

Esame di
Sistemi Multimediali
prof. Giovanni Dimauro

Fascicolo Sanitario Elettronico Multimediale
Realizzazione della sezione "Indagini diagnostiche"

Supervisione: dott. Francesco Girardi

Studente: Giuseppe Perniola, mat. 542844

Introduzione.....	2
Presentazione della sezione indagini.....	3
Diario indagini diagnostiche.....	4
Form di inserimento nuova indagine.....	5
Centri indagini diagnostiche.....	8
Accesso dalla sezione diagnosi.....	9
Modifiche al database.....	10
Modifiche al file utility.php.....	12
Modifiche alla sezione files.....	14
template_page_files.php.....	15
configFiles.php.....	15
funzioniFiles.php.....	16
uploadFiles.php.....	16
Indagini.php.....	16
Recupero dati per costruzione tabelle indagine.....	17
Recupero dati per costruzione form e tabella centro indagini.....	18
Template_page_indagini.php.....	19
Controllo dell'accesso.....	20
Recupero dei dati dalla pagina indagini.php.....	20
Costruzione della pagina html.....	22
Indagini.js.....	27
Funzioni di validazione dei form.....	28
La funzione document.ready e funzioni sui button dei form.....	30
NuovaIndagine.php.....	33
ModificaIndagine.php.....	34
EliminaIndagine.php.....	35

INTRODUZIONE

Per lo sviluppo di questo modulo ho cercato di seguire la stessa metodologia e approccio che gli altri colleghi hanno utilizzato per sviluppare il resto del sistema. In particolare, data la forte relazione con il modulo "Diagnosi", ho cercato di seguire le stesse tecniche nello sviluppo del codice.

Ho utilizzato quindi il linguaggio php per quanto riguarda lo scripting server-side e la generazione di pagine dinamiche.

Ho utilizzato il linguaggio javascript e la libreria jQuery per la manipolazione e gestione degli elementi della pagina html.

Per la costruzione degli elementi html come moduli, bottoni e vari elementi grafici ho utilizzato il framework [Bootstrap](#). Inoltre per la gestione della data all'interno del form ho utilizzato [Bootstrap-datetimepicker](#), un plugin che utilizza Bootstrap e jQuery per creare un widget calendario che permette all'utente di selezionare data e ora.

Come ambiente di sviluppo, oltre a WAMP, ho utilizzato [PhpStorm](#) in quanto mi ha permesso di sviluppare codice sia in php che in javascript, fornendomi dei tool per gestire e modificare il database direttamente dall'IDE.

Come ulteriore strumento di debugging oltre a quelli forniti da WAMP e PhpStorm ho utilizzato i tool messi a disposizione da Google: [Chrome DevTools](#).


Infine ho utilizzato [Git](#) come software di controllo versione distribuito per permettermi di sviluppare il progetto da macchine differenti, tenere traccia del lavoro svolto e delle funzionalità da sviluppare.


PRESENTAZIONE DELLA SEZIONE INDAGINI

E' possibile accedere alla sezione indagini diagnostiche cliccando sulla voce "Indagini Diagnostiche" presente nel menù a sinistra nella pagina principale dell'applicazione. Si aprirà la seguente pagina:

Indagini diagnostiche

In questa pagina è possibile visualizzare tutti gli esami che un paziente deve effettuare o ha già effettuato e l'elenco di tutti gli studi e laboratori dove è possibile effettuare un determinato esame.

 **Diario indagini diagnostiche**





 **Centri indagini diagnostiche**

Nuova indagine

Concludi indagine

Annulla indagine

Indagini Richieste

Indagine	Motivo	Care provider	Opzioni
Terza indagine	Infezione delle vie urinarie	Letizia Ercolino	 
Quarta indagine	motivazione personalizzata 4	Bob Kelso	 

Indagini Programmate


Indagine	Motivo	Care provider	Data	Centro	Opzioni
----------	--------	---------------	------	--------	---------

La pagina sarà già aperta sulla sezione "Diario indagini diagnostiche", cliccando su "centri indagini diagnostiche", in alto, verrà portata in primo piano la tabella con la lista dei centri registrati al fascicolo

Diario indagini diagnostiche

La sezione "Diario indagini diagnostiche" è composta da tre pulsanti per l'inserimento di una nuova indagine e da tre tabelle che mostrano le indagini del paziente suddivise in indagini richieste, programmate e completate.

NOTA IMPORTANTE: Nel caso sia un careprovider ad accedere alla pagina, le indagini collegate ad una diagnosi non accessibile al careprovider (per via del livello di confidenzialità) non verranno mostrate nelle tabelle.

Indagini Richieste			
Indagine	Motivo	Care provider	Opzioni
Terza indagine	Infezione delle vie urinarie	Letizia Ercolino	 
Quarta indagine	motivazione personalizzata 4	Bob Kelso	 

Nella tabella delle indagini richieste viene riportato il nome dell'indagine, la motivazione (o la diagnosi ad essa collegata) ed il careprovider che ha richiesto l'indagine.

Indagini Programmate					
Indagine	Motivo	Care provider	Data	Centro	Opzioni
Prima indagine	Infezione delle vie urinarie	Letizia Ercolino	25/03/2017 17:30	Laboratorio analisi Biallo via Sparano 1 BARI	 

Nella tabella delle indagini programmate viene anche riportata la data in cui verrà effettuata l'indagine e il nome e l'indirizzo del centro in cui si svolgerà.

Indagini completate						
Indagine	Motivo	Care provider	Data	Referto	Allegati	Opzioni
Quinta indagine	motivazione 5	Dr. Strange	01/03/2017 16:00			 
Seconda indagine	ulcera	Bob Kelso	02/03/2017 18:00			 

Nella tabella delle indagini completate vengono riportati anche due tasti che permettono di visualizzare allegati e referto collegati all'indagine (se sono stati inseriti).

Il pulsante verde *"modifica"* provvede ad aprire un form sotto l'indagine selezionata i cui campi saranno già precompilati con le informazioni dell'indagine. I form di modifica sono identici al form di inserimento di una nuova indagine, di cui se ne fornisce una descrizione più avanti. Il pulsante rosso *"elimina"* provvede ad eliminare l'indagine selezionata dal fascicolo.

Form di inserimento nuova indagine

Una volta selezionata la sezione *"Diario indagini diagnostiche"*, i tre pulsanti blu in alto permettono all'utente di creare una nuova indagine.



Cliccando su *"Nuova indagine"* si aprirà un form in cui inserire i dati, gli ultimi due tasti si attiveranno solo dopo aver premuto *"Nuova indagine"*.

Il numero di campi del form varia in base allo stato dell'indagine da inserire. I campi contrassegnati con un asterisco sono obbligatori:

Indagine richiesta

Tipo indagine *	<input type="text"/>
Motivo *	Selezionare una motivazione.. ▼
Care provider *	Selezionare una careprovider.. ▼
Stato *	Richiesta ▼

Indagine programmata

Tipo indagine *	<input type="text"/>
Motivo *	Selezionare una motivazione.. ▼
Care provider *	Selezionare una careprovider.. ▼
Stato *	Programmata ▼
Centro *	Selezionare un centro.. ▼
Data*	Selezionare una data..

Indagine completata

Tipo indagine *	<input type="text"/>
Motivo *	Selezionare una motivazione.. ▼
Care provider *	Selezionare una careprovider.. ▼
Stato *	Completata ▼
Centro *	Selezionare un centro.. ▼
Data*	Selezionare una data..
Referto	Selezionare un file.. ▼
Allegato	Selezionare un file.. ▼

Attenzione: Per selezionare un file come referto o allegato è necessario caricarlo preventivamente nella sezione **Files**.

Una volta inseriti i dati, per procedere occorre premere il pulsante blu *"Concludi indagine"*. Se invece si vuole annullare l'inserimento è necessario premere il tasto *"Annulla indagine"*.

Se l'utente non compila correttamente i campi, verrà avisato da un messaggio di errore quando si premerà il tasto *"Concludi indagine"*:

Attenzione: Compilare correttamente i campi bordati in rosso.

Tipo indagine *	Indagine di prova
Motivo *	Selezionare una motivazione.. ▼
Care provider *	Francesco Girardi ▼
Stato *	Programmata ▼
Centro *	Laboratorio analisi Biallo, BARI ▼
Data*	Selezionare una data..

Passiamo ad analizzare i campi del form:

- **Tipo indagine** è un campo di tipo testuale che accetta una stringa di testo come nome per l'indagine.
- **Motivo** contiene la motivazione per cui l'indagine viene svolta ed è un un menù a tendina contenente la lista delle diagnosi del paziente più il valore "Altra motivazione" che se scelto, attiverà un ulteriore campo testuale dove sarà possibile inserire una stringa di testo come motivazione personalizzata.








NOTA IMPORTANTE: Le diagnosi possiedono un livello di confidenzialità, pertanto se non è il paziente ma un careprovider ad inserire la nuova indagine, nel menù a tendina le diagnosi a cui non ha accesso non verranno visualizzate.




- **Care provider** permette di inserire il nome del careprovider che ha richiesto l'indagine. E' un menù a tendina contenente la lista dei careprovider collegati al paziente, più la voce "Nuovo careprovider". Selezionando quest'ultima opzione si visualizzerà un campo di testo in cui sarà possibile inserire nome e cognome di un careprovider non registrato al fascicolo. Se invece è un careprovider ad inserire la nuova indagine, il suo nome sarà automaticamente visualizzato all'interno di questo campo.
- **Stato** è una lista contenente i valori "richiesta", "programmata" e "conclusa".
- **Centro** è una lista da cui è possibile scegliere dove effettuare l'indagine tra tutti i centri diagnostici registrati al fascicolo.
- **Data** permette di visualizzare un widget calendario da cui è possibile scegliere sia il giorno che l'orario in cui effettuare l'indagine.
- **Referto** e **Allegato** permettono di selezionare dei file da collegare all'indagine. I file visualizzati all'interno di queste liste devono essere caricati preventivamente dalla sezione "Files" nelle categorie "Referti indagini diagnostiche" e "Allegati indagini diagnostiche".








NOTA IMPORTANTE: Anche i files possiedono un livello di confidenzialità, pertanto se è un careprovider ad inserire la nuova indagine, non visualizzerà i files a cui non ha accesso.

Centri indagini diagnostiche

Cliccando su *"Centri indagini diagnostiche"* portiamo in primo piano le tabelle contenenti tutti i centri indagini registrati al fascicolo, divisi per studi specialistici, studi radiologici e laboratori di analisi.

