Istituto di Istruzione Superiore "Benedetto Castelli"

Istituto Tecnico Industriale Statale

"PLAN IT!"
DIARIO ELETTRONICO

Candidata all'Esame di Stato

Martina Guarneri

Classe 5^ Informatica, sezione A

a.s. 2015/2016

Indice

Premessa

Negli ultimi anni la Scuola Italiana ha aperto le sue porte alla tecnologia, portandola tra i banchi attraverso diverse iniziative mirate a migliorare i metodi di apprendimento tradizionali.

Nel nostro Istituto, ad esempio, è in corso la sostituzione di tutte le lavagne a gesso con delle moderne Lavagne Interattive Multimediali, e da alcuni anni sono stati abbandonati i registri cartacei per adottare il Registro Elettronico; anche libri e quaderni stanno iniziando ad essere digitali, alleggerendo gli zaini degli studenti e riducendone il contenuto ad un tablet o un computer portatile. Per quanto riguarda il diario, invece, ancora adesso sono pochissime le alternative al cartaceo. Perché allora non creare un “diario elettronico”?

Come progetto per la mia Tesina ho quindi deciso di realizzare un’agenda digitale che permetta di annotare e gestire gli impegni scolastici come un vero diario, in modo da alleggerire ulteriormente lo zaino degli studenti e avere sempre i compiti a “portata di click”.

Introduzione

Ho deciso di realizzare un'applicazione web piuttosto che una mobile in quanto la prima richiede la conoscenza di molti degli argomenti affrontati in quinta; il nome che ho scelto di darle è "Plan it!", letteralmente "pianificalo", poiché richiama vagamente un'agenda.

"Plan it!" ricopre tutte le funzioni di un classico diario cartaceo: permette di gestire dal browser impegni scolastici (come lezioni, compiti e verifiche), organizzandoli per materia, grazie all'appoggio ad un database; per iniziare a utilizzarlo è necessario registrarsi con la propria e-mail e una password, con le quali verrà effettuato il login in futuro.

Oltre ad essere un'agenda permette anche di tenere una semplice rubrica e un blocco note, in cui segnare le informazioni relative alla scuola che non hanno una data, ma che devono poter sempre essere a portata di mano.

Linguaggi

Per la realizzazione del progetto ho utilizzato i seguenti linguaggi: HTML, CSS per l'aspetto estetico dell'applicazione web, PHP per l'elaborazione dei dati e SQL per la realizzazione e l'utilizzo del database.

Il linguaggio HTML, acronimo di "HyperText Markup Language", è un linguaggio di pubblico dominio utilizzato per la strutturazione di pagine nel World Wide Web. È stato creato alla fine degli anni '80 da Tim Berners-Lee, con lo scopo di condividere l'avanzamento delle ricerche scientifiche tra le migliaia di scienziati all'interno del CERN di Ginevra. HTML non può essere considerato un linguaggio di programmazione in quanto non prevede alcuna definizione di variabili, strutture dati, funzioni o strutture di controllo per la realizzazione di programmi.

Il Cascading Style Sheets invece, è anch'esso un linguaggio utilizzato per la formattazione di pagine web, e non è considerato un linguaggio di programmazione. Il CSS ha il vantaggio di permettere la separazione tra contenuti e formattazione nelle pagine web, rendendone la programmazione più chiara e il codice riutilizzabile.

Il linguaggio PHP, acronimo ricorsivo di "PHP: Hypertext Preprocessor", è un linguaggio di programmazione concepito per la realizzazione di pagine web dinamiche, la cui sintassi è vagamente simile a quella di C. Permette la programmazione ad oggetti ed è in grado di interfacciarsi con database come MySQL.

Infine il linguaggio SQL, acronimo di "Structured Query Language" è un linguaggio di interrogazione per database basato sul modello relazionale, utilizzato per creare, gestire e amministrare appunto database.

Registrazione, login e logout

Registrazione

Il primo passo per iniziare ad usare “Plan it!” è la registrazione: tramite un form l’utente dovrà inserire un indirizzo e-mail, il suo nome e una password; se i dati inseriti supereranno i controlli, l’utente verrà aggiunto al database, verrà aperta una sessione con il suo ID e il browser lo reindirizzerà alla vista principale dell’applicazione. In caso contrario verrà segnalato il motivo per cui non è potuta avvenire la registrazione.

