RELAZIONE PROGETTO TECNOLOGIE WEB 2022/2023

di Emanuele Di Maggio n° matricola 883368

Il sito web simula un tracciatore di libri letti (tracker) permettendo di accedere o registrarsi con le proprie credenziali, aggiungere i libri letti al proprio profilo, visualizzarne l'aggiunta e dare un voto.

Sezioni

Le pagine Html sono costruite in questo modo:

- Header top.html;
- Pagina php contenente il corpo principale *.php e il banner banner.html oppure logged.html;
- Footer bottom.html.

Ogni pagina *.php ha il proprio file *.js che si occupa di gestire gli elementi.

Impostazione degli stili:

- top.html e bottom.html sono gestiti dal file top-bottom.css;
- index.php e search.php sono gestiti dal file index-search.css;
- profile.php è gestito da profile.css;
- book.php è gestito da bookinfo.css

I ruoli permessi sono quelli di utente base o amministratore:

- L'utente base può aggiungere e visualizzare i libri aggiunti sul proprio profilo;
- L'amministratore, oltre alle funzioni di utente base, può aggiungere libri e autori al database.

HOME PAGE, LOGIN/SIGNUP E DATABASE

index.php

La Home Page mostra il logo e la descrizione del sito.

Per utilizzare il sito bisogna fare login o signup attraverso i rispettivi pulsanti nel banner

Nel caso si provi ad accendere alla pagina di ricerca senza essere loggati, si verrà riportati alla home page con un messaggio di errore.

login.php

La pagina di Login mostra una form con due label, uno per l'username e l'altro per la password, e un bottone per entrare.

Al file php sono legati:

- Il file login.js che legge i dati inseriti nei label e quando viene premuto il pulsante invia una richiesta ajax (POST) allo script login-signup.script.php.
- Il file login-signup-script.php si occupa di formulare i dati ricevuti dal file javascript e comunicare con il database.

Login di un account esistente:

Ottenuti i valori di username e password nel file login.js, questi vengono salvati in variabili e ne viene verificata la validità:

- username e password non devono contenere caratteri speciali (*);
- username e password non devono essere vuoti o contenere whitespaces.

Verificato questo, i valori sono inviati tramite ajax (POST) allo script login-signup-script.php che tramite query verificherà che ci sia corrispondenza tra username e password nel database. In caso negativo verrà segnalato errore, sia che l'username non esista o sia che la password non sia corretta per l'username.

signup.php

La pagina di Signup mostra un form con diversi label dove i valori inseriti saranno letti, dopo aver cliccato il bottone, dal file signup.js.

Al file php sono legati:

- Il file signup.js che si occupa di leggere i dati inseriti nella form e tramite richiesta ajax (POST) li invia allo script login-signup-script.php.
- Il file login-signup-script.php si occupa di formulare i dati ricevuti tramite ajax, salvarli ognuno in una variabile e comunicare con il database per permettere l'aggiunta del nuovo utente.

Creazione nuovo account:

Ottenuti i valori di firstname, lastname, dateofbirth, username, password, confirmpassword e del radio button, che indica se l'account è di tipo amministratore o meno, nel file signup.js, questi vengono salvati in variabili e ne viene verificata la validità:

- firstname, lastname, username e password non devono contenere simboli non validi (*);
- firstname, lastname, username, dateofbirth, password e confirm password non devono essere vuoti o contenere whitespaces.

Dopo questo i valori sono inviati tramite ajax allo script login-signup-script.php dove saranno inseriti tramite query nel database.

db.php

Per connettersi al database, ogni script php include il file db.php.

Il database è così costruito:

- Tabella author -> id🔑, fullname, birth
- Tabella books -> book_id🔑, author_id🔑, name, year, main_character, poster, synopsis, rating
- Tabella books_saved -> user🔑, id_book🔑, title, date, rating
- Tabella users -> nickname🔑, password, date_of_creation, administrator
- Tabella user_info -> info_nickname🔑, firstname, lastname, birth

SEARCH E BOOK INFO

Nel sito è presente la possibilità di ricercare i libri che si trovano nel database e visualizzarne le informazioni.

search.php

La pagina mostra un form con un label dove si inseriscono i valori per la ricerca e un bottone per inviare il risultato allo script tramite jquery.

