

# Diario di lavoro

Luogo	SAMT
Data	24.10.2018

## Lavori svolti

Questa mattina ho completato la pagina di ricerca dei corsi, ho dovuto aggiungere l'intestazione alla selezione dei corsi in modo da spiegare la formattazione.

Materia:

Scegli materia...

Categorie

Materia | giorno | ora | crediti | numero massimo allievi

Insegna

Questa è la visualizzazione definitiva della pagina di ricerca dei corsi disponibile agli allievi.

La parte delle categorie verrà copiata anche nella pagina di creazione dei corsi di richiesta dei corsi e dei corsi pendenti, essendo che anche lì c'è una scelta della materia e vanno mostrate anche le categorie.

Ho finito la pagina inserendo le funzioni che si occupano di inserire l'utente e il corso nella tabella che spiega quella utente è registrato a quale corso.

```

131 public function setLearner(){
132     //Setto le variabili del database a cui faro la richiesta
133     require_once 'application/models/connection.php';
134     $connection = new Connection("efof.myd.infomaniak.com", "efof_gestrip18", "GestRip_Admin2018", "efof_gestrip_2018");
135
136     //Controllo che la sessione sia attiva e esista la variabile
137     if (!isset($_SESSION['mail'])) {
138         header("location: ". URL);
139     }else{
140         //Prendo tutti i valori dell'utente
141         $userData = $connection->getUser($_SESSION['mail'], 'nome_tipo');
142
143         //controllo se ha fatto il login un coach
144         if(!strcmp($userData, "coach")){
145             header("location: ". URL . "home/coach/");
146         }else if(!strcmp($userData, "amministratore")){
147             header("location: ". URL . "home/admin/");
148         }
149     }
150
151     //Prendo i dati in entrata dal form
152     if ($_SERVER["REQUEST_METHOD"] == "POST") {
153         $course = $_POST["course"];
154     }
155     //Ritorno i valori presi dal database
156     $connection->setLearner($course);
157     header("location: ". URL . "/user/searchCourses");
158 }

```

Questa è la funzione che viene richiamata al submit del form, che controlla l'utente e dopodiché prende il valore del corso a cui l'utente vuole iscriversi, poi lo invia alla funzione model che si occupa di inserire i dati nel database.

```

561 public function setLearner($course){
562
563     $active = 0;
564
565     //Connect to database
566     $conn = mysqli_connect($this->servername, $this->username, $this->password, $this->dbName);
567     // Check connection
568     if (!$conn) {
569         die("Connection failed: " . mysqli_connect_error());
570     }
571
572     //Inserisco l'utente nel corso che ha richiesto
573     $sql = $conn->prepare("INSERT INTO richiede(mail_utente, id_corso) VALUES(?, ?)");
574     $sql->bind_param("si", $_SESSION['mail'], $course);
575
576     if ($sql->execute()) {
577         return "Inserimento corso riuscito";
578     }else{
579         return $_SESSION['mail'] . " | " . $course . " | Errore nell'inserimento dei dati";
580     }
581 }

```

Questa è la funzione che si occupa di inserire i dati nel database, nella tabella richiede, che serve a contenere gli utenti e i corsi a cui essi sono iscritti.

Questa funzione una volta creata l'ho utilizzata anche nella funzione di richiesta dei corsi, in modo da non ripetere più volte lo stesso codice.

A proposito della funzione di richiesta dei corsi, la funzione nella classe "Connection" ha subito alcune modifiche. Queste modifiche sono state applicate per evitare che lo stesso utente si registri a più corsi se essi sono uguali e per non creare corsi se ci sono ancora posti disponibili in altri corsi uguali.

Ecco spiegata la nuova funzione.

```

302 public function requestCourse($subject, $day, $hour, $learners){
303     //Connect to database
304     $conn = mysqli_connect($this->servername, $this->username, $this->password, $this->dbName);
305     // Check connection
306     if (!$conn) {
307         die("Connection failed: " . mysqli_connect_error());
308     }
309
310     //Inizio a prendere gli id del corso
311     //Prendo l'id del corso in base ai dati precedentemente inseriti
312     $sql = $conn->prepare("SELECT id_corso FROM corso WHERE giorno = ? AND ora = ? AND num_allievi = ? AND nome_materia = ?");
313     $sql->bind_param("siss", $day, $hour, $learners, $subject);
314
315     $learnedCourses = array();
316     if ($sql->execute()) {
317         $result = $sql->get_result();
318         while ($row = $result->fetch_assoc()) {
319             array_push($learnedCourses, $row['id_corso']);
320         }
321     }