Il primo controllo ad essere effettuato dalla pagina di registrazione (“registrazione.php”) è sulla avvenuta ricezione dei dati: se dal form (attraverso il metodo POST) sono arrivate le informazioni la pagina procederà all’elaborazione e agli ulteriori controlli; se invece non è stato ricevuto nulla, la pagina stamperà il form in quanto l’utente non ha ancora compilato i campi. Il form della registrazione si trova nella stessa pagina che elabora i dati, ed è composto da due campi di testo (per l’e-mail e il nome) e un campo password (per appunto la password); una volta che l’utente avrà confermato il form, i dati verranno inviati alla stessa pagina tramite il metodo POST del protocollo HTTP.

Dopo aver confermato l’avvenuta ricezione dei dati potrà essere effettuato il controllo sull’e-mail: il database dovrà essere interrogato per verificare che l’e-mail inserita nel form non sia già in uso da un altro utente. Se il controllo non dovesse essere superato, la pagina non procederà alle successive verifiche e stamperà nuovamente il form con un messaggio d’errore per segnalare all’utente il problema. Nel caso in cui l’interrogazione non abbia prodotto riscontri la password inserita verrà crittografata con l’algoritmo “blowfish” tramite uno script (“PassHashTools.php”) e le informazioni dell’utente verranno salvate nella tabella “utenti” del database. Come ultima cosa prima di reindirizzare alla vista principale dell’applicazione verrà aperta una sessione con l’ID assegnato all’utente dal database, che verrà utilizzato in seguito per associare i dati all’utente.

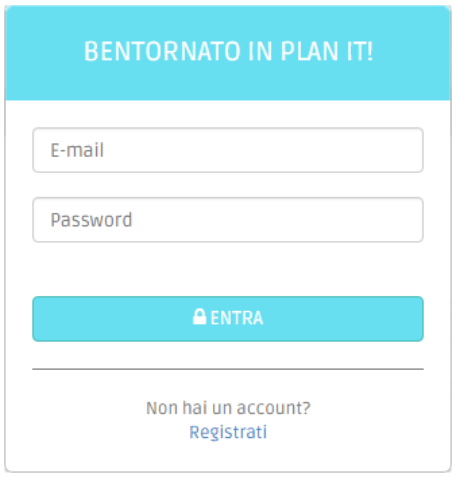
Login

Il login funziona in modo simile alla registrazione: sempre tramite un form l’utente dovrà inserire l’indirizzo e-mail e la password scelti al momento della registrazione; se i dati inseriti supereranno i controlli verrà aperta una sessione con il suo ID e il browser lo reindirizzerà alla vista principale dell’applicazione. In caso contrario verrà segnalato il motivo per cui non è potuto avvenire il login.

Anche questa volta è la stessa pagina a occuparsi di tutto il processo (“login.php”): se dal form (attraverso il metodo POST) sono arrivati e-mail e password, la pagina procederà ai controlli; se invece non è stato ricevuto nulla, la pagina stamperà il form in quanto l’utente non ha ancora compilato i campi.

Se i dati sono stati ricevuti, il database dovrà essere interrogato per verificare la presenza dell’e-mail nel database. Se il controllo non dovesse essere superato, la pagina non procederà alle successive verifiche e stamperà nuovamente il form con un messaggio d’errore per segnalare all’utente il problema. Nel caso in cui l’interrogazione abbia prodotto un riscontro, la password inserita verrà crittografata con l’algoritmo “blowfish” tramite lo

stesso script utilizzato nella fase di registrazione (“PassHashTools.php”) e verrà confrontata con la password associata all’e-mail nel database (che è l’hash della password in chiaro). Superato questo controllo verrà aperta una sessione con l’ID dell’utente (anch’esso prelevato dal database) e l’utente verrà reindirizzato alla vista principale dell’applicazione. Se invece la password non dovesse corrispondere, la pagina stamperà nuovamente il form con un messaggio d’errore per segnalare all’utente il problema.



Finestra di login

Logout

Per quanto riguarda il logout, invece, si tratta di uno script PHP (“logout.php”) chiamato in seguito alla pressione di un bottone, sempre presente in alto a destra in ogni pagina dell’applicazione a seguito al login; questo script semplicemente rimuove dalla sessione l’ID dell’utente settato nella fase di login, distrugge la sessione e reindirizza l’utente alla pagina di login.

```php
<?php
  //Apre la sessione
  session_start();

  //Cancella l'ID
  session_unset($_SESSION["ID"]);

  //Distrugge la sessione
  session_destroy();

  //Reindirizza alla pagina di login
  header("location: login.php");
?>
```

Script di logout (“logout.php”)