Al file php sono legati:

- Il file search.js che si occupa di leggere il testo inserito nel label e tramite ajax (GET) inviare il valore allo script searchpage.php.
- Il file searchpage.php che elabora i dati ottenuti attraverso ajax, cerca una corrispondenza di questi nel database e ritorna i risultati al chiamante.

<u>Come funziona la ricerca di un libro:</u>

Una volta inserita una stringa di testo nel label di ricerca e premuto il bottone, sia attiva la funzione searchBooks() in search.js che legge la stringa, la salva in una variabile e la invia tramite ajax (GET) a searchpage.php, se la stringa ricevuta esiste la salva nella variabile $book alla quale aggiunge il simbolo di “%” che serve per la ricerca nel database. La ricerca viene fatta basandosi sul titolo del libro e ogni corrispondenza trovata verrà salvata in $result; quest’ultimo verrà poi ritornato in formato JSON a search.js.

Una volta ottenuto il risultato dallo script la funzione searchBooks() crea una serie di dinamicamente che inserirà nella pagina search.php; ogni contiene il poster, il titolo, l’autore, l’anno e il protagonista (se esiste). Se il protagonista appare in più libri in fase di creazione dei , il suo nome verrà affiancato da un’icona che lo rappresenta con l’ausilio della funzione checkMain() (**).

Ogni creato ha nel poster implementata la funzione onclick che se attivata, chiama la funzione saveId() che tramite richiesta ajax (GET), invia l'ID del libro cliccato allo script write.php che si occuperà di salvarlo nel file di testo bookid.txt; fatto ciò ajax porterà alla pagina book.php.

book.php

La pagina mostra le informazioni base di un libro, quali: poster, titolo, autore, anno di pubblicazione, protagonista, e sinossi. Oltre a queste vengono rappresentate anche informazioni che riguardano il singolo utente quali: il voto dato e la somma dei voti totali ricevuti.

Al file php sono legati:

- Il file book.js che gestisce tutte le informazioni relative alla pagina e comunica tramite ajax con diversi script php;
- Il file book_searched.php ottiene dal database tutte le informazioni del libro che si sta visualizzando;
- Il file bookid.txt che contiene l'ID del libro che si sta visualizzando;
- Il file read.php che legge l'ID contenuto in bookid.txt;
- Il file profileinfo.php utilizzato per aggiungere il libro che si sta visualizzando ai libri "letti" dall'utente.

Come funziona la visualizzazione di un libro e cosa è possibile fare:

Come detto in precedenza alla pagina vi si accede tramite ajax contenuto in search.js.

Aperta la pagina, per prima cosa book.js richiama la funzione readFIle() che tramite ajax (GET) legge il valore contenuto in bookid.txt e richiama a sua volta la funzione showInfoBook(); quest'ultima ottenuto l'ID del libro lo passa sempre tramite ajax (GET) allo script book_searched.php, dove viene salvato in una variabile e tramite query cercato nel database, una volta trovate tutte le sue informazioni, queste sono ritornate in JSON a book.js che si occuperà di mostrarle in pagina. Aggiunte tutte le informazioni ai rispettivi elementi in html, la funzione showInfoBook() richiama altre due funzioni:

- sumAllRatings() che richiama tramite ajax (POST), passandogli l'ID del libro, book_searched.php per calcolare il voto totale in percentuale sommando tutti i voti di tutti gli utenti (se è stato votato almeno una volta). Viene mostrato accanto al voto numerico un'icona che cambia in base al voto (<25, >25 e <=50, >50 e <=75, >75) tramite la funzione addIconToRating();
- bookInList() che richiama tramite ajax (POST), passandogli l'ID del libro, book_searched.php per verificare che il libro sia stato aggiunto ai libri "letti" dall'utente, in caso affermativo viene mostrato il menù dropdown per votare il libro.