Studi Specialistici				
Studio	Sede	Contatti	Mail	Messaggio FSEM
FG Ematologo	via Amendola 79 BARI	 3472331738  080131313	 f.girardi6293@gmail.com	 Francesco Girardi
Centro Curatutto	via Panacea 13 Castelluccio (FG)	 131313133	 utente@fsem.eu	 Bob Kelso

Studi Radiologici				
Studio	Sede	Contatti	Mail	Messaggio FSEM
Studio Radiologico Mangialardi	via Risorgimento 53 Adelfia (BA)	 360995972	 mangia.lardi@gmail.com	 Letizia Ercolino

Laboratori Analisi				
Studio	Sede	Contatti	Mail	Messaggio FSEM
Laboratorio analisi Biallo	via Sparano 1 BARI	 320292923  080404550	 biallo@gmail.com	 Gustavo BOCCIA
Laboratorio analisi cliniche Dott.ssa Leopizzi	via Taranto 34 LECCE	 33376445	 leopizzi@gmail.com	 Rosanna Maggio

Notiamo che le tabelle contengono tutte gli stessi campi:

- **Studio** è il nome del centro indagini.
- **Sede** è il suo indirizzo.
- **Contatti** è una lista di numeri di telefono per contattare il centro. I numeri di telefono sono cliccabili e vanno ad aprire l'applicazione predefinita per avviare chiamate del dispositivo dal quale è connesso il paziente (ad esempio: Skype).
- **Mail** contiene l'indirizzo email del centro, cliccandoci è possibile inviare una email.
- **Messaggio FSEM** apre un pop-up per inviare un messaggio privato al careprovider responsabile del centro indagini, specificandone come destinatario il suo nome.

ACCESSO DALLA SEZIONE DIAGNOSI


Dalla sezione *"Diagnosi"* è possibile accedere alle indagini che fanno riferimento ad una data diagnosi cliccando sull'apposito pulsante:


Escluse				
Diagnosi	Ultimo aggiornamento	Visibilità	Care provider	Opzioni
Infezione delle vie urinarie	23-04-2016	5	Girardi (Esclusa) Girardi (Sospetta)	  


Una volta cliccato il pulsante, si viene reindirizzati alla pagina *"Indagini"* che però mostrerà solamente le indagini collegate alla diagnosi selezionata.


Indagini diagnostiche


In questa pagina è possibile visualizzare tutti gli esami che un paziente deve effettuare o ha già effettuato e l'elenco di tutti gli studi e laboratori dove è possibile effettuare un determinato esame.

 **Diario indagini diagnostiche**

 **Centri indagini diagnostiche**



 Nuova indagine

 Concludi indagine

 Annulla indagine

Indagini relative alla diagnosi **Infezione delle vie urinarie** del **23-04-2016**

Indagini Richieste

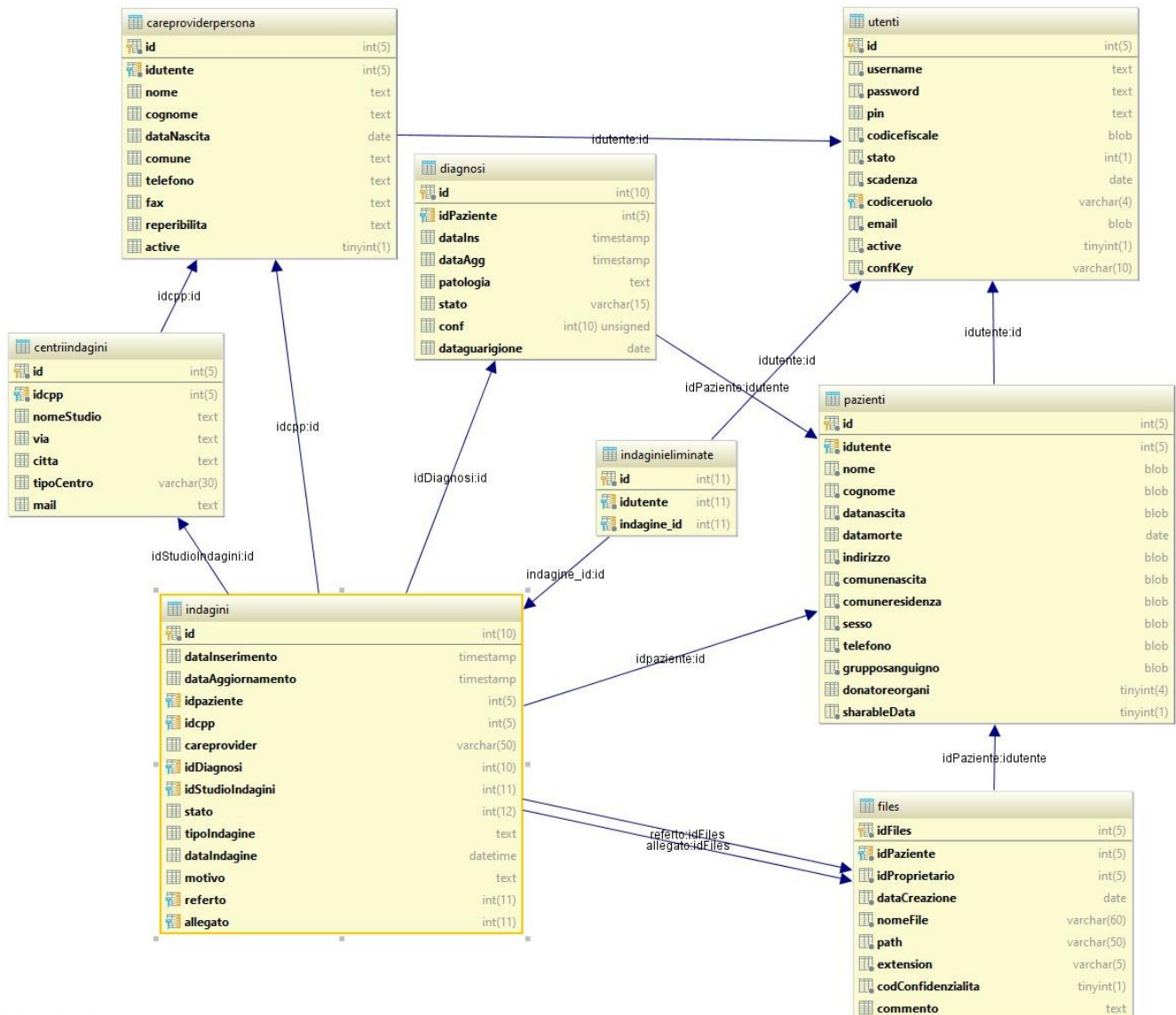
Indagine	Motivo	Care provider	Opzioni
Terza indagine	Infezione delle vie urinarie	Letizia Ercolino	 

Indagini Programmate

Indagine	Motivo	Care provider	Data	Centro	Opzioni
Prima indagine	Infezione delle vie urinarie	Letizia Ercolino	25/03/2017 17:30	Laboratorio analisi Biallo via Sparano 1 BARI	 

MODIFICHE AL DATABASE

Il database è stato modificato per adeguarsi allo sviluppo della sezione indagini, in particolare è stata modificata la tabella "indagini" aggiungendo nuovi campi e ridefinendo le relazioni con le altre tabelle. Inoltre è stata aggiunta una nuova tabella: "indaginieliminate".



Powered by yFiles

Analizziamo campi e le relazioni della tabella indagini:

- **id(int, chiave primaria)**: è l'indice che identifica univocamente ogni indagine.
- **tipoIndagine(text)**: Contiene il nome dell'indagine.
- **stato(int)**: Rappresenta lo stato dell'indagine: 0 per "Indagine richiesta", 1 per "Indagine programmata", 2 per "Indagine completata" e 3 per "Indagine eliminata".
- **dataInserimento(timestamp)**: la data in cui il record indagine è stato creato.

- **dataAggiornamento(timestamp)**: la data in cui è stata registrata l'ultima modifica del record indagine. Cambia ogni volta che viene eseguita una query di tipo UPDATE sul record in questione.
- **dataIndagine(datetime)**: Contiene giorno e orario in cui l'indagine verrà o è stata effettuata, se l'indagine è programmata o completata. Nel caso l'indagine sia ancora in stato di richiesta e non si è ancora a conoscenza di quando sarà svolta questo campo sarà impostato a *NULL*.
- **idpaziente(int, chiave esterna)**: rappresenta l'id del paziente che deve effettuare l'indagine. Fa riferimento alla tabella **pazienti.id**.
- **idcpp(int, chiave esterna)**: rappresenta l'id del careprovider che ha ordinato l'indagine (se il careprovider è registrato al fascicolo, altrimenti è uguale a *NULL*). Fa riferimento alla tabella **careproviderpersona.id**.
- **careprovider(text)**: Contiene nome e cognome del careprovider che ha ordinato l'indagine, se il careprovider è registrato al fascicolo questo campo conterrà il nome del careprovider puntato da **idcpp**. Se invece il careprovider non è registrato, **idcpp** sarà *NULL* e questo campo conterrà il nome di un careprovider inserito dall'utente al momento della creazione della nuova indagine.
- **idDiagnosi(int, chiave esterna)**: Rappresenta la diagnosi a cui fa riferimento l'indagine, se l'indagine non è collegata a nessuna diagnosi allora questo campo sarà *NULL*.
- **motivo(text)**: Contiene la motivazione per la quale è stata richiesta una indagine, se l'indagine è stata richiesta a causa di una diagnosi questo campo conterrà il nome della diagnosi puntata da **idDiagnosi**. Altrimenti se l'indagine non è collegata ad una diagnosi, **idDiagnosi** sarà *NULL* mentre questo campo conterrà la motivazione inserita dall'utente al momento della creazione della nuova indagine.
- **idStudiIndagini(int, chiave esterna)**: Rappresenta l'id del centro in cui si svolgerà o si è svolta l'indagine, se l'indagine è programmata o completata. Nel caso l'indagine sia ancora in stato di richiesta e non si è a conoscenza di dove sarà svolta questo campo sarà impostato a *NULL*.
- **idReferto(int, chiave esterna)**: Rappresenta il file che verrà visualizzato come referto dell'indagine. Fa riferimento a **files.idFiles**.
- **idAllegato(int, chiave esterna)**: Rappresenta il file che verrà visualizzato come allegato dell'indagine. Fa riferimento a **files.idFiles**.

Per quanto riguarda la tabella "indaginieliminate", essa contiene tre campi:

- **id(int, chiave primaria)**: è l'indice che identifica univocamente ogni indagine eliminata.
- **idUtente(int, chiave esterna)**: Indica l'id dell'utente che ha eliminato l'indagine. Fa riferimento a **utenti.id**.
- **indagine_id(int, chiave esterna)**: Indica l'id dell'indagine che è stata eliminata. Fa riferimento a **indagini.id**.