Sessione

In informatica la sessione è l'attività svolta tra due entità per trasferire dati in entrambi i sensi per tutta la durata del collegamento, e comprende tre fasi: avvio (detto anche login), lavoro in sessione (scambio delle informazioni) e chiusura (viene cancellata la sessione). All'interno della mia applicazione ho usato le funzioni offerte da PHP per gestire le sessioni per poter salvare l'ID dell'utente per tutta la durata del suo "soggiorno" sul sito. Grazie all'ID utente, l'applicazione è in grado di estrarre dal database solamente le informazioni associate all'utente loggato (nascondendo quelle degli altri utenti) e associargli le nuove informazioni al momento di inserimento nel database.

Ogni pagina dell'applicazione (eccetto il login, la registrazione e gli script) prima di stampare qualsiasi contenuto apre la sessione e controlla che l'ID dell'utente sia settato: nel caso in cui non lo sia nega l'accesso al contenuto della pagina e reindirizza alla pagina di login, in quanto nessun utente è loggato in quel momento. Se invece l'ID dovesse essere settato, la pagina procederà all'output dei contenuti, e prenderà dal database le informazioni associate a quell'ID.

```
36    <!--SESSIONE  E NOME UTENTE-->
37    <?php
38      //SESSIONE
39      //Apre la sessione
40      session_start();
41      //Controlla che l'id sia settato
42      if(!isset($_SESSION["ID"])) //se nessun utente è loggato
43      {
44        //reindirizza alla vista principale
45        header("location: login.php");
46      }
```

Porzione di codice che si occupa di verificare che l'ID sia settato nella sessione

L'ID dell'utente rimane invariato per tutta la durata della sessione, e verrà rimosso solo alla chiamata dello script di logout ("logout.php"), di cui ho spiegato il funzionamento nel capitolo precedente.

Database

Per salvare i dati degli utenti ho creato un database MySQL (chiamato "planner"): al suo interno le informazioni sono divise in tabelle per categorie (utenti, contatti, note, lezioni,

materie ed eventi) per semplificarne la gestione, con lo svantaggio di raggruppare insieme i dati di diversi utenti. Ciò però non comporta la perdita dell'integrità dei dati di uno specifico utente, poiché ogni tabella è referenziata direttamente o indirettamente alla tabella degli utenti tramite una chiave esterna ID_utente, grazie alla quale è possibile risalire ai dati di un utente.

Una volta terminata la fase di registrazione viene aggiunto un record alla tabella Utenti contenente le informazioni inserite dall'utente nella registrazione, ovvero l'email, il nome e l'hash della password; a questi dati viene aggiunto automaticamente dal database un ID univoco, che permetterà di associare le informazioni a quello specifico utente.

Diagramma ER

Ogni utente ha quindi un ID (che costituisce la chiave primaria), un nome, un'e-mail e una password (viene conservato l'hash della password); un utente può possedere una o più materie, una o più note e uno o più contatti.

Ogni contatto ha un ID (chiave primaria), un ruolo e un'e-mail; un contatto deve essere posseduto da un solo utente.

Ogni nota ha un ID (chiave primaria), un titolo e un testo; una nota deve essere posseduta da un solo utente.

Ogni materia ha un ID (chiave primaria, che permetterà di risalire alle lezioni e gli eventi di un utente) e un nome; una materia può possedere una o più lezioni ed eventi, e deve essere posseduta da un solo utente.

Ogni lezione ha un ID (chiave primaria), un titolo, un luogo, una data, un'ora di inizio e un'ora di fine; una lezione deve essere posseduta da una sola materia.

Ogni evento ha un ID (chiave primaria), un titolo, una categoria, una descrizione, una data, un'ora di inizio e un'ora di fine; una lezione deve essere posseduta da una sola materia.

