UNIVERSITÀ DI PISA

Corso di Laurea in Informatica Umanistica

RELAZIONE

Dagli esercizi sommativi agli esercizi formativi

Candidato: *Lavinia Salicchi*

Relatore: *Giuseppe Fiorentino*

Correlatore: *Daria Carmina Coppola*

Anno Accademico 2015-2016

Indice

1. Introduzione

L’oggetto di questo elaborato è un servizio on line di esercizi a scopo didattico.
Il principale obiettivo del lavoro è proporre un modello di esercitazione autovalutativa in contrapposizione al tradizionale principio di somministrazione di esercizi sommativi generici.
Il progetto si sviluppa in due direzioni: un lato docente (capitolo 3), dove poter caricare e modificare i quesiti, e un lato studente (capitolo 3), dove è possibile svolgere una batteria di esercizi assegnati, al termine della quale vi saranno ulteriori quesiti mirati al ripasso degli argomenti sui quali si è risultati più carenti. La logica e le tecniche con cui vengono generati i nuovi quiz, saranno discussi nel paragrafo 3.3.4.2.
Sebbene il proposito fosse quello di generare una struttura altamente adattabile ad ogni materia scolastica, per un test pratico dell’eserciziario, sono stati utilizzati test di ammissione ai corsi di laurea in Scienze.

2. Base concettuale

2.1 Esercizi sommativi e altri approcci

Nel tradizionale ed ormai ben radicato sistema di valutazione del nostro sistema scolastico, il metodo principale di esame dell'alunno e di esercitazione delle sue competenze sono gli esercizi sommativi.
L'aggettivo *sommativo*, derivante dal verbo *sommare*, richiama ai concetti di "complessivo" e "cumulativo". Per *esercizi sommativi* si intende, quindi, in docimologia, una valutazione globale espressa su un allievo durante un intero periodo di studio e che si concretizza in prove che prendono in esame la quasi totalità degli argomenti trattati durante un determinato periodo di lezioni.
Ad esempio: in seguito ad una serie di lezioni dedicate all'Illuminismo e alla rivoluzione francese, potrebbero essere assegnati esercizi riguardanti:

- le principali personalità del movimento filosofico sopracitato;
- i fondamenti teorici dell'Illuminismo (morali, sociali, politici);
- le date dei momenti cardine della rivoluzione francese;
- i rapporti tra Illuminismo e i moti di fine '700.

Un ripasso della materia impostato in tale modo, permetterà all'alunno di capire solo se ha *globalmente* compreso l'argomento o, tutt'al più, di intuire quale parte del programma preso in esame gli è lacunoso.
In realtà le capacità necessarie per il buon esito del test somministrato sarebbero potute essere:

- memorizzare date ed annessi eventi;
- memorizzare nomi;
- comprendere le dinamiche di causa-effetto.

Ne deriva che all'allievo sarà difficile trovare esercizi mirati al recupero delle sole conoscenze delle quali è più carente, in quanto la quasi totalità dei libri di testo non presentano quesiti mirati.
Il problema è quindi duplice: da una parte un'esercitazione su quiz sommativi non garantisce l'individuazione delle vere lacune di apprendimento, dall'altra, qualora queste venissero individuate da un'ottima capacità autoriflessiva dell'alunno, difficilmente verrebbero messi a disposizione dell'allievo i mezzi per un training di recupero mirato.

Il progetto su cui si concentra questo elaborato, nasce con l'intento di raccogliere quesiti dei quali specificare gli tutti gli argomenti presi in esame, le capacità coinvolte e il peso che questi elementi hanno nel quesito.
Per proseguire con l'esempio precedente, una possibile domanda potrebbe essere:
"*In che modo le teorie di Rousseau hanno influenzato l'operato di Maximilien de Robespierre?"*.
Un quesito simile, risulterebbe toccare i seguenti argomenti storici:

- Illuminismo;
- Rivoluzione francese;
- Rousseau;
- Robespierre.

Ma implicare in sé anche le competenze riguardanti:

- Rapporti causa-effetto;
- Personaggi storici;
- Filosofia.

Come modalità di risposta è stata ritenuta adatta la scelta multipla. Ogni opzione, se selezionata da parte dell'alunno, metterà in luce un diverso grado di acquisizione delle conoscenze. Vi sarà certamente una risposta totalmente corretta ma le restanti opzioni, anziché essere totalmente errate, potrebbero segnalare una buona conoscenza di solo alcuni degli argomenti sottintesi nel quesito.
Pertanto, le opzioni stesse devono contenere una valutazione pesata in relazione agli argomenti proposti dalla domanda.

Il servizio on-line che costituisce il progetto di questa tesi, prevede inoltre la proposta di esercizi di recupero incentrati sugli argomenti sui quali l'alunno si è dimostrato meno performante.
Le modalità con cui ciò può esser fatto sono due:

- far seguire ad ogni risposta il cui livello di correttezza non sia stato sufficiente, un quesito incentrato sull'argomento meno compreso;
- una prima batteria di esercizi, con una sottostruttura completa come indicato precedentemente, a cui far seguire l'attività di recupero in base alle risposte date.

I due approcci si rivolgono, idealmente, a due fasce di età differenti: per gli studenti più giovani, una immediata segnalazione dell'errore e conseguente esercitazione di recupero può aiutare nella comprensione delle proprie competenze e lacune, mentre per alunni più grandi (scuole superiori, ad esempio), possiamo utilizzare il secondo metodo, più improntato su una visione globale e al tempo stesso composita della materia, e che presuppone una buona capacità di analisi dei propri risultati, acquisita durante gli anni di studio.
Nell'eserciziario qui trattato, è stato scelto quest'ultimo approccio.
Viene proposto un set di quesiti, durante lo svolgimento dei quali l'andamento dell'alunno viene costantemente mostrato, attraverso una lista, in continuo aggiornamento, di argomenti trattati e relativa percentuale di successo. In questo modo lo studente ha piena coscienza del percorso che sta intraprendendo.
In seguito ad ogni risposta, viene proposta una pagina di feedback dove l'alunno potrà trovare richiami teorici circa gli argomenti trattati o brevi analisi

del quesito appena posto per aiutarlo nel ragionamento critico circa la risposta data.
Al termine della prima batteria, una lista argomento - risultati verrà proposta allo studente, ed egli stesso potrà scegliere quale argomento ripassare attraverso un nuovo ciclo di esercizi mirati.

3. Lezioni di Moodle e mantenimento dello stato

Uno strumento legato all’e-learning, che può costituire lo stato dell’arte di questo progetto, è l’attività di Moodle “Lezioni”.