Votare un libro tramite menù dropdown attiva la funzione addYourVote() che seleziona la voce scelta dal menu e l'ID del libro e li passa tramite ajax (POST) a book_searched.php, qui verrà aggiunto tramite query il voto al database (Siccome tra le voci del menu dropdown esiste "to vote", in caso l'utente lo scelga viene ritornato un errore dallo script).

Una volta che l'utente ha votato il libro ogni volta che verrà ricaricata la pagina la funzione bookInList() mostrerà la voce del voto nel menù dropdown.

Una volta cliccato il bottone per aggiungere un libro alla tabella nel database che indica il libri "letti", viene chiamata la funzione addToLibrary() in book.js; questa salva l'ID del libro, il titolo del libro e il valore e formato della data corrente, questi tre valori sono passati tramite ajax (POST) a profileinfo.php dove per prima cosa viene verificato che il libro non sia già stato aggiunto, in tal caso viene ritornato errore, al contrario lo aggiunge al database.

PROFILE

Ogni account che accede ha la propria pagina profilo dove sono contenute tutte le informazioni personali, i libri aggiunti e nel caso di account amministratori la possibilità di aggiungere nuovi libri e autori nel database.

profile.php

La pagina mostra le informazioni dell'utente quali username, nome e cognome, la data di nascita e quella di creazione dell'account.

Al file php sono legati:

- il file profile.js che gestisce tutte le informazioni riguardo al profilo comunicando con diversi script php;
- il file profileinfo.php che ritorna le informazioni dell'utente del profilo;
- il file add-book.php che permette l'aggiunta di libri o autori al database.

Come mostrare i libri aggiunti come "letti":

Caricata la pagina viene richiamata la funzione allBookAdded() in profile.js che tramite ajax (POST) richiama lo script profileinfo.php dove ottenuto il nome dell'utente tramite sessione attiva vengono cercati tutti i libri aggiunti nel database da esso. L'array con tutti i libri viene ritornato in formato JSON e viene creata dinamicamente una tabella con ogni libro trovato.

Come eliminare un libro "letto" dalla tabella:

Ogni elemento della tabella può essere rimosso da essa e dal database, venendo considerato di conseguenza come "non letto", tramite drag & drop dello stesso.

Ognuno degli elementi della tabella ha la caratteristica draggable(), mentre sotto alla tabella vi si trova un contenente un immagine con caratteristica droppable().

Tutto questo è possibile tramite l'implementazione di https://ajax.googleapis.com/ajax/libs/jqueryui/1.11.4/jquery-ui.min.js che permette di creare elementi grabbable e droppable.

Rimosso visivamente dalla tabella l'elemento, viene richiamata la funzione deleteFromList() che tramite ajax (POST) richiama add-book.php passandogli l'ID del libro selezionato. Nello script viene rimosso il libro dal database tramite query.

Account amministratore -> tabella autori:

Per permettere di visionare gli autori presenti nel database viene richiamata la funzione uploadAuthorTable() che tramite ajax (POST) richiede a add-book.php un array contenente tutti gli autori.

Ogni nome e ID degli autori verrà aggiunto come nuova riga di una tabella creata dinamicamente.

Account amministratore -> aggiunta di un autore nel database:

In caso di utente amministratore vi è la possibilità di aggiungere un nuovo autore al database, questo è possibile tramite il fieldset NEW AUTHOR.

Inserito il nuovo nome dell'autore, la sua data di nascita e cliccato il bottone viene per prima cosa salvato in profile.php il nome inserito e verificato che sia valido (niente simboli o numeri o vuoto), successivamente viene passato tramite ajax (POST) allo script add-book.php, dove tramite query verrà verificato che l'autore non esista già sulla base del nome, in caso negativo verrà aggiunto al database e sarà auto incrementato il valore dell'ID.