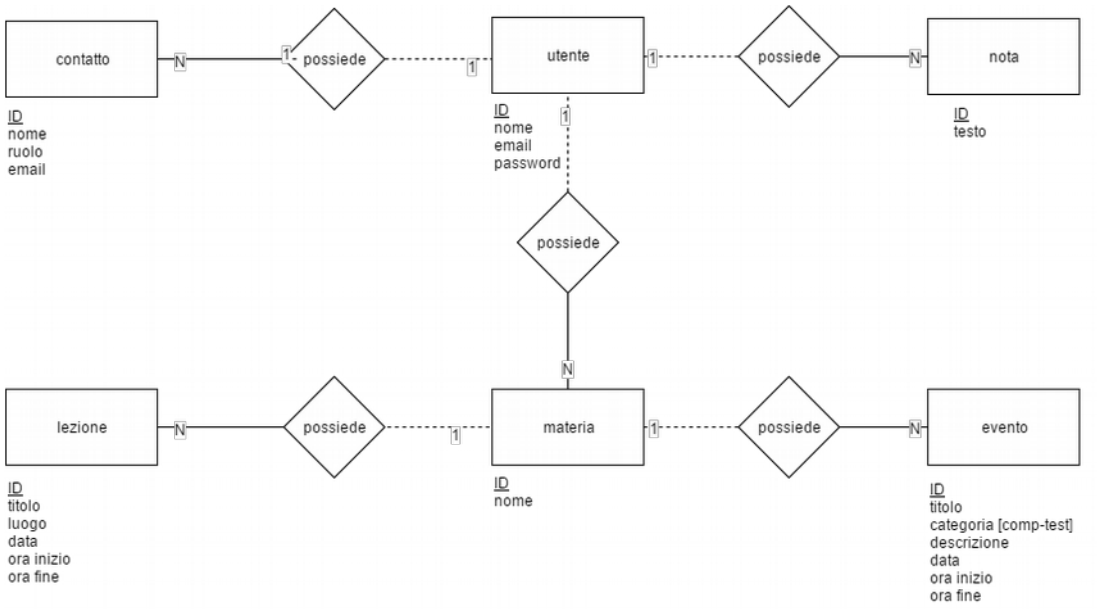


Diagramma ER del database

Per estrapolare e manipolare le informazioni dall'applicazione web ho utilizzato la libreria MySQLi ("MySQL Improved") messa a disposizione da PHP dedicata all'interazione con il DBMS MySQL.

Struttura generica

Ogni pagina dell'applicazione ha lo stesso scheletro, composto da una barra fissa in alto, una barra di navigazione laterale a scomparsa (sidebar) e infine il contenuto vero e proprio, che cambia di pagina in pagina.