MODIFICHE AL FILE UTILITY.PHP

Il file Utility.php (*fsem\modello PBAC\Utility.php*) è stato arricchito con nuove funzioni necessarie allo sviluppo di questo modulo. Sono state inserite funzioni per l'inserimento e modifica dei dati all'interno delle tabelle "indagini" e "indaginieliminate".

```
function nuovaIndagineRichiesta($idPaziente, $careprovider, $careproviderNome, $idMotivo, $motivo, $stato, $tipo){
    global $database;
    $query = 'insert into indagini (idpaziente, idcpp, careprovider, idDiagnosi, motivo, stato, tipoIndagine, dataInserimento)
        values (\'$idPaziente.\',\'$careprovider.\',\'$careproviderNome.\',\'$idMotivo.\',\'$motivo.\',\'$stato.\',\'$tipo.\',CURRENT_TIMESTAMP)';
    executeQuery($query);
    return $query;
}
```

```
function nuovaIndagineProgrammata($idPaziente, $careprovider, $careproviderNome, $idMotivo, $motivo, $stato, $tipo, $data, $centro){
    global $database;
    $query = 'insert into indagini (idpaziente, idcpp, careprovider, idDiagnosi, motivo, stato, tipoIndagine, dataInserimento, dataIndagine, idStudioIndagini)
        values (\'$idPaziente.\',\'$careprovider.\',\'$careproviderNome.\',\'$idMotivo.\',\'$motivo.\',\'$stato.\',\'$tipo.\',CURRENT_TIMESTAMP,\'$data.\',\'$centro.\')';
    executeQuery($query);
    return $query;
}
```

```
function nuovaIndagineCompletata($idPaziente, $careprovider, $careproviderNome, $idMotivo, $motivo, $stato, $tipo, $data, $centro, $referto, $allegato){
    global $database;
    $query = 'insert into indagini (idpaziente, idcpp, careprovider, idDiagnosi, motivo, stato, tipoIndagine, dataInserimento, dataIndagine, idStudioIndagini, referto, allegato)
        values (\'$idPaziente.\',\'$careprovider.\',\'$careproviderNome.\',\'$idMotivo.\',\'$motivo.\',\'$stato.\',\'$tipo.\',CURRENT_TIMESTAMP,\'$data.\',\'$centro.\',\'$referto.\',\'$allegato.\')';
    executeQuery($query);
    return $query;
}
```

Come possiamo vedere, le funzioni **nuovaIndagineRichiesta**, **nuovaIndagineProgrammata** e **nuovaIndagineCompletata** sono molto simili. Il loro compito è utilizzare i dati passati come parametri per costruire una query di inserimento nella tabella "indagini". La differenza tra le tre funzioni sta nello stato dell'indagine che verrà registrato e nel numero di parametri passati, ricordiamo che ad esempio una indagine completata dovrà registrare più dati rispetto ad una indagine richiesta.

```
function modificaIndagineRichiesta($idIndagine, $careprovider, $careproviderNome, $idMotivo, $motivo, $stato, $tipo){
    global $database;
    $query = 'UPDATE indagini SET idcpp=\'$careprovider.\', careprovider=\'$careproviderNome.\', idDiagnosi=\'$idMotivo.\', motivo=\'$motivo.\', stato=\'$stato.\', tipoIndagine=\'$tipo.\' WHERE id=\'$idIndagine\'';
    executeQuery($query);
    return $query;
}
```

```

function modificaIndagineProgrammata($idIndagine, $careprovider, $careproviderNome, $idMotivo,
$motivo, $stato, $tipo, $data, $centro){
    global $database;
    $query = 'update indagini set idcpp='.$careprovider.', careprovider="'.
$careproviderNome.'",idDiagnosi='.$idMotivo.',motivo="'.$motivo.'",
    stato="'.$stato.'",tipoIndagine="'.
$tipo.'",dataAggiornamento=CURRENT_TIMESTAMP,dataIndagine=\''.$data.'\',$idStudioIndagini='.
$centro.' where id='.$idIndagine;
    executeQuery($query);
    return $query;
}

function modificaIndagineCompletata($idIndagine, $careprovider, $careproviderNome, $idMotivo,
$motivo, $stato, $tipo, $data, $centro, $referto, $allegato){
    global $database;
    $query = 'update indagini set idcpp='.$careprovider.', careprovider="'.
$careproviderNome.'",idDiagnosi='.$idMotivo.',motivo="'.$motivo.'",
    stato="'.$stato.'",tipoIndagine="'.
$tipo.'",dataAggiornamento=CURRENT_TIMESTAMP,dataIndagine=\''.$data.'\',$idStudioIndagini='.
$centro.',referto='.$referto.',allegato='.$allegato.' where id='.$idIndagine;
    executeQuery($query);
    return $query;
}

```

Notiamo come anche le funzioni **modificaIndagineRichiesta**, **modificaIndagineProgrammata** e **modificaIndagineCompletata** siano simili. Il loro compito è utilizzare i dati passati come parametri per costruire una query di aggiornamento su una indagine nella tabella "indagini". Queste funzioni vengono invocate quando stiamo modificando le informazioni di una indagine già inserita precedentemente

```

function modificaStatoIndagine($id,$stato){
    global $database;
    $query = 'update indagini set stato="'.$stato.'" where id='.$id;
    executeQuery($query);
}

function eliminaIndagine($id, $idutente){
    global $database;
    $query = 'insert into indaginieliminate (idutente, indagine_id) values ('.$idutente.','.
$id.')';
    executeQuery($query);
    return $query;
}

```

La funzione **modificaStatoIndagine** si limita a modificare solo lo stato di una indagine. Viene utilizzata per le richieste di eliminazione, dato che essa comporta il cambiamento dello stato al valore "3" nella tabella "indagini".

La funzione **eliminaIndagine** viene anch'essa utilizzata per eliminare una indagine. Il suo compito è inserire un nuovo record nella tabella "indaginieliminate" dove verrà registrato l'id dell'indagine cancellata e l'id dell'utente che ha eseguito l'azione.

MODIFICHE ALLA SEZIONE FILES

La sezione files è stata leggermente modificata per permettere l'inserimento di referti e allegati all'interno di una indagine. Siccome referti e allegati devono essere gestiti come gli altri tipi di files, ho ritenuto opportuno utilizzare questa sezione per caricare i file e dare la possibilità dal form di creazione indagini di scegliere referti e allegati da una lista di file precedentemente caricati.

template_page_files.php

In questo file ho aggiunto due categorie dove poter caricare i file per le indagini: Referti indagini diagnostiche e Allegati indagini diagnostiche.

```
<div class="col-lg-4">
  <a class="accordion-toggle" data-toggle="collapse" data-parent="#collapseTwo_A"
href="#collapseReferti">
    <h2>Referti indagini diagn.</h2>
  </a>
</div><!--col-lg-4-->
<div class="col-lg-4">
  <a class="accordion-toggle" data-toggle="collapse" data-parent="#collapseTwo_A"
href="#collapseAllegati">
    <h2>Allegati indagini diagn.</h2>
  </a>
</div><!--col-lg-4-->
```

Il codice html integrale è riportato e commentato nel sorgente.

Files

In questa pagina sarà possibile visualizzare ed inviare files di immagini di lesioni cliniche immagini di indagini diagnostiche, registrazioni, brevi video, risultati di esami o documenti testuali.

Files Caricati

Carica nuovi files

Foto del paziente

Video del paziente

Registrazioni

Video Esami Strumentali

Immagini Dicom

Documenti di testo

Referti indagini diagn.

Allegati indagini diagn.

Referti per indagini diagnostiche

accetta i formati: pdf, doc, docx,txt, odt. Nel caso i files contengano informazioni sensibili è raccomandata la protezione con password.

Sfoglia...

Note sul file caricato:

visibilità riservato

Invia

Reset

configFiles.php

```
/**
 * Added 04/03/17 per modulo indagini: class7 per referti indagini e class8 per allegati
 */
$formato_class7 = array ( ".pdf", ".doc", ".docx", ".txt", ".odt");
$formato_class8 = array ( ".pdf", ".doc", ".docx", ".txt", ".odt", "jpeg", "JPG", "gif",
"png", "PNG", "jpg",
".3gp", ".DivX", ".MPEG", ".MOV", ".wlmv", ".wmv", ".mp3", ".wav", ".ogg", ".m4a");
```

Qui sono state inserite le classi con le estensioni accettate per referti e indagini. *\$formato_class7* fa riferimento ai referti e *\$formato_class8* agli allegati.

funzioniFiles.php

```
/**
 * Added 04/03/17 per modulo indagini: controllo file per referti indagini
 */
case 7 :
    foreach ( $formato_class7 as $formato)
    {
        if (strrpos($nomeFile, $formato))
            return TRUE;
    }
    return FALSE;
case 8 :
    foreach ( $formato_class8 as $formato)
    {
        if (strrpos($nomeFile, $formato))
            return TRUE;
    }
    return FALSE;
```

Qui sono stati inseriti i controlli per *\$formato_class7* e *\$formato_class8*

uploadFiles.php

```
case 7 :
    if ( controllaFormato($nome, 7) )
        $uploadPath = $uploadPath . "refertiIndagini/";
    else $formatoErrato = TRUE;
    break;
case 8 :
    if ( controllaFormato($nome, 8) )
        $uploadPath = $uploadPath . "allegatiIndagini/";
    else $formatoErrato = TRUE;
    break;
```

Qui sono stati inseriti i path dove vengono salvati i file per i referti e gli allegati. Nella cartella *FSEM/files/uploads* sono state create altre due sottocartelle: *refertiIndagini/* e *allegatiIndagini/*

La pagina indagini.php contiene il codice per il recupero di tutti i dati necessari a costruire le tabelle per il diario indagini diagnostiche e per i centri indagini diagnostiche.

Ricordiamo che quando una qualsiasi pagina viene caricata, si hanno a disposizione due variabili: `$cp_id` e `$pz_id` che contengono rispettivamente gli id utente del careprovider che sta visualizzando la pagina e del paziente. I record della tabella indagini, però, non fanno riferimento all'id utente ma al relativo id paziente (o id careprovider). Dunque è necessario interrogare il database per ottenere queste informazioni, partendo dall'id utente.

```
// Partendo dall'id utente ($pz_id) ottengo il relativo id paziente
$idPaziente = getInfo('id', 'pazienti', 'idutente = '.$pz_id);
$pag_indagini -> set_var('idPaz', $idPaziente);
// Faccio la stessa cosa per il cp, ottenendo il suo id careproviderpersona
$idCp = getInfo('id', 'careproviderpersona', 'idutente='.$cp_id);
$pag_indagini -> set_var('idUtenteCp', $idCp);
```

Inoltre, recupero il livello di confidenzialità tra careprovider e paziente, questa informazione sarà necessaria per stabilire quali indagini visualizzare e quali no (se è un careprovider a visualizzare la pagina).

```
//Estraggo il livello di confidenzialità tra careprovider e paziente
$confidenzialità = getInfo('confidenzialità', 'careproviderpaziente', 'idutente=' .
$pag_indagini -> set_var('confidenzialità', $confidenzialità);
```

Recupero dati per costruzione tabelle indagine

Recupero anche il nome e il cognome del careprovider che sta visualizzando la pagina.

```
//Inoltre, estraggo il mio nome e cognome (mi serve per le operazioni di
inserimento/modifica diagnosi)
$cpNome = getInfo('nome', 'careproviderpersona', 'id='.$idCp);
$cpCognome = getInfo('cognome', 'careproviderpersona', 'id='.$idCp);
$pag_indagini -> set_var('mioCpNome', $cpNome);
$pag_indagini -> set_var('mioCpCognome', $cpCognome);
```