3.1 Le Lezioni di Moodle

Moodle (acronimo di Modular Object-Oriented Dynamic Learning Environment, ambiente per l'apprendimento modulare, dinamico, orientato ad oggetti) è un ambiente informatico per la gestione di corsi, basato sull'ideologia costruzionista secondo la quale ogni apprendimento sarebbe facilitato dalla produzione di oggetti tangibili.
Tra le numerose attività e risorse che la piattaforma mette a disposizione troviamo le Lezioni. Questa attività è costituita da pagine HTML che si dividono nei tipi contenuti e domande.
Le prime possono presentare testo scritto e multimedia, mentre le seconde sono veri e propri quiz, le cui domande possono essere di sei tipi:

- Corrispondenza;
- Numerica;
- Risposta breve;
- Scelta multipla;

- Testo libero;
- Vero / Falso.

Ad ogni opzione proposta è possibile impostare un feedback, effettuare un reindirizzamento ad un'altra pagina della Lezione (che può essere di tipo "contenuto" e quindi avere nozioni teoriche) e assegnare un punteggio.
Sebbene l'attività Lezioni presenti un mantenimento di stato, questo è molto debole, e segnala all'utente, qualora quest'ultimo chiudesse il browser, solo l'ultima pagina visitata.

3.2 Mantenimento dello stato

Nel web si intende con "stato" l'insieme delle interazioni utente-applicazione. Mantenere lo stato significa quindi rendere utilizzabili queste informazioni per tutto il tempo di navigazione dell'user (la "sessione").
Ciò che è richiesto al server è di essere in grado di "accorgersi" che le richieste inviate provengano dalla stessa macchina. Le due caratteristiche principali che un sistema di gestione delle sessioni dovrebbe avere sono due: scoprire quando più chiamate provengano da uno stesso client e registrare conseguentemente i vari dati di quella sessione.
I principali metodi per mantenere lo stato sono:

- Querystring;
- Sessioni;
- Cookies.

Querystring e Sessioni

Con le *querystring* si ha il passaggio di informazioni tra due pagine attraverso la modifica della stringa dell'URL. Tipicamente si effettua nella forma

```
pagina.php?parametro1=valore1&parametro2=valore2
```

In cui il simbolo **?** introduce la querystring, **=** lega il nome del parametro passato al suo valore e **&** concatena le variabili tra di loro.

Le *sessioni* permettono di mantenere informazioni relative all'utente tra le varie pagine navigate senza necessitare dell'identificazione dell'utente.
Quando un user accede per la prima volta ad una pagina PHP impostata

per creare una sessione, attraverso la funzione `session_start()`, a questi viene associato un ID univoco che verrà utilizzato da PHP come chiave per recuperare l'array `$_SESSION` salvato specificamente per il determinato utente. L'ID è un codice univoco che, al momento della creazione della sessione, viene salvato automaticamente da PHP all'interno di un cookie oppure, se i cookie risultano disabilitati, accodato agli URL relativi generati dallo script PHP.
Sebbene le sessioni siano un buon metodo per mantenere lo stato di navigazione, poiché danno anche la possibilità di salvare qualunque tipo di dato, a differenza dei cookie che permettono di registrare solo stringhe e numeri, trovano il loro limite nel fatto che terminano nel momento in cui l'utente chiude il browser.
Pertanto i dati raccolti dalle sessioni dovrebbero essere salvati in un database per poter essere riutilizzati, causando così l'occupazione dello spazio della base di dati.

3.2.1 I Cookie

Un cookie è una coppia chiave/valore avente una data di scadenza ed un dominio di validità che viene salvata sul browser dell'utente ed inviata ad ogni pagina per poter accedere nuovamente ai dati precedentemente registrati.
L'utilizzo dei cookie presenta due principali benefici.
Il primo deriva dal fatto che il valore del cookie, che consiste in una stringa composta da lettere, numeri e simboli, viene salvato lato client, evitando così di occupare memoria sul server; il secondo dalla permanenza sul browser del cookie per un lasso di tempo che va oltre la singola sessione.

3.2.1.1 Creazione e modifica di un cookie

Un cookie viene creato, in una pagina PHP, grazie alla funzione `setcookie()`:

```
setcookie(name,value,expire,path,domain,secure,httponly);
```

dei parametri che è possibile inserire nella funzione, solo il primo, in cui specificare il nome del cookie, è obbligatorio.
Value indica il valore del cookie, *expire* è la durata espressa in secondi di

permanenza del cookie sul browser, *path* indica il percorso ed il valore "/" estende il cookie all'intero dominio, specificato nel campo *domain*. *Secure* specifica se il cookie dovrà essere trasmesso solo tramite una connessione sicura HTTPS ("TRUE") o meno ("FALSE" - valore di default). Se il campo *httponly* viene settato su "TRUE", il cookie sarà accessibile al solo protocollo HTTP e non da linguaggi di scripting.
La modifica dei valori dei parametri di un cookie avviene semplicemente richiamando nuovamente la funzione `setcookie()` e specificando nuovi valori.

3.2.1.2 Lettura dei valori dei cookie e sua eliminazione

Una volta registrate nei cookie informazioni che si desidera utilizzare nuovamente, è possibile accerdervi tramite la variabile superglobale `$_COOKIE`, all'interno della quale specificare il nome del cookie che vogliamo consultare.
Il dato restituito è, come detto in precedenza, una stringa. È pertanto possibile manipolarla tramite PHP, aggiornandola tramite concatenazione di caratteri, suddividerla tramite la funzione `explode()`, o sostituirla con una nuova stringa.
Per cancellare il cookie dal browser dell'utente, è possibile utilizzare la funzione `setcookie()`, impostando come negativo il valore di *expire*, ovvero una "scadenza" del cookie nel passato.

4. Progetto

In questo capitolo verranno trattati gli strumenti utilizzati per la realizzazione del progetto e la sua struttura e funzionalità in dettaglio.

4.1 Linguaggi

Il progetto su cui si concentra questa relazione è stato sviluppato utilizzando i seguenti linguaggi:

- HTML;
- CSS;
- JavaScript;
- PHP;
- SQL.