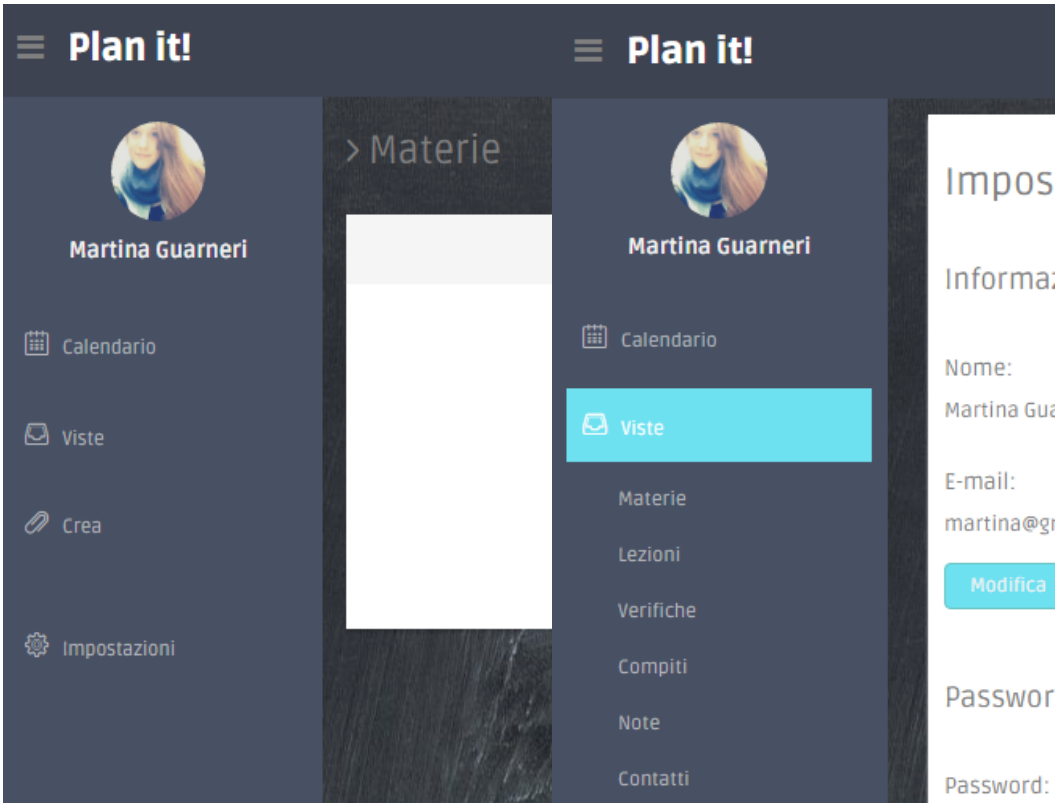
Top bar

La barra in alto, di color grigio scuro, è caratterizzata da tre elementi: un bottone che controlla la sidebar, la scritta "Plan it!", e infine il tasto "Esci". Il primo bottone (la cui icona sono tre righe incolonnate) permette di mostrare o nascondere la sidebar sottostante, per rendere più spaziosa la visualizzazione del contenuto della pagina (lo screenshot della sidebar è visibile nel prossimo paragrafo); passando il cursore sul bottone comparirà la scritta "Mostra/nascondi", per chiarirne l'utilizzo. Accanto al bottone è visibile in bianco il nome dell'applicazione: cliccandoci sopra si verrà reindirizzati alla vista principale del sito. Infine all'estrema destra della pagina (mi è risultato difficile mostrarlo in uno screenshot, in quanto se avessi mantenuto le proporzioni originali sarebbe stato illeggibile) compare il tasto "Esci": non è altro il tasto che esegue lo script di logout, e una volta cliccato l'utente si vedrà reindirizzato alla pagina di login.

Screenshot della top bar (zommato e tagliato)

Sidebar

La sidebar, invece, è una barra di navigazione lunga quanto tutta la lunghezza del contenuto che permette all'utente di spostarsi attraverso le svariate pagine dell'applicazione; all'inizio compaiono il nome specificato dall'utente al momento di registrazione e una foto, seguiti da un piccolo menù organizzato per funzioni: il calendario, le viste, i form per la creazione di nuovi elementi (lezioni, compiti, eccetera) e le impostazioni dell'account. Quando l'utente si trova in una pagina, il contorno di quella specifica pagina sarà colorato di azzurro nella barra di navigazione, per rendere chiaro in quale parte del sito sta navigando. Per spostarsi a un'altra scheda gli sarà sufficiente cliccare la voce corrispondere nella sidebar; nel caso di "viste" e "crea" si aprirà un sottomenù a scomparsa che permetterà di scegliere la categoria.



Sidebar, lunga quanto la lunghezza della pagina

Calendario

Nella vista principale dell'applicazione è presente un calendario, per permettere all'utente di controllare facilmente gli impegni di un determinato giorno. Per la realizzazione del calendario ho utilizzato uno script di Star Tutorial, che ho modificato e adattato alle mie esigenze. Lo script consiste in una classe PHP (contenuta nella pagina "calendar.php"), da cui si ottiene un calendario in HTML creando un'istanza della classe e chiamando la funzione "show()". Nel calendario è possibile scorrere i mesi con i bottoni "Precedente" e "Successivo"; cliccando sui giorni, invece, verrà aperta la vista relativa al giorno cliccato che mostrerà le lezioni, i compiti e le verifiche relative a quel giorno. Ciò è possibile poiché ogni casella è un collegamento ipertestuale che ha come attributo nell'URL la data cliccata. La vista del giorno quindi otterrà attraverso il metodo GET la data scelta, e la userà come condizione nell'interrogazione del database, per ottenere solo gli impegni relativi a quel giorno.