Infine, procedo a recuperare tutti i record del paziente interessato dalla tabella "indagini", estraendoli per campo e inserendoli ciascuno in un array differente:

- **id** (Id del record indagine)
- **tipolIndagine** (il nome dell'indagine)
- **dataIndagine** (la data in cui verrà effettuata una indagine, se programmata)
- **referto** (l'id del referto collegato all'indagine)
- **allegato** (l'id dell'allegato collegato all'indagine)
- **idcpp** (l'id del careprovider che ha richiesto l'indagine)
- **idStudioIndagini** (l'id del centro indagini in cui verrà effettuata l'indagine)

- **motivo** (la motivazione per cui viene effettuata l'indagine)
- **stato** (lo stato dell'indagine, può essere richiesta, programmata o completata)
- **idDiagnosi** (l'id della diagnosi a cui può essere collegata l'indagine)
- **careprovider** (nome e cognome del careprovider che ha richiesto l'indagine)

Per ogni indagine, identificata univocamente dall'id, vado a recuperare ulteriori informazioni dalle altre tabelle. In particolare:

- Per ogni id del campo **referto** e **allegato** vado a recuperare dalla tabella **files** le informazioni relative a nome del file, data di creazione, path del file, e confidenzialità.
- Per ogni id del campo **idcpp** vado a recuperare dalla tabella **careproviderpersona** le informazioni relative a nome, cognome e reperibilità.
- Per ogni id del campo **idStudioIndagini** vado a recuperare dalla tabella **centriindagini** le informazioni relative a nome, via e città del centro indagini.
- Per ogni id del campo **idDiagnosi** vado a recuperare dalla tabella **diagnosi** le informazioni relative al nome e al livello di confidenzialità dell'indagine.

Infine trasmetto queste informazioni alla pagina **template_page_indagini.php** codificando ogni dato all'interno di una variabile dichiarata nel seguente modo:

ind.campo.indice

Dove **campo** indica il nome dell'informazione che stiamo passando (motivo, stato, etc.) e **indice** rappresenta l'identificativo dell'indagine a cui l'informazione appartiene.

Di seguito propongo un estratto del codice come esempio:

```
$indaginiId = getArray('id', 'indagini', 'idPaziente='.$idPaziente . ' ORDER BY
dataIndagine ASC');
$indaginiTipo = getArray('tipoIndagine', 'indagini', 'idPaziente='.$idPaziente . '
ORDER BY dataIndagine ASC');

...

for($i=0; $i<$n; $i++){ //PER OGNI INDAGINE...
    $pag_indagini -> set_var('ind.id.'.$i, $indaginiId[$i]);
    $pag_indagini -> set_var('ind.tipo.'.$i, $indaginiTipo[$i]);

//RECUPERO I DATI PER IL CAREPROVIDER COLLEGATO ALL'INDAGINE
$pag_indagini -> set_var('ind.cpId.'.$i, $indaginiCp[$i]);
$pag_indagini -> set_var('ind.careprovider.'.$i, $indaginiCareprovider[$i]);
$careproviderNome = getInfo('nome', 'careproviderpersona', 'id='.$indaginiCp[$i]);
$careproviderCognome = getInfo('cognome', 'careproviderpersona', 'id='.
$indaginiCp[$i]);
$careproviderRep = getInfo('reperibilita', 'careproviderpersona', 'id='.
$indaginiCp[$i]);
$pag_indagini -> set_var('ind.cpNome.'.$i, $careproviderNome);
$pag_indagini -> set_var('ind.cpCognome.'.$i, $careproviderCognome);
$pag_indagini -> set_var('ind.cpRep.'.$i, $careproviderRep);

...

} // FINE CICLO PER OGNI INDAGINE
```

Recupero dati per costruzione form e tabella centro indagini

E' necessario recuperare dati da altre tabelle per la costruzione dei form di inserimento e modifica e per la costruzione della tabella contenente la lista dei centri indagini.

In particolare, per visualizzare i careprovider nel menù a tendina dei form di inserimento e modifica, andiamo a recuperare i dati dei careprovider collegati al paziente dalle tabelle **careproviderpaziente** e **careproviderpersona**.

```
$cpRegistratiIdUtente = getArray('idcpp', 'careproviderpaziente', 'idutente='.$idPaziente);
$v = count($cpRegistratiIdUtente);
for($i=0; $i<$v; $i++){
    $cpRegistratoId = getInfo('id', 'careproviderpersona', 'idutente='.$cpRegistratiIdUtente[$i]);
    $cpRegistratoNome = getInfo('nome', 'careproviderpersona', 'id='.$cpRegistratoId);
    $cpRegistratoCognome = getInfo('cognome', 'careproviderpersona', 'id='.$cpRegistratoId);
    $pag_indagini -> set_var('careprovider.id.'.$i, $cpRegistratoId);
    $pag_indagini -> set_var('careprovider.nome.'.$i, $cpRegistratoNome);
    $pag_indagini -> set_var('careprovider.cognome.'.$i, $cpRegistratoCognome);
}
$pag_indagini -> set_var('careproviderNum', $v);
```

Seguendo la stessa tecnica, per tutti gli altri menù a tendina andiamo a recuperare le informazioni dalle rispettive tabelle.

Per la costruzione della tabella centri indagini sarà necessario recuperare i dati dalle tabelle **centriindagini** e **telefonocentriindagini**.

```
/* *****
 * RECUPERO I CENTRI DIAGNOSTICI
 * ***** */
$centriId = getArrayNoCondition('id', 'centriindagini ORDER BY id');
$centriIdCpp = getArrayNoCondition('idcpp', 'centriindagini ORDER BY id');
$centriNome = getArrayNoCondition('nomeStudio', 'centriindagini ORDER BY id');
$centriVia = getArrayNoCondition('via', 'centriindagini ORDER BY id');
$centriCitta = getArrayNoCondition('citta', 'centriindagini ORDER BY id');
$centriTipo = getArrayNoCondition('tipoCentro', 'centriindagini ORDER BY id');
$centriEmail = getArrayNoCondition('mail', 'centriindagini ORDER BY id');
$m = count($centriId);
for($i=0; $i<$m; $i++){
    $pag_indagini -> set_var('centro.id.'.$i, $centriId[$i]);
    $pag_indagini -> set_var('centro.nome.'.$i, $centriNome[$i]);
    $pag_indagini -> set_var('centro.via.'.$i, $centriVia[$i]);
    $pag_indagini -> set_var('centro.citta.'.$i, $centriCitta[$i]);
    if ($centriTipo[$i] == "Studio specialistico")
        $pag_indagini -> set_var('centro.tipo.'.$i, 0);
    else if ($centriTipo[$i] == "Studio radiologico")
        $pag_indagini -> set_var('centro.tipo.'.$i, 1);
    else if ($centriTipo[$i] == "Laboratorio analisi")
        $pag_indagini -> set_var('centro.tipo.'.$i, 2);
    else
        $pag_indagini -> set_var('centro.tipo.'.$i, 3);
    $pag_indagini -> set_var('centro.mail.'.$i, $centriEmail[$i]);
    $pag_indagini -> set_var('centro.responsabileId.'.$i, $centriIdCpp[$i]);
    $responsabileNome = getInfo('nome', 'careproviderpersona', 'id='.$centriIdCpp[$i]);
    $responsabileCognome = getInfo('cognome', 'careproviderpersona', 'id='.$centriIdCpp[$i]);
    $pag_indagini -> set_var('centro.responsabileNome.'.$i, $responsabileNome);
    $pag_indagini -> set_var('centro.responsabileCognome.'.$i, $responsabileCognome);
    $contattiTel = getArray('telefono', 'telefonocentriindagini', 'idCentroIndagini='.$centriId[$i]);
    $numeriTelefono = "";
    foreach($contattiTel as $stel) //RECUPERO TUTTI I CONTATTI TELEFONICI COMBINANDOLI
        $numeriTelefono = '<a href="tel:'.$stel.'"><i class="glyphicon glyphicon-earphone"
    </i> '.$stel.' </a><br>' . $numeriTelefono;
    $pag_indagini -> set_var('centro.contatti.'.$i, $numeriTelefono);
}
$pag_indagini -> set_var('centriNum', $m);
```

TEMPLATE_PAGE_INDAGINI.PHP

Il file *template_page_indagini.php* si occupa di costruire la pagina principale della sezione Indagini diagnostiche che l'utente andrà a visualizzare. La pagina deve contenere tabelle e form il cui contenuto varia in base al paziente e alle modifiche che apporta mentre è sulla pagina, per questo motivo gran parte del codice html è generato tramite php, mentre per la modifica in tempo reale dei form si utilizza javascript e jquery, in parte presente su questa stessa pagina e in parte in *indagini.js*.

Il codice di questa pagina si può scomporre in sezioni nel seguente modo:

- Si stabilisce se l'utente sia arrivato sulla pagina indagini dal menù principale o dalla sezione Diagnosi.
- Si procede a caricare i dati da visualizzare dalla pagina *indagini.php*
- In php, si generano tabelle e form contenenti le indagini del paziente
- In javascript, si gestisce il controllo dei dati e l'esecuzione dei processi per l'inserimento e la modifica dei dati (compito rimandato alle pagine *indagini.js*, *nuovaIndagine.php*, *modificaIndagine.php* e *eliminaIndagine.php*)

Controllo dell'accesso

Innanzitutto, dopo aver recuperato gli id del paziente e del careprovider (se connesso), provvediamo a stabilire se l'accesso è stato fatto dal menù principale o dalla sezione Diagnosi. Se l'accesso è stato fatto dalla sezione Diagnosi allora dobbiamo interrogare il database per recuperare i dati sulla diagnosi di cui vogliamo conoscere le indagini connesse.

```
//SE L'ACCESSO VIENE ESEGUITO TRAMITE POST, OVVERO TRAMITE CLICK SU UNA DIAGNOSI...
$accesso_da_menu;
if ($_SERVER['REQUEST_METHOD'] == 'POST'){
    //RECUPERO I DATI RELATIVI ALLA DIAGNOSI COLLEGATA ALLE INDAGINI
    $idDiagnosi = $_POST["idDiagnosi"];
    $idPaziente = getInfo('idPaziente','diagnosi','id='.$idDiagnosi);
    $nomePatologia = getInfo('patologia','diagnosi','id='.$idDiagnosi);
    $dataDiagnosi = getInfo('DATE(datains)','diagnosi','id='.$idDiagnosi);
    $accesso_da_menu = false;
    $stringa_diagnosi = "Indagini relative alla diagnosi <b>". $nomePatologia . "</b> del
<b>" . italianFormat($dataDiagnosi) . "</b>";
}else{
    //ALTRIMENTI, L'ACCESSO E' AVVENUTO DA MENU...
    $accesso_da_menu = true
}
```

Poi, creo un div con dei meta-data per informare la pagina indagini.js riguardo la modalità di accesso:

```
//VARIABILE PER L'USO NEL FILE .JS
echo '<div id="menu_mode" data-menu="' . $accesso_da_menu . '"></div>';
```

Se l'utente connesso è un careprovider, recupero il suo livello di confidenzialità col paziente. Questo è necessario per stabilire quali indagini visualizzare e quali valori mostrare nei form di inserimento e modifica:

```
//SE SONO CAREPROVIDER, RECUPERO ID E CONFIDENZIALITA' COL PAZIENTE
$selfCareproviderId = $this->get_var('idUtenteCp');
$selfCareproviderConf = $this->get_var('confidenzialita');
```

Recupero dei dati dalla pagina indagini.php

Passiamo ora a recuperare i dati dalla pagina indagini.php. I dati vengono recuperati dalle variabili e inseriti in diversi array che poi verranno utilizzati al momento di generare l'html per le tabelle. Prima di tutto dichiaro le variabili che utilizzerò:

```
$stato_richiesta = 0;
$stato_programmata = 1;
$stato_completata = 2;
$array_richieste = array(); //array indagini richieste
$array_programmate = array(); //array indagini programmate
$array_completate = array(); //array indagini completate
$nRic = 0; //numero indagini richieste
$nPro = 0; //numero indagini programmate
$nCom = 0; //numero completate
global $offset; $offset = 24; //offset del vettore per l'inserimento di tutti i dati
global $nIndagini; //numero totale di indagini
$nIndagini = $this->get_var('indaginiNum');
```

Dividiamo le indagini da mostrare in tre array, uno per ogni tabella da mostrare.