4.1.1 HTML

HTML, acronimo di HyperText Markup Language (in italiano “linguaggio a marcatori per ipertesti) è il linguaggio di markup solitamente usato per la formattazione e impaginazione di documenti ipertestuali disponibili nel World Wide Web sotto forma di pagine web.
HTML ha come scopo quello di gestire i contenuti, specificandone allo stesso tempo la struttura grafica (layout) all'interno della pagina web da realizzare grazie all'utilizzo di tag diversi. Ogni tag (ad esempio <h1> o <p>) specifica

un ruolo dei contenuti che esso contrassegna (<h1> definirà le intestazioni mentre <p> i paragrafi). I browser che leggono il codice mostrano all'utente formattazioni predefinite per ogni tag che incontrano (i contenuti marcati con il tag <h1> avranno carattere 18pt e i contenuti marcati da <p> avranno carattere 12pt). Queste specifiche di formattazione possono essere tuttavia modificate dallo sviluppatore grazie all'utilizzo del CSS.

4.1.2 CSS

CSS, acronimo di Cascading Style Sheets (in italiano "fogli di stile a cascata") è un linguaggio usato per definire la formattazione di documenti HTML, XHTML e XML.
Gli elementi presenti in un foglio di stile CSS sono strutturati secondo lo schema:

```
selettore{
  proprietà:valore;
}
```

I selettori possono essere:

- di tipo (ad esempio "body" o "h1");
- di classe (come ".classe_1");
- di identificatore ("#nome_identificatore");
- di pseudo-classe ("p:first-child");
- di pseudo-elementi ("p:first-line");
- di gerarchia ("div > p");
- di attributo ("a[title=Titolo]").

4.1.3 JavaScript

JavaScript è un linguaggio di scripting orientato agli oggetti e agli eventi, comunemente utilizzato nella programmazione Web lato client per la creazione, in siti web e applicazioni web, di effetti dinamici interattivi tramite funzioni di script invocate da eventi innescati dall'utente che naviga sulla pagina.

4.1.4 PHP

PHP, acronimo ricorsivo di PHP: Hypertext Preprocessor (in italiano “preprocessore di ipertesti”) è un linguaggio di scripting interpretato, originariamente concepito per la programmazione di pagine web dinamiche. Uno dei principali utilizzi del PHP è quello di interfacciarsi con i database, attraverso l’inserzione di stringhe SQL.

4.1.5 SQL

SQL, acronimo di Structured Query Language (in italiano “linguaggio di interrogazioni strutturate”) creato per l’interrogazione e modifica di database. I principali comandi con cui interagire con un database sono:

- SELECT: per estrarre i dati;
- INSERT: per inserire nuovi dati;
- UPDATE: per aggiornare i dati;
- DELETE: per eliminare dati.

I principali elementi di sintassi sono:

- FROM: a seguito del quale specificare il nome della tabella su cui effettuare l’interrogazione o modifica (utilizzato dai comandi SELECT e DELETE);
- WHERE: per specificare eventuali vincoli sui campi delle tabelle così da affinare l’interrogazione o selezione del record da modificare (utilizzato dai comandi SELECT, UPDATE e DELETE);
- INTO e VALUES: elementi del comando INSERT, per specificare rispettivamente la tabella (e i rispettivi campi) in cui inserire i dati e i valori che questi dovranno assumere;
- SET: elemento del comando UPDATE, a seguito del quale specificare quali valori devono assumere i campi da modificare, attraverso la struttura `campo=valore`.

SQL utilizza operatori che possono essere:

- di assegnazione;
- di confronto;
- stringa;
- aritmetici;

- condizionali;
- logici;
- tra bit.

4.2 Software, applicazioni web e librerie

Nello sviluppo del progetto sono stati utilizzati alcuni software, applicazioni web e librerie JavaScript.

4.2.1 Software

4.2.1.1 XAMPP

XAMPP, acronimo di Apache, MariaDB (precedentemente:MySQL), PHP, Perl e la cui X iniziale significa “cross-platform”, è una piattaforma software che offre una serie di strumenti, server web e PHP per la realizzazione e la gestione di siti web.
Il software mette a disposizione un’interfaccia di controllo grazie la quale è possibile, tra le varie opzioni disponibili, selezionare i moduli da attivare, configurarli e visualizzare i logs (registrazioni sequenziali e cronologiche delle operazioni effettuate, da un utente, un amministratore o automatizzate, man mano che vengono eseguite dal sistema o applicazione).

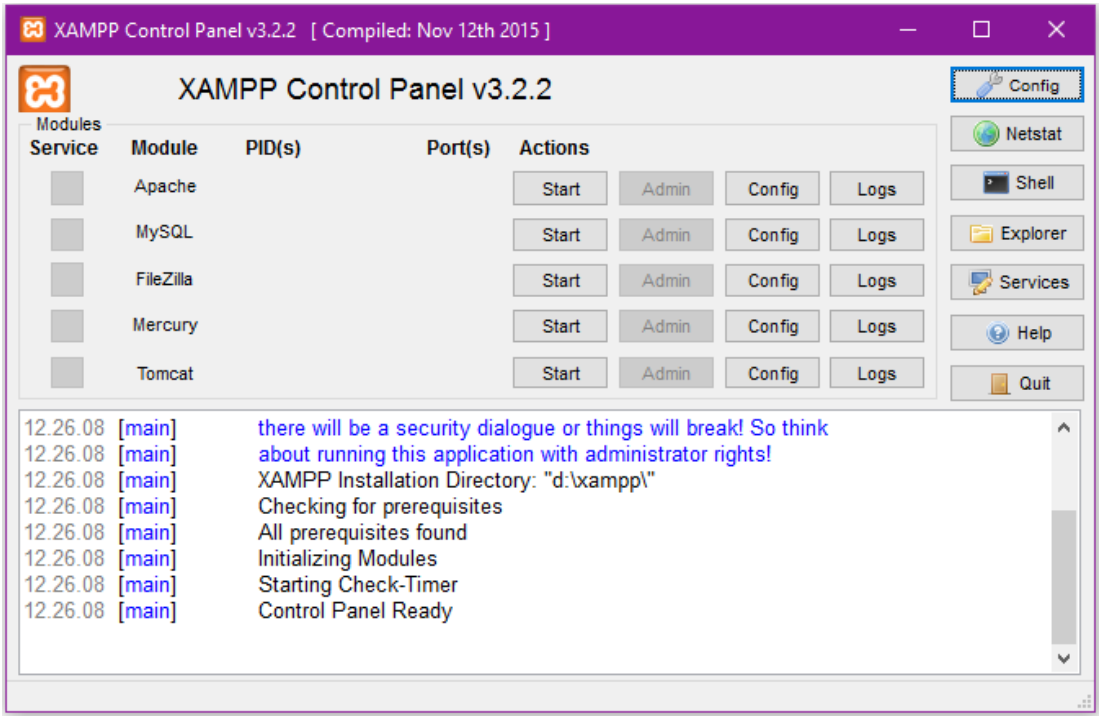

Fig1 - Pannello di controllo di XAMPP.