Se l'autore è stato aggiunto la funzione ajax chiamata aggiungerà una nuova riga alla tabella degli autori.

Account amministratore -> aggiunta di un libro nel database:

In caso di utente amministratore vi è la possibilità di aggiungere un nuovo autore al database, questo è possibile tramite il fieldset NEW BOOK.

Aggiungere un libro richiede diversi attributi: il titolo, l'ID dell'autore, la data di pubblicazione, il protagonista (se esiste), la sinossi e il poster. Tutti questi valori sono letti dai rispettivi label quando viene cliccato il bottone di aggiunta richiamando la funzione, in profile.js, addNewBook(). Quest'ultima crea un FormData e vi aggiunge ogni valore letto, ma non prima di avere verificato che:

- il titolo, l'ID dell'autore, l'anno, il poster e la sinossi non siamo vuoti;
- l'anno e l'ID dell'autore siano composti solo da numeri.

Se viene selezionato il checkbox, viene aggiunto a fianco dell'anno la stringa B.C. che indica se il libro è stato scritto prima o dopo Gesù.

Il FormData viene passato tramite ajax (POST) e inviato allo script add-book.php, ogni elemento viene letto e salvato in una variabile. I passi sono:

- Viene verificato che l'ID dell'autore esista nel database o non sarebbe possibile aggiungere un nuovo libro. In caso negativo viene ritornato un errore;
- Se l'autore esiste, allora è possibile aggiungere il libro nel database tramite query;
- Aggiunto il libro, viene ritornata una stringa ad ajax che indica l'andata a buon fine dell'aggiunta.

SESSION

Tutte le volte che si accede al proprio account viene creata una variabile $_SESSION[“username”] con il valore *username*, questa verrà utilizzata in diverse pagine per mostrare l’username dell’utente oppure verrà utilizzata come dato per le query in php.

Nel caso l’account sia di tipo amministratore, oltre alla sessione con l’username, verrà creata una $_SESSION[“admin”] con il valore *yes*, quest’ultima viene utilizzata per mostrare nella pagina profile.php gli elementi che permettono l’aggiunta di un autore o di un libro.

Da notare che se $_SESSION[“username”] non è impostata non si possono cercare libri, di conseguenza se si provasse ad aprire la scheda di ricerca dall’index si verrebbe riportati indietro con un errore che viene generato grazie alla variabile $_SESSION[“message”], specificando che bisogna fare login o signup per poter utilizzare il sito.

Per chiudere la sessione ci sono diverse modalità:

- Cliccare la voce di Log Out dal banner, in questo caso si aprirà una pagina che mostra un timer dopo il quale si verrà riportati all’index. Nel file logout.php vengono fatte: session_regenerate_id(TRUE), session_unset() e session_destroy();
- Aprire dall’index la pagina di login o signup, in entrambi i casi la sessione viene unsettata con session_unset() nel caso si stia tentando di accedere con un nuovo account essendo già loggati e, successivamente, accedendo o creando un nuovo account, verrà assegnato un nuovo valore a $_SESSION[“username”].

RIFERIMENTI E NOTE

Tutte le immagini/icone, tranne il logo, sono state prese dal sito: https://icons8.com/

Il sito è stato provato nei browser Google Chrome, Microsoft Edge, Opera, Safari e Mozilla Firefox e funziona correttamente, tuttavia a causa dell’utilizzo di jquery-ui e di FormData, non funziona su browser Internet Explorer < 11.

I documenti html e css sono stati verificati e non è stato trovato nessun errore.

(*) I caratteri speciali non ammessi sono: `@#$%()_\=\[\]{};':"\\|,<>\/. La verifica di questi, delle stringhe vuote e dei whitspaces avviene nelle funzioni nel file functions.js.*

(**) La funzione checkMain() funziona solo con dei protagonisti pre-impostati, perché ho preferito renderla non scalabile.