L'offset è necessario per stabilire a quale indagine corrisponde il valore presente nell'array ad una data posizione. Dato che l'offset è di 24, i valori dalla posizione 0 a 23 apparterranno alla prima indagine, i valori dalla posizione 24 a 47 alla seconda e così via.

```
//RECUPERO DATI PER OGNI INDAGINE...
for ($i = 0; $i < $nIndagini; $i++){
    //SE L'ACCESSO E' DA MENU O SE IL MOTIVO DELL'INDAGINE E' ASSOCIATO ALLA DIAGNOSI...
    if($accesso_da_menu || $this->get_var('ind.idDiagno.' . $i) == $idDiagnosi ) {
        //SE SONO IL PAZIENTE O SE SONO UN CAREPROVIDER CON CONFIDENZIALITA' SUFFICIENTE...
        if($role == "pz" || $SelfcareproviderConf >= $this->get_var('ind.conf.' . $i)){
            $stato = $this->get_var('ind.stato.' . $i); //verifica lo stato
            switch ($stato) {
                case $stato_richiesta:
                    $array_richieste[$nRic + 0] = $this->get_var('ind.id.' . $i);
                    $array_richieste[$nRic + 1] = $this->get_var('ind.tipo.' . $i);
                    ...
                    $array_richieste[$nRic + 23] = $this->get_var('ind.allegatoConf.' . $i);
                    $nRic = $nRic + $offset;
                    break;
                case $stato_programmata:
                    $array_programmate[$nPro + 0] = $this->get_var('ind.id.' . $i);
                    $array_programmate[$nPro + 1] = $this->get_var('ind.tipo.' . $i);
                    ...
                    $array_programmate[$nPro + 23] = $this->get_var('ind.allegatoConf.' . $i);
                    $nPro = $nPro + $offset;
                    break;
                case $stato_completata:
                    $array_completate[$nCom + 0] = $this->get_var('ind.id.' . $i);
                    $array_completate[$nCom + 1] = $this->get_var('ind.tipo.' . $i);
                    ...
                    $array_completate[$nCom + 23] = $this->get_var('ind.allegatoConf.' . $i);
                    $nCom = $nCom + $offset;
                    break;
            }
        }
    }
}
//FINE RECUPERO PER OGNI INDAGINE
```

Possiamo vedere che l'inserimento dei dati per ogni indagine è regolato da due IF.

- Il primo IF ci dice che possiamo inserire l'indagine in uno degli array solo se l'accesso è stato effettuato dal menu (quindi dobbiamo inserire tutte le indagini) OPPURE se l'accesso è avvenuto dalla sezione Diagnosi e quindi dobbiamo inserire l'indagine solo se è collegata alla diagnosi selezionata.
- Il secondo IF ci dice che, una volta rispettate le condizione del primo IF, possiamo inserire l'indagine solo se o siamo il paziente (quindi posso visualizzare tutte le indagini) OPPURE se, essendo un careprovider, il mio livello di confidenzialità è sufficiente per visualizzare quell'indagine.

Utilizzando la stessa tecnica, ovvero recuperare i dati e inserirli in array, recuperiamo anche gli altri dati:

- **Recupero di tutte le diagnosi collegate al paziente** (necessario per la costruzione del menù a tendina riguardo la motivazione dell'indagine).
- **Recupero di tutti i careprovider registrati al paziente** (necessario per la costruzione del menù a tendina riguardo il careprovider che ha richiesto l'indagine).
- **Recupero dei file di tipo referto e allegato collegati al paziente** (necessario per la costruzione del menù a tendina riguardo il referto e gli allegati da collegare all'indagine).
- **Recupero delle informazioni relative a tutti i centri diagnostici** (necessario per la costruzione delle tabelle della sottosezione "Centri indagini diagnostiche").

Riportiamo qui sotto solo l'esempio per quanto riguarda il recupero delle diagnosi:

```
//RECUPERO INFORMAZIONI DI TUTTE LE DIAGNOSI COLLEGATE AL PAZIENTE...
$array_diagnosi = array();
$ndiagnoIns = 0;
global $n_diagnosi;
$n_diagnosi = $this->get_var('diagnosiNum');
for ($i = 0; $i < $n_diagnosi; $i++) {
    $array_diagnosi[$ndiagnoIns + 0] = $this->get_var('diagnosi.id.'.$i);
    $array_diagnosi[$ndiagnoIns + 1] = $this->get_var('diagnosi.data.'.$i);
    $array_diagnosi[$ndiagnoIns + 2] = $this->get_var('diagnosi.patologia.'.$i);
    $array_diagnosi[$ndiagnoIns + 3] = $this->get_var('diagnosi.conf.'.$i);
    $ndiagnoIns = $ndiagnoIns + 4;
}
```

Costruzione della pagina html

Si procede quindi a costruire la pagina utilizzando html e html generato da codice php. Gran parte del codice viene ripetuto più volte per la costruzione delle tre tabelle per il diario indagini e le tre tabelle dei centri indagini diagnostiche in quanto possiedono una struttura identica. Stessa cosa per i form, dato che il form di inserimento è identico ai form di modifica generati per ogni indagine visualizzata, il codice utilizzato è identico. Per cui commenterò i punti più importanti.

Costruzione del form di inserimento nuova indagine

Prima di tutto costruiamo il form per inserire una nuova indagine, in particolare mostriamo come è costruito il menù a tendina per la selezione della motivazione (che ricordiamo può contenere o una diagnosi collegata al paziente o una stringa digitata dall'utente).


```

<div class="form-group">
  <label class="control-label col-lg-4">Motivo *</label>
  <div class="col-lg-4">
    <select id="motivoIndagine_new" class="form-control">
      <option selected hidden style='display: none' value="placeholder">Selezionare una
motivazione..</option>
      <optgroup label="Diagnosi del paziente">
        <?php
          for($i = 0; $i < $nDiagnoIns; $i +=4 ){
            if($role == "pz" || $SelfcareproviderConf >= $array_diagnosi[$i+3] ){
              echo '<option value="'. $array_diagnosi[$i+0] .' ">' .
                $array_diagnosi[$i+1] .' - ' . $array_diagnosi[$i+2] .'</option>';
            }
          }
        ?>
      </optgroup>
      <option value=''>Altra Motivazione..</option>
    </select>
    <input id="motivoAltro_new" type="text" placeholder="Inserire motivazione.."
class="form-control"/>
  </div>
</div>

```

Come possiamo vedere, le opzioni del menù a tendina vengono generate utilizzando un ciclo *FOR* che cicla su *\$array_diagnosi* la cui costruzione è stata commentata più sopra. Vediamo che il *FOR* cicla incrementando l'indice di + 4 ogni volta, questo perchè l'offset è di 4, cioè i valori dell'array in posizione da 0 a 3 saranno della prima diagnosi, da 4 a 7 della seconda e così via.

Prima di inserire la diagnosi facciamo un controllo:

```
if($role == "pz" || $SelfcareproviderConf >= $array_diagnosi[$i+3] )
```

Ovvero stiamo controllando se l'utente connesso è paziente (quindi può visualizzare nel menu' tutte le diagnosi. Altrimenti se è un careprovider, dobbiamo vedere se il suo livello di confidenzialità è maggiore o uguale a quello della diagnosi per poterla visualizzare nel menu'.

Infine inseriamo l'opzione:

```
echo '<option value="'. $array_diagnosi[$i+0] .' ">' . $array_diagnosi[$i+1] .' - ' .
$array_diagnosi[$i+2] .'</option>';
```

il valore dell'opzione corrisponderà all'id della diagnosi (memorizzata in *\$array_diagnosi[\$i+0]*) mentre nella lista verrà visualizzato *\$array_diagnosi[\$i+1] . ' - ' . \$array_diagnosi[\$i+2]* ovvero la data della diagnosi e il suo nome.

Infine notiamo come l'ultima opzione non sia generata dal ciclo:

```
<option value=''>Altra Motivazione..</option>
```

Questa è infatti una opzione con valore nullo, quando questa opzione viene selezionata si rende visibile la textbox per l'inserimento della motivazione da parte dell'utente:

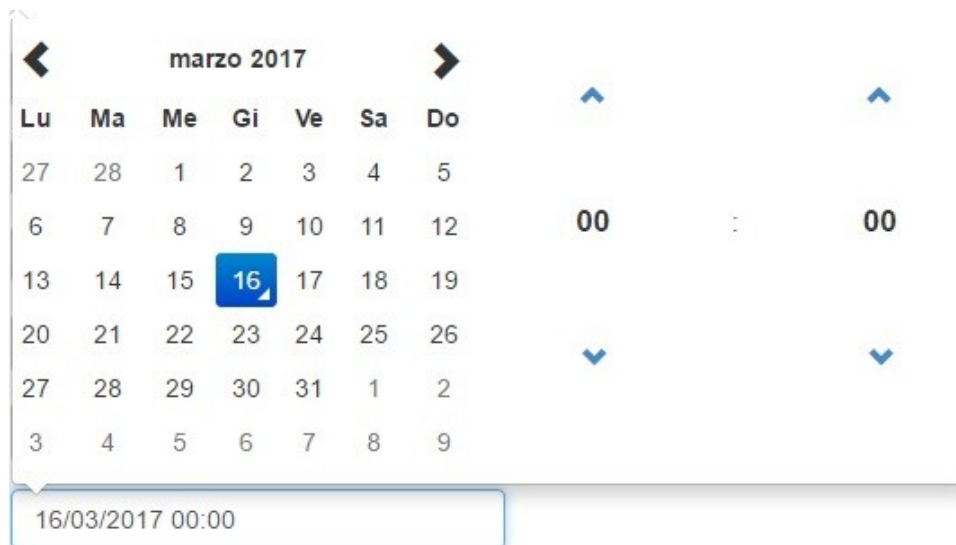
```
<input id="motivoAltro_new" type="text" placeholder="Inserire motivazione.." class="form-control"/>
```

La gestione della visualizzazione dei campi del form è gestita dal file *indagini.js* di cui parleremo più avanti.