4.2.2 Applicazioni web

4.2.2.1 phpMyAdmin

phpMyAdmin è un'applicazione web scritta in PHP grazie alla quale è possibile gestire un database MySQL tramite browser: possiamo infatti creare, modificare, cancellare intere tabelle o singoli record; fare un backup dei dati contenuti. In phpMyAdmin vi è la possibilità di importare database, in vari formati, tra cui CSV, SQL e XML.

4.2.3 Librerie JavaScript

4.2.3.1 jQuery

jQuery è una libreria JavaScript per applicazioni web. Nasce con l'obiettivo di semplificare la selezione, la manipolazione, la gestione degli eventi e l'animazione di elementi DOM in pagine HTML, nonché implementare funzionalità AJAX.

JQuery offre una vasta gamma di funzionalità, tra le quali ricordiamo:

- inserire o eliminare elementi nelle pagine HTML;
- manipolare lo stile degli elementi;
- gestire gli eventi (come ad esempio ".click");

- aggiungere o modificare gli attributi degli elementi;
- gestire le chiamate asincrone grazie agli eventi AJAX o l'interazione con file JavaScript per caricare un oggetto JSON.

La selezione di un elemento HTML può avvenire in base al suo id (`#id_elemento`), alla classe di appartenenza (`.classe`), al valore di un attributo (`[value = "val"]`), ad una pseudo-classe (`:first`) e alla gerarchia degli elementi (`sibling`).

4.2.3.2 TinyMCE

TinyMCE è una libreria JavaScript per l'implementazione di un ricco editor di testi su pagine web.
Grazie a questo editor è possibile formattare il testo e inserire elenchi, puntati e numerati, tabelle, immagini, video, caratteri speciali più utilizzati.
TinyMCE offre la possibilità di aggiungere funzionalità all'editor grazie a plugin sviluppati dalla stessa Ephox Corporation o da terzi.
La gestione dell'interfaccia è anch'essa personalizzabile, attraverso la modifica delle entità costituenti la funzione JavaScript che genera l'editor: è possibile, ad esempio, modificare le dimensioni della finestra, caricare un tema per dare un aspetto diverso all'editor, eliminare o inserire nuovi elementi della toolbar.

4.3 Struttura del progetto e lavoro svolto

Il progetto preso in esame si divide in due parti: un lato docente, dove l'insegnante può impostare domande, risposte e argomenti, e un lato studente, dove l'alunno può esercitarsi.
La comunicazione fra questi due elementi avviene grazie ad un database, gestito con phpMyAdmin.

4.3.1 I Database

Per database si intende una collezione di dati organizzati e strutturati. I software per la gestione dei database sono indicati con il termine DBMS, acronimo di DataBase Management System.

Un sistema di gestione di basi di dati è un sistema software in grado di gestire collezioni di dati che siano grandi, condivise e persistenti, assicurando la loro affidabilità e privatezza. Come ogni prodotto informatico, un DBMS deve essere efficiente ed efficace.

4.3.1.1 Struttura del database

Il database del progetto presenta sei tabelle, qui rappresentate mediante sintassi SQL:

TABLE argomenti
{*idA* integer **PRIMARY KEY**,
descr varchar(50)}

Raccoglie gli argomenti contenuti nelle domande. Ad ogni argomento è associato un ID (*idA*) e i valori di quest'ultimo campo si incrementano in modo automatico all'inserimento di un nuovo record.

TABLE pagine
{*id* integer **PRIMARY KEY**,
testo blob}

Raccoglie i testi delle domande, ad ognuna delle quali è assegnato un ID (*id*). I valori nel campo *id* sono auto-incrementali.

TABLE link
{*n_opzione* integer **PRIMARY KEY**,
opzione varchar(100),
dest int(3),
FOREIGN KEY *sorg* integer **REFERENCES** pagine(id)}

Raccoglie i testi delle opzioni (*opzioni*), ad ognuna delle quali è assegnato un ID (*n_opzione*), campo auto-incrementale. Il campo *sorg* indica la domanda cui fa riferimento l'opzione, attraverso una foreign key collegata al campo *id* della tabella *pagine*

TABLE pesi
{**FOREIGN KEY** *idQ* integer **REFERENCES** pagine(id),
FOREIGN KEY *idA* integer **REFERENCES** argomenti(idA),

peso int(11)}

Tabella di collegamento tra le tabelle *pagine* e *argomenti*, che specifica il peso che ogni argomento assume nella domanda in cui è presente.

TABLE pesi_opz

{**FOREIGN KEY** *idO* integer **REFERENCES** link(n_opzione),
FOREIGN KEY *idA* integer **REFERENCES** argomenti(idA),
peso int(11)}

Tabella di collegamento tra le tabelle *link* e *argomenti*, che specifica il grado di conoscenza che si ha di un determinato argomento scegliendo una determinata opzione.

TABLE feedback

{*idF* integer **PRIMARY KEY**,
testo blob,
FOREIGN KEY *idQ* integer **REFERENCES** pagine(id)}

Tabella che raccoglie i testi delle pagine di feedback. Ogni feedback corrisponde ad una domanda.

TABLE utenti

{*idUser* integer **PRIMARY KEY**,
username varchar(30),
password varchar(255),
tipo varchar(1)}

Tabella che raccoglie i dati degli utenti che si registrano al servizio. Ad ogni utente viene assegnato un ID (*idUser*, campo auto-incrementale), memorizzati username e password e, all'interno del campo *tipo*, si specifica se si tratta di un docente o uno studente.

4.3.2 Registrazione e login

Il progetto si apre con una pagina in cui sia possibile registrarsi al servizio online o, se si dispone già di un account, effettuare il login.