```

Il primo cambiamento è subito all'inizio, perché al posto di creare direttamente il nuovo corso si va a prendere tutti i corsi che hanno i requisiti richiesti dall'utente.

```

322     //Controllo se il numero di allievi richiesto per il corso non è stato superato.
323     //Variabile che contiene i partecipanti al corso
324     $courseSub = array();
325     if(count($learnedCourses) > 0){
326         for($i = 0; $i < count($learnedCourses); $i++){
327
328             //Prendo i valori dalla tabella "richiede"
329             $sql = $conn->prepare("SELECT * FROM richiede WHERE id_corso = ?");
330             $sql->bind_param("i", $learnedCourses[$i]);
331
332             if ($sql->execute()) {
333                 $result = $sql->get_result();
334                 $alreadyIn = false;
335                 //inserisco i valori degli utenti che richiedono il corso in un array
336                 while ($row = $result->fetch_assoc()) {
337                     array_push($courseSub, $row);
338
339                     //Controllo se l'utente è già in un corso uguale a quello richiesto
340                     if(!strcmp($row['mail_utente'], $_SESSION['mail'])){
341                         $alreadyIn = true;
342                         break;
343                     }
344                 }

```

Una volta presi tutti i corsi controlla che effettivamente ci siano corsi con quelle determinate caratteristiche, in caso ci fossero inizia prendendo tutti gli utenti già registrati a quel corso, che potrebbe essere 0 perché se il corso è stato creato da un coach è possibile che non ci siano ancora allievi registrati. Una volta presi gli utenti fa un controllo che l'utente registrato ad esso non sia lo stesso che vuole entrarci, in caso positivo allora porta la variabile di controllo "\$alreadyIn" a true e finisco di controllare, perché se l'utente è già registrato non deve rifarlo.

```

346 //Se l'utente è già in un corso uguale a quello richiesto non va avanti ma passa al prossimo corso.
347 if(!$alreadyIn){
348 //Se gli utenti iscritti sono meno degli utenti possibili inserisce l'utente
349 if(count($courseSub) >= $learners){
350 //Query per inserire i valori con il docente
351 $sql = $conn->prepare("INSERT INTO corso (giorno, ora, num_allievi, nome_materia) VALUES (?, ?, ?, ?)");
352 $sql->bind_param("siis", $day, $hour, $learners, $subject);
353
354 //Controllo che la connessione sia andata a buon fine e invio un messaggio
355 if ($sql->execute()) {
356 }
357
358 //Inserisco l'utente che si è iscritto al corso e il corso al quale si iscrive
359 return $this->setLearner($learnedCourses[$i]);
360 $courseSub = array();
361 }else{
362 break;
363 }
364 }else{
365 return "Errore nell'inserimento dei dati";
366 }
367 }
368 }
369 }
370 }

```

Controllo se la variabile di controllo è a false, quindi l'utente non è registrato in quel corso, e in quel caso faccio un altro controllo, ovvero che il corso non abbia raggiunto il numero massimo di iscritti possibili, in caso sia così crea il nuovo corso e inserisce l'utente in caso contrario inserisce solo l'utente nel corso. Questo controllo va ancora ricontrollato perché con i test bisogna assicurarsi che se il primo corso che controlla è pieno, e quindi crea un nuovo corso, c'è il rischio che il corso che controlla dopo non sia pieno e lo inserisca inutilmente.

```

371 }else{
372 //Query per inserire i valori con il docente
373 $sql = $conn->prepare("INSERT INTO corso (giorno, ora, num_allievi, nome_materia) VALUES (?, ?, ?, ?)");
374 $sql->bind_param("siis", $day, $hour, $learners, $subject);
375
376 //Essendo che il corso è nuovo sarà unico
377 //perciò lo ricerco e prend l'id per inserirci l'allievo nuovo
378 if ($sql->execute()) {
379
380 //Inizio a prendere gli id del corso
381 //Prendo l'id del corso in base ai dati precedentemente inseriti
382 $sql = $conn->prepare("SELECT id_corso FROM corso WHERE giorno = ? AND ora = ? AND num_allievi = ? AND nome_materia = ?");
383 $sql->bind_param("siis", $day, $hour, $learners, $subject);
384
385 //Essendo che il corso è nuovo sarà unico
386 //perciò lo ricerco e prend l'id per inserirci l'allievo nuovo
387 $learnedCourses = array();
388 if ($sql->execute()) {
389 $result = $sql->get_result();
390 while ($row = $result->fetch_assoc()) {
391
392 //Inserisco l'utente che si è iscritto al corso e il corso al quale si iscrive
393 $this->setLearner($row['id_corso']);
394 }
395 }
396 }
397 }
398
399 //INSERT INTO richiede (mail_utente, id_corso) VALUES (?, ?)
400 $conn->close();
401 }

```

Se nel controllo dei corsi esistenti, non trova nessun corso che esiste già con quei criteri allora ne crea uno e inserisce l'utente in quel corso, ricercando il suo id.

### Problemi riscontrati e soluzioni adottate

Non ho avuto grandi problemi se non i soliti legati a errori di scrittura del codice già citati negli scorsi diari, perché quello che ho fatto oggi era molto simile a quanto già fatto finora di conseguenza non c'erano errori nuovi.

### Punto della situazione rispetto alla pianificazione

Sono in line con il gantt, devo solo fare un ultimo controllo alla pagina di richiesta del corso, spiegato nella sezione "Lavori svolti" sotto la terza immagine della funzione "requestCourse".

Programma di massima per la prossima giornata di lavoro
Controllare la pagina di richieste del corso e mettere a posto eventuali problemi
Iniziare le pagine dell'admin.