Gli altri menù a tendina sono realizzati allo stesso modo, iterando con un ciclo *FOR* su uno degli array creati in precedenza.

Per la data invece ho utilizzato il [bootstrap-datetimepicker](#), un widget javascript che permette di selezionare data e orario attraverso un calendario.

```
<div class="form-group"><label class="control-label col-lg-4">Data *</label>
  <div class="col-lg-4">
    <input id="data".$array_richieste[$i+0].'" type="text" placeholder="Selezionare una
data.." class="form-control"
    value="'.$array_richieste[$i+4].'" />
  </div>
  <script>
    $j(document).ready(function() {
      $j('#data'.$array_richieste[$i+0].'\').datetimepicker({
        locale:'\it\'',
        sideBySide:true
      });
    });
  </script>
</div>
```



Il file necessario per il funzionamento del plugin si trova in *fsem/assets/js/moment-with-locales.min.js*

Costruzione delle tabelle indagini

Anche le tabelle vengono costruite generando del codice tramite php, mostriamo qui quella della tabella "indagini richieste":

```
<?php //popolamento tabella indagini richieste
for($i = 0; $i < $nRic; $i+=$offset){
  echo '<tr class="info" id="r'.$array_richieste[$i+0].'">';
  echo '<td id="tipoRichiesta'.$array_richieste[$i+0].'">' . $array_richieste[$i+1] .
'</td>';
  echo '<td id="motivoRichiesta'.$array_richieste[$i+0].'">' . $array_richieste[$i+2] .
'</td>';
  echo '<td id="careRichiesta'.$array_richieste[$i+0].'">' .
$array_richieste[$i+16] . '</td>';
  echo '<td style="text-align:center"><div id="btn-group">
<button id="'.$array_richieste[$i+0].'" class="modifica btn btn-success "><i class="icon-pencil
icon-white"></i></button>
  <button id="'.$array_richieste[$i+0].'" class="elimina btn btn-danger"><i class="icon-remove
icon-white"></i></button>
</div></td></tr>';
```


Come possiamo vedere, utilizziamo un ciclo *FOR* simile a quello utilizzato per la costruzione dei form. Ogni iterazione genera una riga contenente un'indagine e ogni istruzione di *echo* genera il contenuto di una cella per quella riga. In questo esempio possiamo vedere che stiamo generando le celle contenenti il tipo, il motivo e il careprovider dell'indagine, più i due pulsanti per modifica ed eliminazione.

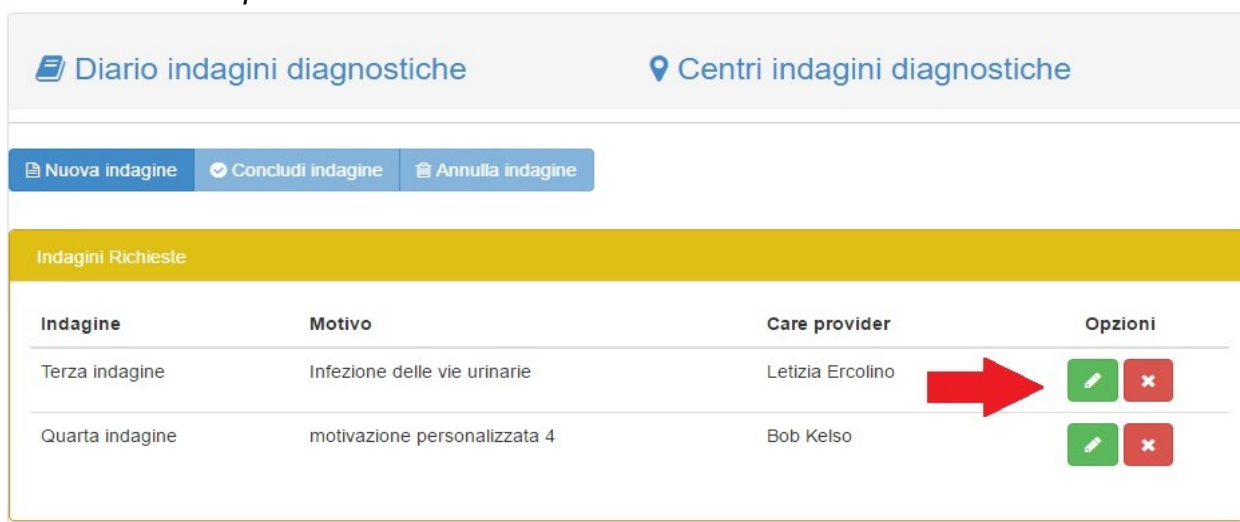
Notiamo come gli id di ogni cella sono nella forma `<nome_campo><id indagine>`

come ad esempio: `id="tipoRichiesta".$array_richieste[$i+0]."`


In questo caso l'id di tale cella potrebbe essere ad esempio `"tipoRichiesta54"`. In questo modo ogni cella ha un identificativo unico dettato dall'id dell'indagine. Utilizzeremo questa tecnica anche per generare gli id dei campi dei form di modifica, necessari per passare i dati alla pagina *indagini.js*.

Prima di chiudere il ciclo *FOR* andiamo a generare un'altra riga immediatamente sotto, che però sarà nascosta. Questa riga conterrà il form di modifica, identico al form di inserimento, salvo che i campi saranno tutti precompilati coi valori dell'indagine che stiamo modificando.

Prima di cliccare sul pulsante "modifica" indicato:

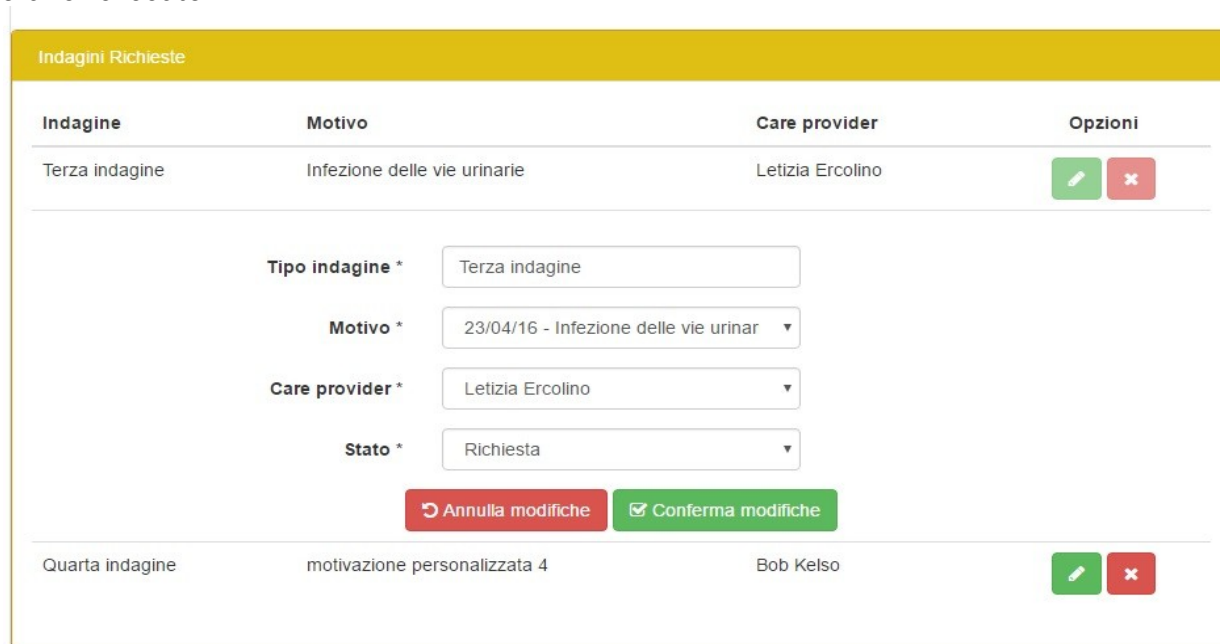


The screenshot shows a web interface titled "Diario indagini diagnostiche" with a sub-header "Centri indagini diagnostiche". Below the header are three buttons: "Nuova indagine", "Concludi indagine", and "Annulla indagine". The main section is titled "Indagini Richieste" and contains a table with the following data:

Indagine	Motivo	Care provider	Opzioni
Terza indagine	Infezione delle vie urinarie	Letizia Ercolino	 
Quarta indagine	motivazione personalizzata 4	Bob Kelso	 

A red arrow points to the pencil icon in the "Opzioni" column for the first row.

Dopo aver cliccato:



The screenshot shows the same interface as before, but the form for editing the first investigation is now visible. The form contains the following fields:

- Tipo indagine ***: Terza indagine
- Motivo ***: 23/04/16 - Infezione delle vie urinar
- Care provider ***: Letizia Ercolino
- Stato ***: Richiesta

Below the form are two buttons: "Annulla modifiche" and "Conferma modifiche". The table below the form remains the same.

Anche le tabelle per i centri indagini diagnostiche sono costruite utilizzando lo stesso metodo, mostriamo l'esempio per la tabella degli studi specialistici:

```
<?php //popolamento tabella studi specialistici
$link = $this->get_var('link_mostratutti');
for($i = 0; $i < $nCentriIns; $i+=10) {
    if ($array_centri[$i + 4] == 0){
        echo '<tr class="info" id="studioSpecialistico" . $array_centri[$i + 0] . ">';
        echo '<td id="nomeStudioSpecialistico" . $array_centri[$i + 0] . ">' .
            $array_centri[$i + 1] . '</td>';
        echo '<td id="sedeStudioSpecialistico" . $array_centri[$i + 0] . ">' .
            $array_centri[$i + 2] . '<br>' . $array_centri[$i + 3] . '</td>';
        echo '<td id="contattiStudioSpecialistico" . $array_centri[$i + 0] . ">' .
            $array_centri[$i + 9] . '</td>';
        echo '<td><a href="mailto:'. $array_centri[$i + 5].'">
            <button class="btn btn-warning" type="button" id="mailStudioSpecialistico".
            $array_centri[$i + 0] . ">
            <i class="icon-envelope"></i></button> ' . $array_centri[$i + 5]. '</a></td>';
        echo '<td ><a class="a-messaggio" id="'. $array_centri[$i + 0] .'" data-toggle="modal"
            data-target="#messageModal" href="'. $link .'" >
            <button class="btn-messaggio btn" type="button" id="'.
            $array_centri[$i + 0] . ">
            <i class="icon_custom-chat"></i></button> ' . $array_centri[$i + 7] . ' '.
            $array_centri[$i + 8] . '</a>
            <div id="careproviderStudio". $array_centri[$i + 0] .'" data-nome="'.
            $array_centri[$i + 7] . ' '. $array_centri[$i + 8] . '"></div></td>
            </tr>';
    }
}
```

Vediamo che il *FOR* cicla sul vettore *\$array_centri*, che contiene tutti i centri indagine, con un offset di 10. Prima di inserire il centro nella tabella degli studi specialistici, controlla che sia della tipologia giusta: `if ($array_centri[$i + 4] == 0)` dove 0 rappresenta "studio specialistico" e *\$array_centri[\$i+4]* contiene la tipologia del centro che stiamo esaminando.