Al momento della registrazione vi è la possibilità di specificare il proprio ruolo: all'interno del form contenente i campi in cui inserire username e

password, vi è una checkbox, spuntando la quale si viene registrati come docenti.

```
<form action="registrazione.php" method="post">
      <input type="text" name="utenteReg" size="40"><br />
      <input type="password" name="passwordReg" size="40"><br />
      <input type="checkbox" name="tipo" value="d"/> Sono un docente
      <input type="submit" value="Conferma">
</form>
```

La specificazione dei ruoli è indicata nel campo *tipo* della tabella *utenti*. Spuntando la casella, infatti, viene assegnato il valore “d” a tale campo, altrimenti viene assegnato il valore “s”. La differenziazione dei ruoli implicherà l’accesso al lato docente o al lato studente.
Dopo l’inserimento dei dati da parte dell’utente, la registrazione, tramite file PHP, avviene attraverso l’inserimento di un nuovo record all’interno della tabella *utenti*, mediante una query del tipo:

```
INSERT INTO utenti (username,password,tipo)
VALUES('$NewUser','$hashpw','$tipo')
```

Del campo *tipo* è stato già parlato poc’anzi.
`$NewUser` è la variabile a cui è stato assegnato il valore dell’input di tipo text denominato “utenteReg”, ed ottenuta dal file *registrazione.php* grazie all’utilizzo della variabile superglobale $_POST.
`$hashpw` è la variabile a cui è stato assegnato il risultato della funzione `password_hash()`, alla quale era stata passata la variabile `$NewPw`, ottenuta dall’input di tipo password denominato “passwordReg” grazie a $_POST.
La funzione `password_hash()`, viene utilizzata per creare una versione criptata della password. In generale per funzione crittografica di hash si intende un algoritmo matematico che converte dati di dimensioni arbitrarie in stringhe di dimensioni fisse. Questo tipo di algoritmo è ideato per essere “one-way”, ovvero impossibile da invertire: ciò lo rende particolarmente adatto al generare versioni salvabili in database delle credenziali degli utenti.

Le password registrate dopo aver subito la funzione di hash, richiedono, per essere verificate al momento del login la funzione `password_verify()`:

```
password_verify ($password, $hash)
```

Dove `$password` sarà la password inserita dall'utente che vuole effettuare il login e `$hash` la stringa salvata nel campo *password*, ovvero la password dell'utente che aveva subito la funzione di hash al momento della registrazione.
L'ultima fase della registrazione consiste nella creazione di un cookie avente per nome l'ID che è stato assegnato all'utente, come valore il simbolo "/", come tempo di vita 30 giorni e come path l'intero dominio.
Al termine della registrazione l'utente viene reindirizzato all'area di sua competenza.

4.3.3 Lato docente

Per permettere l'inserimento di domande da parte del docente, senza che questi debba relazionarsi direttamente con il database, è stata creata un'interfaccia per guidare la creazione dei quesiti.

4.3.3.1 Gestione domande

Il lato docente si apre con un riepilogo delle domande inserite nel database. Ad ogni quesito elencato è affiancato un bottone "Visualizza", che permette l'editing delle domande: testo, argomenti, opzioni e feedback.
In coda alla lista di esercizi è presente un bottone "Aggiungi", che porterà alla creazione, guidata passo-passo, di un nuovo quesito.

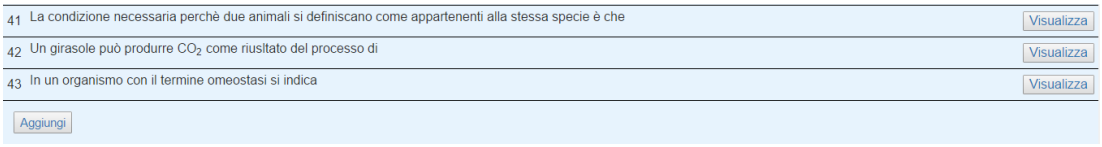

Fig2 - Tre domande tratte dalla pagina di gestione dei quesiti.

La modifica dei testi delle domande presenta il testo della domanda (caricato, attingendo dal database, allo stesso modo in cui vengono stampati i quesiti

proposti nel lato studente, che verrà analizzato in dettaglio nella sezione **), un bottone *Modifica* ed un bottone *Prosegui*.
Cliccando su *Prosegui*, il quesito non viene modificato, per passare quindi alla pagina successiva, in cui è possibile modificare gli argomenti.
Cliccando su *Modifica*, in corrispondenza del testo della domanda viene caricato l'editor di testo TinyMCE, contenente il quesito ora modificabile.
Ciò è possibile grazie a jQuery:

```
$("#dommod").click(function() {
      var domOri=$("#testo").text();
      var editableText = $('<textarea
            id="mytextarea">'+domOri+'</textarea>');
      $("#testo").replaceWith(editableText);
```

Viene selezionato il bottone il cui id è *dommod*, l'evento .click attiva una funzione all'interno della quale viene registrato nella variabile *domOri* il contenuto del div *testo*. Tale variabile sarà inserita in una textarea chiamata *mytextarea*. Il div *testo* viene quindi sostituito dalla textarea appena definita, mediante la funzione `replaceWith()`.

```
      tinymce.init({
            selector: 'textarea',
            height: 500,
            plugins: [
                  'advlist autolink lists link image charmap
                        print preview anchor',
                        'searchreplace visualblocks code
                        fullscreen',
                        'insertdatetime media table contextmenu
                        paste code'
            ],
            toolbar: 'insertfile undo redo | styleselect |
                        bold italic | alignleft aligncenter
                        alignright alignjustify | bullist
                        numlist outdent indent | link image',
            content_css:
```

```
                //www.tinymce.com/css/codepen.min.css'
    });
```

La funzione in analisi chiama a sua volta `tinymce.init()`, reperibile nella documentazione dell'editor. All'interno di questa funzione viene specificato il selettore, ovvero dove caricare l'editor. I selettori possono essere id, classi, o, come in questo caso, tipi.
La funzione chiamata dal click sul bottone *dommod* porta ulteriori modifiche al documento HTML:

```
var newbut= $('<button id="gostampa">Anteprima</button>');
    $("#dommod").replaceWith(newbut);
    $("#gostampa").click(function() {
        tinyMCE.triggerSave();
        var str = $('#mytextarea').val();
        var stmp = htmlUnescape(str);
        var save = htmlEscape(str);
        $("#stampaDomanda").prop('action',
'api/fromeditorMod.php?id='+iddom+'&user='+userid);
        $("#stampaDomanda").prepend('<input type="hidden"
name="testo" value="'+save+'">'+stmp);
        $("#stampaDomanda").append('<input type="submit"
id="salva" value="Salva" /> ');
});
```