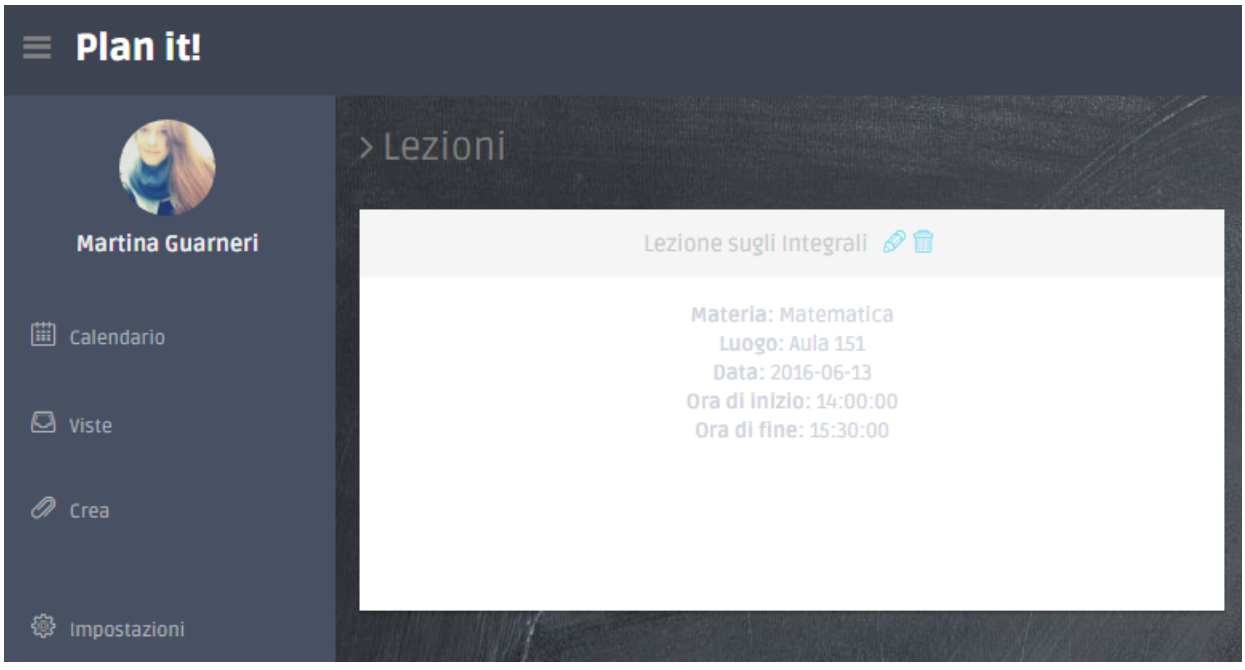
La classe contenuta in "calendar.php" contiene al suo interno sei funzioni, che se usate insieme permettono la realizzazione del calendario; ogni funzione si occupa di un determinato aspetto, come ad esempio le settimane presenti nel mese da mostrare, il numero di giorni, e la disposizione in base al giorno della settimana. Per far lavorare insieme queste funzioni è necessario chiamarne l'unica pubblica, nominata "show()": una volta chiamata, essa definisce la struttura del calendario richiamando al suo interno le altre funzioni private, e restituisce il calendario completo. La realizzazione avviene nel seguente modo: prima di tutto viene determinato quante settimane (e quindi righe) ha il mese da visualizzare, e per ogni settimana vengono create sette celle e stampate tutte su una stessa riga. Stampandole viene aggiunto il numero in base al corrispondente giorno della settimana.

Calendario della vista principale

Viste

All'interno dell'applicazione la visualizzazione dei dati (eventi, compiti, eccetera) dell'utente è affidata alle "viste": ogni categoria ha una vista dedicata in cui vengono mostrate solo le informazioni dell'utente relative a quella categoria. Oltre a queste viste dedicate, ne ho realizzate altre due che racchiudono all'interno più categorie: la prima è la vista "per giorno", a cui si può accedere selezionando un giorno dal calendario nella vista principale; la seconda invece è la vista per materia.

Tutte le viste hanno la stessa struttura grafica: in alto a sinistra compare il titolo della categoria della vista, e sotto di esso vengono disegnati dei pannelli bianchi in stile "post-it" contenenti i dati relativi alla categoria, disposti uno di fianco all'altro. Ogni pannello contiene le informazioni di un solo elemento, ovvero di un solo record (riga) della tabella nel database corrispondente alla categoria; se nella tabella non dovessero esserci righe associate all'utente loggato, ciò verrà segnalato all'utente. Nel pannello inoltre sono disponibili due icone, una matita e un cestino: la matita permette di modificare le informazioni del pannello, e il cestino di eliminarle. Entrambi sono collegamenti ipertestuali, il primo (modifica) alla pagina di modifica delle informazioni relativa alla categoria, nel cui URL è aggiunto l'ID delle informazioni da modificare (tramite il metodo GET), e il secondo allo script di eliminazione ("elimina.php") nel cui URL sono specificate la categoria e l'ID delle informazioni da eliminare. Nel prossimo capitolo spiegherò entrambe le pagine nel dettaglio.