L'istruzione `` crea un link che permette di inviare un email all'indirizzo contenuto in *\$array_centri[\$i + 5]*

L'istruzione ``

Apri *#messageModal*, per l'invio di un messaggio privato al responsabile del centro.

Nel file `indagini.js` andiamo a definire le funzioni, in javascript, per gestire l'interazione dell'utente con i form e il controllo dei dati inseriti. Una volta controllati i dati, vengono inoltrati alle pagine `nuovaIndagine.php`, `modificaIndagine.php` e `eliminaIndagine.php` a seconda dell'operazione richiesta.

Passiamo ad esaminare il codice:

```
/**
 * Funzioni per la visualizzazione dei textbox per motivo e careprovider
 * nel caso vengano scelti motivazione e careprovider non presenti in db
 */
function motivoChange() {
    var index = $(this).attr("id").replace('motivoIndagine', '');
    var value = $(document.getElementById("motivoIndagine" + index)).val();
    if (value == "") {
        document.getElementById("motivoAltro" + index).style.display = "block";
    }
    else {
        document.getElementById("motivoAltro" + index).style.display = "none";
    }
}
function careproviderChange() {
    var index = $(this).attr("id").replace('careproviderIndagine', '');
    var value = $(document.getElementById("careproviderIndagine" + index)).val();
    if (value == "") {
        document.getElementById("careproviderAltro" + index).style.display = "block";
    }
    else {
        document.getElementById("careproviderAltro" + index).style.display = "none";
    }
}
```

Queste due funzioni **motivoChange** e **careproviderChange** mostrano o nascondono i textbox per l'inserimento di careprovider e motivazione dell'indagine se nei rispettivi menu a tendina è stata scelta l'opzione di inserirli manualmente. I campi avvalorati con `display = "block"` sono visibili.

Campi visibili

Tipo indagine *	<input type="text"/>
Motivo *	<div>Altra Motivazione.. ▾ Inserire motivazione..</div>
Care provider *	<div>Nuovo careprovider.. ▾ Inserire careprovider..</div>
Stato *	<div>Richiesta ▾</div>

Campi non visibili

Tipo indagine *	<input type="text"/>
Motivo *	<div>23/04/16 - Infezione delle vie urinari ▾</div>
Care provider *	<div>Letizia Ercolino ▾</div>
Stato *	<div>Richiesta ▾</div>

La funzione **statoChange()** opera allo stesso modo ma permette di visualizzare e nascondere determinati campi in base allo stato dell'indagine selezionata.

```
/**
 * Funzione per la visualizzazione dei campi appropriati in base allo stato dell'indagine
 */
function statoChange() {
    var index = $(this).attr("id").replace('statoIndagine', '');
    switch ($(document.getElementById("statoIndagine" + index)).val()) {
        case "0":
            document.getElementById("divCentro" + index).style.display = "none";
            document.getElementById("divData" + index).style.display = "none";
            document.getElementById("divReferto" + index).style.display = "none";
            document.getElementById("divAllegato" + index).style.display = "none";
            break;
        case "1":
            document.getElementById("divCentro" + index).style.display = "block";
            document.getElementById("divData" + index).style.display = "block";
            document.getElementById("divReferto" + index).style.display = "none";
            document.getElementById("divAllegato" + index).style.display = "none";
            break;
        case "2":
            document.getElementById("divCentro" + index).style.display = "block";
            document.getElementById("divData" + index).style.display = "block";
            document.getElementById("divReferto" + index).style.display = "block";
            document.getElementById("divAllegato" + index).style.display = "block";
            break;
    }
}
```

Le funzioni **setError(id)** e **unsetError(id)** utilizzano bootstrap per evidenziare eventuali campi del form che presentano errori.

```
/**
 * Funzioni per la visualizzazione e rimozione del bordo rosso dei campi del form con errori
 * @param id : ID del campo da bordare
 */
function setError(id) {
    $(id).closest(".form-group").addClass('has-error');
}
function unsetError(id) {
    $(id).closest(".form-group").removeClass('has-error');
}
```

La classe **"has-error"** viene utilizzata in bootstrap per evidenziare automaticamente un modulo che contiene errori (in questo caso, viene bordato in rosso).

Funzioni di validazione dei form

Esaminiamo le funzioni che vengono invocate per controllare se un form è stato compilato correttamente. Utilizziamo tre funzioni di validazione da utilizzare a seconda dello stato impostato dell'indagine, questo perché a seconda dello stato il numero di campi da compilare sarà diverso.

Le funzioni restituiscono un valore booleano: **TRUE** se il form non ha errori, **FALSE** altrimenti.

```
function validateRichiesta(tipoId, motivoId, motivoAltroId, careproviderId,
careproviderAltroId) {
    var isValid = true;
    var tipo = $(tipoId).val().trim();
    var motivo = $(motivoId).val().trim();
    var motivoAltro = $(motivoAltroId).val().trim();
    var careprovider = $(careproviderId).val().trim();
    var careproviderAltro = $(careproviderAltroId).val().trim();
    if (tipo == ''){
        isValid = false; //tipo vuoto
        setError(tipoId);
    } else unsetError(tipoId);
    if (motivo == "placeholder" || (motivo == '' && motivoAltro == '')){
        isValid = false; //motivo placeholder or empty
        setError(motivoId);
    } else unsetError(motivoId);
    if (careprovider == "placeholder" || (careprovider == '' && careproviderAltro == '')){
        isValid = false; //care placeholder or empty
        setError(careproviderId);
    } else unsetError(careproviderId);
    return isValid;
}
```

Nella funzione **validateRichiesta** controlliamo i dati per una indagine richiesta. Prima di tutto recuperiamo i dati utilizzando gli id in input, poi controlliamo che *"tipo"* (ossia il nome dell'indagine) non sia vuoto. Poi controlliamo *"motivo"* e *"careprovider"*:

```
if (motivo == "placeholder" || (motivo == '' && motivoAltro == ''))
```

Che può essere letto come: *SE il motivo (ossia il menù a tendina) è settato col valore placeholder OPPURE SE il motivo è settato come vuoto E ANCHE motivoAltro (cioè il campo di testo) è vuoto ALLORA il form non è valido.*

Per il campo *"careprovider"* utilizziamo lo stesso metodo. Notiamo infine che oltre a restituire **FALSE**, invochiamo anche la funzione *setError* per bordare di rosso il campo con dati non valido.

```
function validateProgrammata(tipoId, motivoId, motivoAltroId, careproviderId,
careproviderAltroId, dataId, centroId){
    var isValid = validateRichiesta(tipoId, motivoId, motivoAltroId, careproviderId,
careproviderAltroId);
    var centro = $(centroId).val().trim();
    var dataMoment = $j(dataId).data("DateTimePicker").date();
    if(dataMoment == null && $(dataId).val() == ""){
        isValid = false; //data is empty
        setError(dataId);
    } else unsetError(dataId);
    if(centro == "placeholder" || centro == "") {
        isValid = false; //centro is empty
        setError(centroId);
    } else unsetError(centroId);
    return isValid;
}
```

In **validateProgrammata** prima di tutto controlliamo gli stessi campi di **validateRichiesta**, per questo motivo invochiamo subito quella funzione. Poi controlliamo che sia *"data"* e *"centro"* non siano campi vuoti.

```
function validateCompletata(tipo, motivo, motivoAltro, careprovider, careproviderAltro, data,
centro, referto, allegato){
    var isValid = validateProgrammata(tipo, motivo, motivoAltro, careprovider,
careproviderAltro, data, centro);
    // NESSUN VINCOLO
    return isValid;
}
```

In **validateCompletata** a parte gli stessi vincoli di **validateProgrammata** non ce ne sono di nuovi, questo perché gli unici campi aggiuntivi sono "referto" e "allegato" che ho ritenuto opportuno lasciare all'utente la scelta di inserirli o meno, in quanto bisognerebbe prima caricarli nella sezione Files. In questo modo, è comunque possibile aggiungerli successivamente modificando l'indagine.

La funzione `document.ready` e funzioni sui button dei form

La funzione `$(document).ready(function()` viene invocata quando il file, ossia la pagina *template_page_indagini.php*, è stata caricata. All'interno di questa funzione andiamo a definire gli eventi da attivare al momento della pressione dei button. Inoltre, prima di tutto, andiamo ad assegnare ad ogni menù a tendina presente sulla pagina le funzioni **motivoChange**, **statoChange** e **careproviderChange** in modo tale che ogni qual volta che venga modificata l'opzione scelta, le tre funzioni mostrino o nascondano i campi associati.

```
$("#id^=motivoIndagine").change(motivoChange);
$("#id^=statoIndagine").change(statoChange);
$("#id^=careproviderIndagine").change(careproviderChange);
```

La stringa `"id^=motivoIndagine"` indica tutti i div che abbiano come id una stringa che cominci per *"motivoIndagine"* e termini con qualsiasi valore. Ricordiamo che nella pagina *template_page_indagini.php* abbiamo tanti div *"motivoIndagine"* nella forma *"motivoIndagine + id indagine"* quante sono le indagini mostrate nelle tabelle (esempio: *"motivoIndagine45"*, *"motivoIndagine58"*), in questo modo possiamo assegnare la proprietà `change` a tutti quei div contemporaneamente.