Un nuovo bottone, *gostampa* che si presenta all'utente come *Anteprima*, permette di salvare il contenuto dell'editor, che viene salvato nella variabile denominata *str*.
Poiché la formattazione dei contenuti generati in TinyMCE si esprime in HTML, *str* conterrà caratteri estremamente "delicati", quali virgolette, apici e segni di minore e maggiore. Per permettere sia una corretta visualizzazione dell'anteprima del testo sia di salvare la domanda e relativo codice di formattazione in modo adeguato, si è fatto ricorso alle funzioni `htmlUnescape()` e `htmlEscape()`:

```
function htmlEscape(str) {
    return str
```

```
        .replace(/&/g, '&amp;')
        .replace(/"/g, '&quot;')
        .replace(/'/g, '&#39;')
        .replace(/</g, '&lt;')
        .replace(/>/g, '&gt;');
}
function htmlUnescape(str){
    return str
        .replace(/&quot;/g, '"')
        .replace(/&#39;/g, "'")
        .replace(/&lt;/g, '<')
        .replace(/&gt;/g, '>')
        .replace(/&amp;/g, '&');
}
```

i risultati delle quali sono stati assegnati come valori rispettivamente di *stmp* e *save*. Queste ultime due variabili sono state inserite all'interno del form *stampaDomanda*, inserendo la prima come testo visibile, la seconda come valore dell'attributo value dell'input *save*, inserito in HTML tramite jQuery, di tipo hidden.
A chiusura della funzione è stato poi generato un bottone *salva* di tipo submit, che invierà, tramite il form *stampaDomanda*, i dati al file fromeditorMod.php, al quale sono passati l'id della domanda e l'username dell'utente tramite querystring.
Al file PHP spetta il compito di aggiornare il database, attraverso una query:

```
UPDATE pagine SET testo='$str' WHERE id='$id'
```

Dove *str* e *id* sono le informazioni passate rispettivamente dalla form, tramite variabile superglobale $_POST e dalla querystring, utilizzando la variabile superglobale $_GET.
La pagina effettua quindi un redirect automatico alla fase successiva della modifica dell'esercizio, ovvero il settaggio degli argomenti, attraverso la funzione `header()`:

```
header("Location:../modificaargomenti.php?id=$id&user=$user")
```

4.3.3.2 Gestione argomenti

La pagina per la gestione degli argomenti inclusi nei quesiti è composta da una lista degli argomenti assegnati e relativi pesi, la medesima lista espressa tramite una serie di select, un bottone *Aggiungi argomento* ed un bottone *Prosegui.*

La lista testuale con elencati gli argomenti già assegnati, è creata interrogando il database tramite query SQL contenuta in un file PHP, grazie al quale è stato possibile specificare l'id della domanda, utilizzato per affinare la select, dalla quale verranno estratti l'id dell'argomento, il peso che ogni argomento ha all'interno della domanda e la descrizione dell'argomento:

```
SELECT DISTINCT p.$peso AS peso, a.$idA AS id, a.$d AS des
                FROM $arg AS a
                INNER JOIN $pesi AS p ON p.$idAp = a.$idA
                WHERE p.$idQ = $id"
```

successivamente sono stati inseriti i risultati della query in un array multidimensionale, passato a jQuery tramite funzione `json_encode()`, e da lì passato come variabile ad una funzione.

In jQuery l'array è stato quindi scansionato tramite ciclo for, ed ogni elemento, sia di indice 'peso' che di indice 'id', è stato inserito all'interno di una stringa a sua volta aggiunta come elemento HTML, all'interno del div *listaarg*:

```
for(i=0; i<data.length; i++){
      $("#listaarg").append('<li>Argomento:
"'+data[i]['des']+'" - Peso: '+data[i]['peso']+'</li>');
...}
```

I select sottostanti utilizzano il medesimo array:

```
...
var arrEl = [];
```

```
$('.arg').each(function(){
      arrEl.push( $(this).attr('id') );
});
$("#dati").append('<select class="arg"
name="modarg'+(arrEl.length+1)+'"
id="modarg'+(arrEl.length+1)+'">');
$("#modarg"+(arrEl.length+1)).append('<option
value="elimina">Elimina</option></select>');
$("#modarg"+(arrEl.length+1)).append('<option
selected="selected"
value="'+data[i]['id']+'">'+data[i]['des']+'</option>');
...}
```

Ogni select, inserita nel form *dati*, deve assumere un nome diverso. Ciò è stato ottenuto facendo seguire alla stringa “modarg” un numero, che indica la lunghezza dell’array *arrEl* aumentata di un’unità. L’array *arrEl* è costituito da tutti gli elementi di classe *arg* e, impostando *arg* come classe di ogni select, la lunghezza di *arrEl* si aggiorna ad ogni iterazione.
Di pari passo avviene la creazione dei select per i pesi che ogni argomento assume nella domanda, le cui option sono inserite grazie ad un for annidato che scorre un’array `arrPesi = [1,2,3,4];`
Ogni argomento può quindi avere un peso che va da 1 (importanza marginale nella domanda) a 4 (necessità di una buona conoscenza dell’argomento).
Cliccando sul bottone *Aggiungi argomento* viene generata una select contenente tutti gli argomenti presenti nella tabella *argomenti* del database. Qualora l’argomento che si desidera aggiungere alla domanda non fosse già presente nel database, è prevista l’opzione ***Nuovo***, che, se selezionata, porterà alla trasformazione della select in un input di tipo *text*, il cui nome inizia con la stringa *editarg* concatenata ad un numero il cui valore segue la medesima logica dei *modarg*, in cui poter editare un nuovo argomento.
Cliccando sul bottone *Salva*, generato nel momento in cui vi sia un evento .change su una select, vengono generati due input di tipo hidden.
Il primo ha come valore una stringa contenente tutti gli ID (uguali ai name)

degli elementi di classe *arg*, mentre il secondo ha come valore una string contenente tutti gli ID degli elementi di classe *pes* (ovvero le select dei pesi). Il file PHP a cui questi dati sono inviati ha il compito di:

- sciogliere le stringhe contenenti i nomi dei campi attraverso la funzione `explode()`;
- riconoscere le select *modarg* attraverso la funzione `substr()` impostata per estrarre i primi sei caratteri di ogni nome;
- aggiornare i dati provenienti dai *modarg* nel database tramite query SQL con comando UPDATE;
- inserire i testi provenienti dagli *editarg* nel campo *descr* della tabella *argomenti*
- inviare i dati sui pesi dei nuovi argomenti ad un altro file PHP, il quale otterrà l'*idA* assegnato ad ogni nuovo argomento per utilizzarlo nell'INSERT nella tabella *pesi*.