Porzione di screenshot della vista delle lezioni, con titolo e "post-it"

Ogni vista (dopo aver effettuato i controlli sulla sessione e la stampa dei contenuti generici della pagina) apre la connessione con il database e lo interroga richiedendo le informazioni relative alla categoria della vista, specificando nella query la condizione

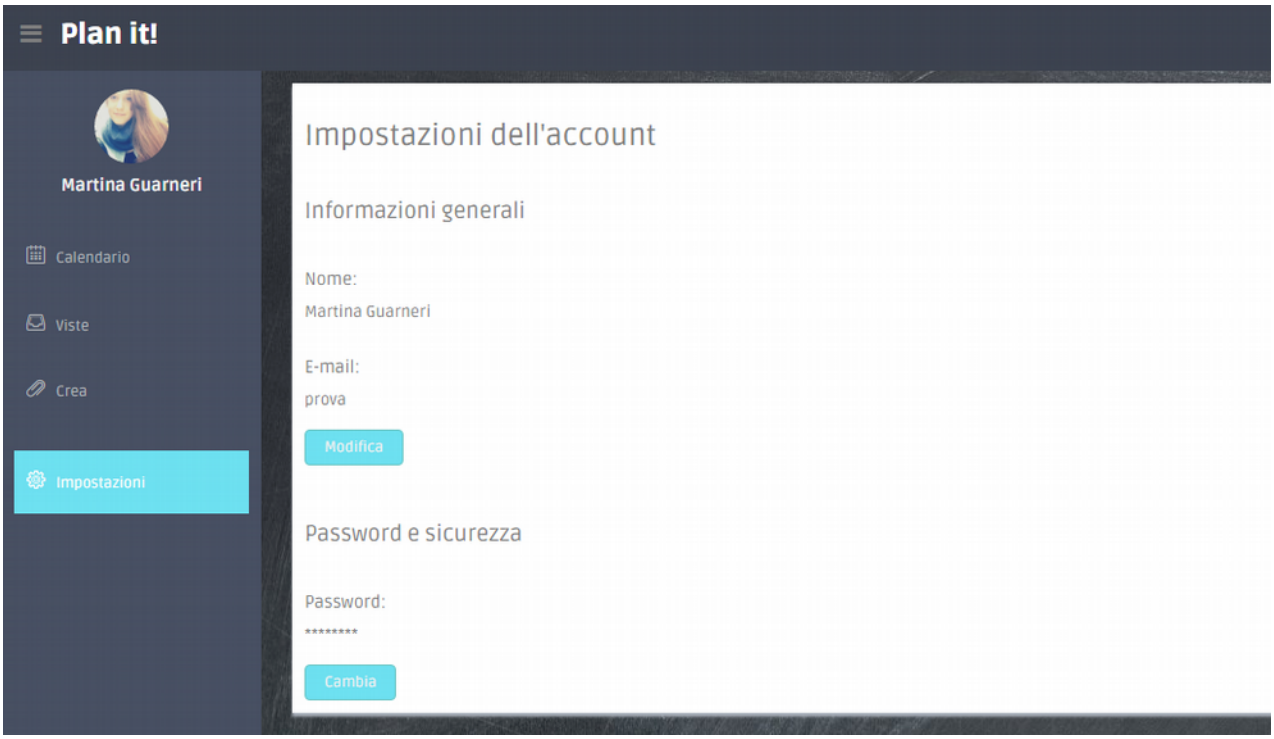
sull'ID dell'utente, ottenuto grazie alla sessione. La tabella prodotta come risultato viene analizzata record per record all'interno di un ciclo while: per ogni riga, che corrisponde a un elemento di una categoria (quindi per ad esempio ad una lezione o ad un compito) viene stampato ogni componente del pannello. Nell'intestazione del pannello viene mostrato il titolo (o il nome) dell'elemento, con accanto le icone della matita e del cestino contenute in due tag con un collegamento ipertestuale; il resto delle informazioni invece viene stampato all'interno del pannello vero e proprio, specificando accanto di che valore si tratta. Terminato di stampare le informazioni di un elemento della categoria, il ciclo procede all'elemento successivo ripetendo gli stessi passaggi fino al termine della tabella.

Viste per giorno e per materia

Nelle viste per categoria, la fase precedentemente descritta viene ripetuta una sola volta, poiché le informazioni che devono essere mostrate sono relative solamente a quella specifica categoria. Nella vista giorno e nella vista per materia, invece, vengono mostrate tutte le informazioni di tutte le categoria (ovvero eventi, compiti e verifiche) associate a uno specifico giorno o a una specifica materia; i passaggi da eseguire per mostrare le informazioni di una categoria dovranno essere ripetuti per ciascuna delle categorie da mostrare.