Inoltre, controlliamo da dove è stato effettuato l'accesso: se è avvenuto dal menù principale, allora possiamo attivare i pulsanti per l'inserimento di una nuova indagine, altrimenti se è stato effettuato tramite la sezione Diagnosi allora dobbiamo disattivare i pulsanti.

```
//se l'accesso alla pagina è eseguito dal menu, abilita i pulsanti per l'inserimento di nuove
indagini
//se l'accesso è eseguito da un indagine, disabilita i pulsanti
var menu = document.getElementById("menu_mode").getAttribute('data-menu');
if(menu){
    $('#nuovoFile').prop('disabled',false);
    $('#concludi').prop('disabled',true);
    $('#annulla').prop('disabled',true);
}
else{
    $('#nuovoFile').prop('disabled',true);
    $('#concludi').prop('disabled',true);
    $('#annulla').prop('disabled',true);
    $('#collapse1').collapse('show');
}
```

Passiamo alla funzione `$("#concludi").click(function()` che viene invocata ogni qualvolta si preme sul pulsante "concludi indagine" del form di inserimento. Prima di tutto carichiamo i dati dal form.

```
$("#concludi").click(function(){
    var idPaziente = $("#idPaziente").val().trim();
    var idCareprovider = $("#cpId").val().trim();
    var tipoValue = $("#tipoIndagine").val().trim();
    var motivoValue = $("#motivoIndagine_new").val().trim();
    var motivoAltroValue = $("#motivoAltro_new").val().trim();
    var careproviderValue = $("#careproviderIndagine_new").val().trim();
    var careproviderAltroValue = $("#careproviderAltro_new").val().trim();
    var statoValue = $("#statoIndagine_new").val().trim();
```

```

var centroValue = $("#centroIndagine_new").val().trim();
var refertoValue = $("#refertoIndagine_new").val().trim();
var allegatoValue = $("#allegatoIndagine_new").val().trim();
var dataMoment = $j("#data").data("DateTimePicker").date();
if(dataMoment != null && $("#data").val() != "")
    var dataValue = dataMoment.format("YYYY-MM-DD HH:mm:ss").toString();
else
    var dataValue = "";

```

Poi, in base allo stato dell'indagine, richiamiamo la funzione di validazione dei dati:

```

//CONTROLO DEL FORM
var formIsValid = false;
switch(statoValue){
    case "0":
        formIsValid = validateRichiesta(<parametri>);
        break;
    case "1":
        formIsValid = validateProgrammata(<parametri>);
        break;
    case "2":
        formIsValid = validateCompletata(<parametri>);
        break;
}

```

Infine, se il form è valido, inviamo i dati a *nuovaIndagine.php* tramite *POST*. Una volta che l'inserimento è stato completato la pagina viene ricaricata e riaperta sulla sezione Diario indagini diagnostiche per visualizzare i dati appena inseriti. Se il form invece non è valido, si mostra all'utente un messaggio di errore.

```

if(formIsValid){
    $("#formAlert_new").collapse();
    $.post("formscripts/nuovaIndagine.php",
        {
            idPaziente: idPaziente,
            idCare: idCareprovider,
            tipo: tipoValue,
            idMotivo: motivoValue,
            motivoAltro: motivoAltroValue,
            careprovider: careproviderValue,
            careproviderAltro: careproviderAltroValue,
            stato: statoValue,
            centro: centroValue,
            data: dataValue,
            referto: refertoValue,
            allegato: allegatoValue
        },
        function(status){
            $('#formIndagini')[0].reset();
            window.location.reload(); //RICARICO PAGINA PER AGGIORNARE VALORI
        });
}
else{
    $("#formAlert_new").show();
}
});

```

La funzione `$(document).on('click', "a.conferma", function()` viene invocata quando si preme su uno dei tasti *"Conferma modifiche"* presente sotto tutti i form di modifica. La struttura della funzione è identica a quella di `$("#concludi").click(function()` commentata sopra. L'unica differenza è quella che i dati non vengono inviati alla pagina *nuovaIndagine.php* ma alla pagina *modificaIndagine.php*

La funzione `$(document).on('click', "button.modifica", function ()` viene invocata quando si preme su uno dei tasti *"modifica indagine"*. I form di modifica sono già stati creati e compilati preventivamente dalla pagina *template_page_indagini.php* al momento del suo caricamento, questa funzione ha il solo scopo di renderli visibili.

```
/**
 * Funzione per click sul pulsante "Modifica indagine": presente su ogni riga:
 * apre un form sotto di esso per la modifica dei dati
 */
$(document).on('click', "button.modifica", function () {
    $(this).prop('disabled', true);
    $('#'+$(this).attr('id')+'.elimina').prop('disabled', true);
    var id = '#riga'+$(this).attr('id');
    $(id).show(200);
});
```

La funzione `$(document).on('click', "button.elimina", function ()` viene invocata quando si preme su uno dei tasti *"elimina indagine"*. La funzione provvede a recuperare l'id dell'indagine a cui fa riferimento il button e a inviare i dati alla pagina *eliminaIndagine.php*. Infine ricarica la pagina per aggiornare i dati visualizzati.

```
/**
 * Funzione per click sul pulsante "Elimina indagine": presente su ogni riga:
 * chiede conferma all'utente e in caso affermativo provvede a cancellare l'indagine
 */
$(document).on('click', "button.elimina", function () {
    if (confirm("Sei sicuro di voler eliminare l'indagine?")){
        $.post("formscripts/eliminaIndagine.php",
            {
                idIndagine: $(this).attr('id')
            },
            function(status){
                window.location.reload();
            });
    }
});
```

La funzione `$(document).on('click', "a.a-messaggio", function ()` viene invocata quando si preme sul tasto *"Messaggio FSEM"* presente sui vari centri indagini elencati nelle tabelle della sezione *"Centri indagini diagnostiche"*. Si occupa di recuperare il nome del careprovider responsabile del centro e inserirlo nel campo destinatario della finestra per l'invio di messaggi privati.

```
/**
 * Funzione per click sul pulsante "Invia messaggio privato" della tabella centri
 indagini:
 * apre una finestra di messaggio inserendo il nome del responsabile nel centro nel
 campo destinatario
 */
$(document).on('click', "a.a-messaggio", function () {
    var id = $(this).attr('id');
    var careprovider = document.getElementById("careproviderStudio" +
id).getAttribute('data-nome');
    $('#sendToUser').val(careprovider);
});
```


Il codice in questo file si occupa di ricevere i dati da inserire in database dal file indagine.js, controllarli e invocare le funzioni di utility.php per l'inserimento.

Prima di tutto recuperiamo l'id paziente e l'id del careprovider (se connesso) per svolgere dei controlli:

- Controlliamo che l'id del paziente che stiamo modificando sia effettivamente l'id del paziente connesso al sistema.
- Controlliamo se l'utente ha selezionato un careprovider registrato o ne ha inserito uno manualmente, nel primo caso, interroghiamo il database per recuperare nome e cognome del careprovider partendo dal suo id.
- Controlliamo se l'utente ha selezionato una diagnosi come motivazione o se ne ha inserita una manualmente. Nel primo caso, andiamo a recuperare il nome della diagnosi partendo dal suo id.
- Controlliamo se sono stati collegati referti e allegati.
- Controlliamo lo stato dell'indagine e invochiamo la funzione corretta.

```
if ($idPazienteConnesso == $idPaziente){
    if($careprovider != '') //se ho un ID per un careprovider registrato...
        //usando l'ID ne estraggo nome e cognome per l'inserimento in tabella
        $careproviderNome = getInfo('nome', 'careproviderpersona', 'id='.$careprovider) . ' ' .
            getInfo('cognome', 'careproviderpersona', 'id='.$careprovider);
    else{
        //altrimenti utilizzo il nome passato in input
        $careprovider = "NULL";
        $careproviderNome = $careproviderAltro;
    }
    //se ho un ID diagnosi per il motivo, ne estraggo il nome
    if($idMotivo != '') $motivo = getInfo('patologia', 'diagnosi', 'id='.$idMotivo);
    else{
        //altrimenti utilizzo la motivazione passata in input
        $idMotivo = "NULL";
        $motivo = $motivoAltro;
    }
    if($idReferto == '') $idReferto = "NULL";
    if($idAllegato == '') $idAllegato = "NULL";
    if($stato == "0"){
        echo nuovaIndagineRichiesta($idPaziente, $careprovider, $careproviderNome, $idMotivo,
            $motivo, 0, $tipo);
    }else if ($stato == "1"){
        echo nuovaIndagineProgrammata($idPaziente, $careprovider, $careproviderNome, $idMotivo,
            $motivo, 1, $tipo, $data, $centro);
    }else if ($stato == "2") {
        echo nuovaIndagineCompletata($idPaziente, $careprovider, $careproviderNome,
            $idMotivo, $motivo, 2, $tipo, $data, $centro, $idReferto, $idAllegato);
    }
}
else
    echo '<script>alert("Errore: il paziente in modifica non corrisponde al paziente
    connesso");</script>';
```

MODIFICAINDAGINE.PHP

In modificaIndagine.php operiamo gli stessi controlli di nuovaIndagine.php più un altro controllo per verificare che l'indagine che stiamo modificando appartenga realmente al paziente. Andiamo a recuperare questa informazione dal database. Una volta superati i controlli si invocano i metodi di update presenti in utility.php

```
// Prendo l'id del paziente a cui è assegnata l'indagine
$idPazienteIndagine = getInfo('idpaziente', 'indagini', 'id = ' . $idIndagine);

//Se l'idPaziente passato per la modifica è lo stesso del paziente connesso e del paziente
indagine...
if ($idPazienteConnesso == $idPaziente){
    if($idPazienteConnesso == $idPazienteIndagine){
        if($careprovider != '')
            $careproviderNome = getInfo('nome', 'careproviderpersona', 'id=' . $careprovider) .
            ' ' . getInfo('cognome', 'careproviderpersona', 'id=' . $careprovider);
        else{
            $careprovider = "NULL";
            $careproviderNome = $careproviderAltro;
        }
        if($idMotivo != '') $motivo = getInfo('patologia', 'diagnosi', 'id=' . $idMotivo);
        else{
            $idMotivo = "NULL";
            $motivo = $motivoAltro;
        }
        if($idReferto == '') $idReferto = "NULL";
        if($idAllegato == '') $idAllegato = "NULL";
        if($stato == "0"){
            echo modificaIndagineRichiesta($idIndagine, $careprovider, $careproviderNome,
            $idMotivo, $motivo, 0, $tipo);
        }else if ($stato == "1"){
            echo modificaIndagineProgrammata($idIndagine, $careprovider, $careproviderNome,
            $idMotivo, $motivo, 1, $tipo, $data, $centro);
        }else if ($stato == "2") {
            echo modificaIndagineCompletata($idIndagine, $careprovider, $careproviderNome,
            $idMotivo, $motivo, 2, $tipo, $data, $centro, $idReferto, $idAllegato);
        }
    }
    else
        echo '<script>alert("Errore: L\'indagine in modifica non appartiene al paziente
connesso");</script>';
}
else
    echo '<script>alert("Errore: il paziente in modifica non corrisponde al paziente
connesso");</script>';
```

ELIMINAINDAGINE.PHP

In *eliminaIndagine.php*, dopo aver ricevuto i dati da *indagini.js*, andiamo a controllare se l'indagine da eliminare appartenga effettivamente al paziente. Una volta superato il controllo invochiamo due funzioni da *utility.php*: con *modificaStatoIndagine* andiamo a impostare lo stato dell'indagine a "3", in questo modo l'indagine non sarà più visibile e allo stesso tempo non si perderanno possibili dati sensibili che potrebbero essere recuperati in caso di errore.

Infine, invochiamo la funzione *eliminaIndagine* che va a inserire un nuovo record nella tabella *indaginieliminate*, memorizzando l'id dell'utente che ha autorizzato l'eliminazione e l'id dell'indagine.

```
if($idPazienteConnesso == $idPazienteIndagine){
    modificaStatoIndagine($idIndagine, 3); //cambio lo stato dell'indagine senza
    cancellarla, ponendolo a 3
    eliminaIndagine($idIndagine, $id_prop); //inserisco l'id dell'indagine cancellata
    nella tabella indaginiEliminate
}
else
    echo'<script>alert("Errore: L\'indagine in eliminazione non appartiene al paziente
    connesso");</script>';
}
```