4.3.3.3 Gestione opzioni

Dopo la gestione degli argomenti, viene presentata al docente la gestione delle opzioni. La pagina si presenta come una lista delle opzioni assegnate alle domande (div di classe *opz*) e per ognuna, una sottolista degli argomenti assegnati alla domanda. Ogni elemento della sottolista è affiancato da una select contenente la percentuale di conoscenza che la risposta dimostra dell'argomento. Le percentuali vanno dallo 0% al 100% con un intervallo di 25 unità tra un valore e l'altro.
Al termine della lista vi sono i bottoni *Modifica*, *Aggiungi* e *Prosegui*.
Le opzioni sono inserite con le stesse modalità con cui sono inseriti gli argomenti assegnati alla domanda, ma gli array scorsi da jQuery sono frutto di tre interrogazioni differenti al database:

1. la prima per estrarre tutte le opzioni assegnate alla domanda (dalla tabella *link*);
2. la seconda per estrarre tutti gli argomenti assegnati alla domanda (dalla tabella *pesi* con una INNER JOIN alla tabella *argomenti*);
3. la terza per estrarre i pesi che ogni opzione ha per ogni argomento (dalla tabella *pesi_opz* con una INNER JOIN alla tabella *pesi*).

Inserendo ogni risultato in un array e combinando i dati attraverso cicli for annidati, vengono generate stringhe da stampare nella pagina HTML.
Cliccando su *Modifica* in corrispondenza di ogni opzione elencata si aprirà un editor, similmente a quanto accadeva durante la modifica del testo della domanda. In questo caso, però, il selettore non sarà per tipo ma per id, in modo da creare una relazione univoca “testo presente - testo modificabile”:

```
...
<textarea class="txtopz" id="textarea'+i+'">
...
selector: '#textarea'+i+'',
...
```

In cui la variabile *i* è l’indice dell’array *arrOpz*, i cui elementi sono gli ID di tutti gli elementi di classe *opz*.
Cliccando su *Aggiungi* si passa ad una pagina composta da una textarea e un bottone *Apri editor*, cliccando sul quale si aprirà un editor in cui poter inserire il testo della nuova opzione.
Cliccando sul bottone *Anteprima* viene stampato il testo appena editato, similmente a quanto accadeva durante la modifica della domanda, a seguito del quale verrà chiesto di effettuare una scelta: se aggiungere una nuova opzione o terminare la modifica.
Nel primo caso l’opzione appena editata viene salvata nel database e la pagina corrente ricaricata; nel secondo l’opzione viene salvata e vi è un redirect verso la pagina di provenienza.
Cliccando su *Salva* le informazioni circa testi delle opzioni, loro id, argomenti ed annessi pesi, vengono passate tramite il metodo POST ad un file PHP sotto forma di stringhe.
Il file PHP incaricato di aggiornare il database ha un comportamento simile al file PHP che gestiva gli argomenti.

4.3.3.4 Feedback

L’ultima fase della modifica di una domanda è la gestione del feedback generico della domanda. La pagina è strutturata come la sezione per la modifica del testo del quesito: testo e possibilità di modificarlo tramite editor.

4.3.3.5 Aggiungere domande

Nella pagina che apre il lato docente del servizio online è presente il bottone *Aggiungi*. Cliccando su tale bottone si accede ad una pagina per editare il testo di una nuova domanda, che presenta come elementi una textarea e un bottone *Apri editor*, che farà caricare l'editor in corrispondenza della textarea. Successivamente le pagine per aggiungere gli argomenti, le opzioni e i feedback, sono le medesime presentate per la fase di modifica.

4.3.4 Lato studente

Come già anticipato, il lato studente del progetto è il vero e proprio eserciziario: una prima batteria di esercizi seguiti da esercizi di recupero mirati.

Fig3 - Quesito con relative opzioni e andamento dello studente

4.3.4.1 Prima batteria

La prima batteria di esercizi propone quesiti scelti in modo casuale dalla raccolta di domande presente sul database. Attraverso una query SQL in PHP, vengono estratti tutti gli *id* dei record dalla tabella *pagine*, e i risultati inseriti in un array:

```
$arrId = select($mysqli, "SELECT id AS idQ FROM `pagine`");
```

Attraverso la funzione `array_rand()` viene estratto un indice casuale dall'array *arrId*. La domanda proposta allo studente sarà dunque quella corrispondente all'id estratto.
Le opzioni vengono inserite, come input di tipo radio e aventi come valore il proprio ID, in un form con metodo POST attraverso una stampa jQuery nell'HTML, previa query SQL in PHP che interroga la tabella *link*, estraendo le risposte che hanno come *sorg* l'ID della domanda.
Alla scelta dell'alunno, verranno passate ad un file PHP tramite querystring lo username dell'utente, tramite POST l'ID dell'opzione selezionata ed i dati della domanda.
Quest'ultimi sono contenuti in una stringa che si presenta nella forma `arg`*idargomento*/`pes`*pesoargomento*/ dove l'ID dell'argomento ed il suo peso all'interno della domanda sono frutto di una query che interroga la tabella *pesi* estraendo i record nei quali il campo *idQ* ha lo stesso valore dell'ID della domanda.
Il file PHP action del form, ha il compito di dividere la stringa contenente le informazioni sulla domanda, attraverso la funzione `explode()` in cui il separatore è il simbolo / , analizzare i primi tre caratteri di ogni sezione, attraverso la funzione `substr()`: se i primi tre caratteri corrispondono ad "arg", il numero che segue sarà l'ID dell'argomento, altrimenti, il numero si riferisce al peso. Le due serie di porzioni così riconosciute vengono inserite in due array distinti di uguale lunghezza e legati dal fatto che all'indice dell'array contenente gli ID degli argomenti, corrisponde il peso impostato nella domanda corrente per il tale argomento nell'array contenente i pesi.
Attraverso una query sulla tabella *pesi_opz*, vengono estratti i dati sull'opzione scelta dall'alunno e combinati ai dati estratti dalla stringa.
Ne risulta una nuova stringa contenente i campi:

- *dom*: con ID della domanda;
- *arg*: con ID dell'argomento;
- *pun*: con i punti ottenuti dell'alunno per l'argomento specificato precedentemente. Questo valore è ottenuto dividendo i punti assegnati all'argomento nella domanda corrente per 100 e

moltiplicando questo valore per il peso (valore percentuale) che l'opzione ha per il tale argomento;

- *tot:* con il peso che l'argomento ha nella domanda.

