherare o superare i momenti difficili che abbiamo vissuto.

Ringrazio i ragazzi e le ragazze del mio reparto per condividere il loro percorso con me e per riempire ogni giorno il mio zaino con le avventure vissute insieme. Sono entrato in comunità capi per restituire un po' di quello che avevo ricevuto ma ogni giorno mi rendo conto che ciò che mi trasmettete continua ad essere molto più grande di ciò che io riesco a darvi.

Ringrazio la comunità capi del gruppo scout Folignano 1, la mia seconda famiglia. Grazie per avermi fatto vivere in una famiglia felice, per avermi fatto assaporare la bellezza dell'avventura, per avermi fatto macinare chilometri di strada di fianco a dei compagni fidati con null'altro che uno zaino sulle spalle. Grazie, quindi, per aver fatto di me un uomo della partenza e per darmi l'occasione di viverla ogni giorno nel servizio che insieme portiamo avanti.

Ringrazio il mio staff, perché rendete questo servizio meno stancante tirandomi su e consigliandomi nei momenti di sconforto che insieme abbiamo imparato ad affrontare.

Infine, non posso che non ringrazio me stesso, per aver dedicato con pazienza e sacrificio questi cinque anni allo studio. Non è sempre stato semplice, non sempre è stato bello. Ma se c'è una cosa che ho imparato è che occorre affrontare la salita per godere della vetta e, soprattutto, che se il sentiero sembra semplice, in fondo, è quello sbagliato.