Vista delle impostazioni

Un'altra vista particolare che non ho menzionato prima è la vista delle impostazioni ("impostazioni.php"): questa pagina permette all'utente di visualizzare e modificare le informazioni relative al suo account, ovvero e-mail e nome e password. Questa vista sfrutta l'ID contenuto nella sessione per accedere alle informazioni relative all'utente contenute nella tabella "utenti" del database, mostrandole all'interno di un panello con accanto specificato di che informazione si tratta. Oltre a poterle visualizzare la pagina permette anche di modificarle: cliccando su uno dei due bottoni, l'utente verrà reindirizzando alla pagina per modificare le-mail e o il nome ("modifica_informazioni.php") nel caso di click sul primo bottone, e a quella per modificare la password ("modifica_password.php") nel caso abbia cliccato sul secondo bottone.



Porzione di screenshot della vista delle impostazioni

Form

All'interno dell'applicazione web ho fatto ricorso all'uso di form per l'inserimento e la modifica dei dati, ovvero per tutte le pagine appartenenti alla categoria "crea" e "modifica". Ognuna ha lo stesso meccanismo di funzionamento della pagina di registrazione e login: tramite un form l'utente inserisce i dati, e se i dati inseriti supereranno i controlli verranno aggiunti al database (o aggiornati dei dati già esistenti, nel caso si tratti della pagina di modifica) e verrà comunicato all'utente il successo dell'operazione; in caso contrario verrà segnalato il motivo per cui non è potuta avvenire la creazione (o la modifica) nel database.

Anche questa volta è la stessa pagina a occuparsi di tutto il processo ("crea_nomedellacategoria.php" o "modifica_nomedellacategoria.php"): se dal form (attraverso il metodo POST) sono arrivate le informazioni, la pagina procederà ai controlli; se invece non è stato ricevuto nulla, la pagina stamperà il form in quanto l'utente non ha ancora compilato i campi.

Per ogni categoria esiste una pagina di creazione e una pagina di modifica, con gli stessi controlli e lo stesso funzionamento, eccetto per due dettagli: nelle pagine di modifica, a differenza delle pagine di creazione, i campi del form vengono compilati automaticamente con le informazioni dell'elemento da modificare; la seconda cosa è la query inviata al database dopo aver superato il controllo sui dati, infatti le pagine "crea" creeranno delle nuove istanze, mentre le pagine "modifica" aggiorneranno le informazioni dell'elemento modificato.

Nella pagine di modifica la compilazione automatica dei campi dei form avviene grazie all'ID dell'elemento ottenuto tramite il metodo GET: questo ID viene incluso nel link dei tag delle matite delle viste, e una volta giunti sulla pagina di modifica viene estratto e utilizzato per recuperare le informazioni da inserire nel form. L'ID inoltre verrà utilizzato per aggiornare le informazioni nel database.

Conclusioni

Realizzando questa applicazione mi sono resa conto di quante funzioni mi sarebbe piaciuto aggiungere alla mia applicazione; più ne implementavo e più me ne venivano in mente altre. Non sono riuscita purtroppo ad aggiungere tutto quello che avrei voluto, sia per via del tempo, che per la difficoltà nel realizzare il progetto curandolo nei dettagli.

Per il momento ho cercato di dare a "Plan it!" tutte le funzioni di un classico diario cartaceo, e cioè…..

Per gli sviluppi futuri si potrebbero aggiungere le seguenti funzionalità: ecc.

Bibliografia e sitografia

Libri

"Beginning PHP & MySQL Development", Pawprints Learning Technologies

Siti web

Html.it (www.html.it)

Repository di alecos71 (https://github.com/alecos71) [calendario]

Star Tutorial (www.startutorial.com) [calendario]

Template "DASHGUM" di Carlos Alvarez (alvarez.is) [template dell'applicazione]

W3Schools (www.w3schools.com)

Wikipedia (it.wikipedia.org)

Ringraziamenti

Vorrei ringraziare tutti gli insegnanti che mi hanno accompagnato in questi cinque anni e che hanno contribuito alla mia formazione come Perito e come persona; in particolare ringrazio i docenti di quinta, tra cui la professoressa Laura Pasolini, senza la quale non avrei superato il test di ammissione all'Università.

Ci tengo inoltre a fare un ringraziamento speciale al professor Alessandro Bugatti: oltre ad essere stato un valido insegnante di informatica (come pochi), è riuscito a darmi la motivazione di cui avevo bisogno per continuare la mia scelta scolastica, incoraggiandomi a non arrendermi nonostante mi sentissi inferiore ai miei compagni. Sono felice di aver avuto l'opportunità di essere sua alunna, e anche di questo devo ringraziarlo: se cinque anni fa all'Open day dell'ITIS non avesse tranquillizzato i miei genitori, ora probabilmente sarei in tutt'altra scuola.