I campi sopra elencati sono separati dal simbolo "/" e la stringa si concatena a sé stessa grazie ad un ciclo for sull'array contenente i pesi che l'opzione ha per ogni argomento (della medesima lunghezza, quindi dei due array già analizzati). La stringa risultante alla fine dell'iterazione viene poi inserita come valore del cookie avente come nome l'id dell'utente.
Poiché il file PHP che gestisce questa elaborazione di informazioni viene richiamato dopo ogni quesito della prima batteria, la stringa creata in modo iterativo si concatena al valore del cookie, precedentemente richiamato ed associato ad una variabile:

```
$strCookie = $_COOKIE[$nome];
for($i = 0; $i < count($records); $i++){
      $punti = ($indexP[$i]/100)*$records[$i]['peso'];
      $strCookie.=
"dom:".$id."/arg:".$records[$i]['arg']."/pun:".$punti."/tot:".
$indexP[$i]."/";
}
```

Il numero di quesiti della prima batteria è regolato da un controllo sui campi *dom* presenti nel cookie. Ogni campo che inizia con "dom" è seguito dall'ID della domanda, e ripetuto tante volte quanti sono gli argomenti collegati alla domanda. Per sapere quante domande sono state fatte, le sezioni *dom* vengono riconosciute, inserite in un array e su tale array viene applicata la funzione `array_unique()` ed il nuovo array, privo di ripetizioni viene salvato in una variabile denominata *cont*. Su *cont* avviene il controllo finale del file:

```
if(sizeof($cont) == 20){
      header("Location: ../risultati.php?user=$user");
}
else{
      header("Location: ../feedback.php?user=$user&id=$id");
}
```

Se la lunghezza dell'array arriva a 20, ad esempio, vi sarà un redirect alla pagina dei risultati, altrimenti a quella del feedback di domanda.
La pagina di feedback presenta i contenuti teorici caricati dall'insegnante ed un bottone *Prosegui*, che permetterà di proseguire il quiz.
Ogni volta che si viene reindirizzati sulla pagina dei quesiti, l'andamento posto in basso sarà aggiornato.
La pagina dei risultati si presenta divisa in due sezioni. La prima vede elencati tutti gli argomenti con rispettive percentuali di successo, la seconda propone una lista di argomenti per i quali si è risultati più carenti, ognuno dei quali dispone di un bottone *Ripassa*, che, se cliccato, permette di accedere alla batteria di esercizi di recupero contenenti l'argomento desiderato.
Ciò è possibile perché ogni elemento dell'elenco formato da argomento-bottone è racchiuso in un form contenente al suo interno un input di tipo hidden, con name *argRecupero*, il cui valore è l'ID dell'argomento, seguito da un ulteriore input di tipo hidden con name *percRecupero*, il cui valore è la percentuale di successo ottenuta nella prima batteria di esercizi.

4.3.4.2 Batteria di recupero

La pagina degli esercizi di recupero è impostata allo stesso modo di quella preliminare. Si differenzia solo dal fatto che le domande sono frutto di query sulla tabella *pesi* in cui:

- il valore del campo *idA* deve corrispondere all'argomento che si desidera ripassare (ottenuto dal file tramite $_POST dal form della pagina dei risultati);
- il valore del campo *peso* deve essere compreso tra due valori, scelti in base alla percentuale di successo ottenuta nella prima batteria.

```
SELECT idQ AS id, peso FROM pesi WHERE idA=$karg AND
(peso=$min OR peso=$max)
```

I valori *min* e *max* sono impostati tramite una serie di if:

```
if($perc == 0){
```

```
            $min=$max=1;
        }
        if ($perc == 25){
            $min = 1;
            $max = 2;
        }
        if ($perc == 50){
            $min = 2;
            $max = 3;
        }
        if ($perc == 75){
            $min = 3;
            $max = 4;
        }
```

Quest'ultimo affinarsi della ricerca è dovuto al fatto che, poiché i pesi assegnati agli argomenti implicano diversi livelli di competenza, può risultare inutile somministrare esercizi considerabili "troppo semplici" a chi ha ottenuto risultati medio-alti, così come sottoporre esercizi troppo complessi a chi ha dimostrato di avere serie lacune su un determinato argomento.
L'iter nella batteria di recupero ricalca quello della prima batteria, sia nell'impostazione del quesito, sia nel calcolo dei punteggi, tramite continua analisi e aggiornamento dei cookie, sia nella proposta di pagine di feedback.
Arrivati al termine degli esercizi di recupero, viene proposta una schermata finale, nella quale si comunica la percentuale di successo nel quiz di recupero appena svolto, confrontandola con quella precedentemente ottenuta. Segue la possibilità di effettuare tre scelte:

1. iniziare una nuova batteria di esercizi i cui argomenti non sono specificati;
2. continuare gli esercizi di recupero;
3. effettuare il logout.

La prima scelta porterà alla pagina dei quiz, la seconda invierà alla pagina che cura gli esercizi di recupero, passandovi il numero dell'argomento e la nuova percentuale, mentre la terza porterà alla pagina di login.

5. Conclusioni

5.1 Conoscenze applicate e apprese

Durante lo svolgimento di questo progetto, ho avuto modo di entrare in contatto con gli elementi che stanno alla base della teorie di valutazione, analizzando le differenze che vi sono tra valutazione sommativa e formativa. Questo mi ha permesso di progettare l'eserciziario basandomi su logiche valutative differenti da quelle preponderanti nel mondo della scuola.
Dal punto di vista informatico, il progetto è stato estremamente stimolante, in quanto occasione di approfondimento circa le mie conoscenze di linguaggi quali PHP e jQuery, scoprendone nuove funzioni.

5.2 A chi si rivolge questo progetto

Il progetto su cui si è concentrata questa relazione è stato pensato per avere come target alunni e insegnanti della scuola secondaria, soprattutto di secondo grado. Ciò è dovuto al fatto che l'eserciziario, per la sua natura e per come è strutturato, presuppone una già sufficiente capacità di autoanalisi dei propri risultati, che verrebbe a mancare negli alunni più giovani.
Si è cercato di creare una struttura che si possa adattare a qualunque materia, sia scientifica che umanistica.

5.3 Sviluppi possibili e futuri

I possibili sviluppi, alcuni dei quali già in programma anche dopo la discussione della tesi, di questo progetto, che, allo stato attuale delle cose, si trova in forma tutto sommato ancora embrionale, potrebbero essere:

- inserire la possibilità di ripassare insieme più argomenti, proponendo, alla fine della prima batteria di esercizi, un elenco di checkbox;
- estendere i feedback dalla domanda alle singole opzioni;
- ottimizzare il range di punti con cui selezionare le domande di recupero.

6. Bibliografia e sitografia

Bibliografia

Paolo Azteni et al. *Basi di dati. Modelli e linguaggi di interrogazione*. McGraw-Hill, 2009.

Sitografia

W3Schools - http://www.w3schools.com/ ;

HTML.it - http://www.html.it/ ;

Wikipedia - https://en.wikipedia.org/ - https://it.wikipedia.org ;

Dev-oClock.com - http://www.dev-oclock.com/