Progetto di Tecnologie Web

Elisa Solinas

811737

https://gitlab2.educ.di.unito.it/st159373/t-web

A.A. 2018/2019

1 Tema del sito

Il sito realizzato costituisce una piattaforma per l'acquisto di musica heavy metal in formato digitale.
La home page del sito presenta una galleria di immagini che mostra le copertine dei dischi più venduti, mentre mediante la barra di navigazione del sito è possibile visualizzare i prodotti appartenenti a un dato genere musicale.
L'utente può, inoltre, visualizzare gli acquisti precedentemente effettuati, inserire un prodotto nella "wishlist" e salvarlo per un acquisto successivo, oppure acquistarlo immediatamente.

2 Le sezioni principali

- **Accesso al sito**: mediante la pagina di accesso, l'utente può effettuare il login oppure la registrazione. In quest'ultimo caso, i suoi dati verranno inseriti nel database e il login verrà effettuato automaticamente.
- **Home page**: da questa pagina, l'utente può visualizzare i prodotti più venduti.
- **Navigazione per genere**: questa pagina permette di visualizzare tutti gli album del genere selezionato.
- **Pagina del prodotto**: questa pagina permette di visualizzare il prodotto selezionato e di acquistarlo (oppure aggiungerlo alla wishlist) trascinando la copertina del disco sull'area appropriata.
- **Prodotti acquistati**: questa pagina permette di visualizzare i prodotti che sono già stati acquistati dall'utente.
- **Wishlist**: questa pagina permette di visualizzare i prodotti che l'utente ha salvato per un eventuale acquisto.

3 Funzionalità

- **Login**: l'utente effettua il login inserendo la propria email e la password nell'apposito form della pagina iniziale.
  Quando l'utente clicca sul bottone `Login`, viene inviata una richiesta di tipo `POST` AJAX al server, che verifica che le informazioni fornite siano valide e che corrispondano a quelle presenti nel database e setta la variabile `$_SESSION["user"]`.
  A questo punto la funzione JavaScript reindirizza l'utente alla home page del sito.
- **Logout**: l'utente può effettuare il logout in qualsiasi momento, cliccando sul pulsante `logout` presente nella barra di navigazione.
  Al click, viene eseguita la funzione `php-server/logout.php`, che, dopo aver distrutto le variabili di sessione, reindirizza l'utente alla pagina di accesso al sito.
- **Registrazione**: l'utente si registra inserendo la propria email e una password nell'apposito form della pagina iniziale.
  Quando l'utente clicca sul bottone `Subscribe`, viene inviata una richiesta di tipo `POST` AJAX al server, che verifica che le informazioni fornite siano valide, inserisce il nuovo utente nel database ed effettua il login (settando la variabile `$_SESSION["user"]`).
  A questo punto la funzione JavaScript reindirizza l'utente alla home page del sito.
- **Gestione del contenuto generato dall'utente**: l'utente può organizzare i prodotti a cui è interessato in una wishlist (che viene memorizzata sul database), oppure acquistare immediatamente un prodotto (che viene inserito nello storico dei prodotti acquistati).

4 Caratteristiche

- **Usabilità**: il sito è stato realizzato cercando di seguire le linee guida per una buona usabilità. In quest'ottica, si è evitato l'utilizzo di pop-up, messaggi di errore difficili da comprendere per l'utente e assenza di feedback dopo un'azione dell'utente.

- **Interazione/animazione**: l'animazione è stata realizzata nella funzione di aggiunta di un prodotto alla wishlist o di acquisto del prodotto.
  L'utente vede a sinistra il prodotto selezionato e a destra, due *"box"* che riportano rispettivamente la dicitura *"Trascina qui per acquistare"* e *"Trascina qui per aggiungere il prodotto alla wishlist"*.
  L'utente, mediante drag and drop, trascina la copertina dell'album sul box dell'azione che intende effettuare.

- **Sessioni**: la sessione dell'utente viene aperta mediante login (oppure registrazione) e viene chiusa, dall'utente stesso, mediante il click sul pulsante logout (oppure, come da default per PHP, dopo 24 minuti).

- **Interrogazione del database**: per l'interrogazione del database, effettuata unicamente lato server, è stata usata la classe PDO, in modo da poterne sfruttare le caratteristiche di sicurezza (per evitare gli attacchi di tipo SQL Injection) e di efficienza.

- **Validazione dei dati in input**: i dati inseriti dall'utente vengono validati lato client utilizzando i vincoli forniti da HTML5, cioè `required` e `type`, che impediscono l'invio del form nel caso in cui i campi non siano stati compilati, oppure i dati inseriti non siano del formato adeguato (in particolare, nel caso in cui l'email inserita non sia valida e/o la password non rispetti la lunghezza minima di 8 caratteri).
  Anche lato server si verifica che i campi siano stati compilati, che l'email inserita sia valida e che la password rispetti la lunghezza minima di 8 caratteri, e, inoltre, tutte le query del database che prevedono l'utilizzo di dati inseriti dall'utente vengono realizzate mediante il metodo `prepare` della classe PDO.

- **Presentazione**: il layout del sito è volutamente aggressivo (testo neon su sfondo nero) in accordo con la tematica del sito e si è utilizzato Bootstrap 4 in modo tale da renderlo responsive.

5 Front-end

- **Separazione presentazione/contenuto/comportamento**: il codice HTML di ogni pagina del sito (**contenuto**) è inserito in un file *`filename`*`.php`, il quale include come riferimenti il foglio di stile `style.css` (**presentazione**) e il file contenente gli script utilizzati dalla pagina, *`filename`*`.js` (**comportamento**).

- **Soluzioni cross-platform**: Il sito è stato testato sulle ultime versioni dei seguenti browser:

  - Safari
  - Google Chrome
  - Firefox
  - Opera

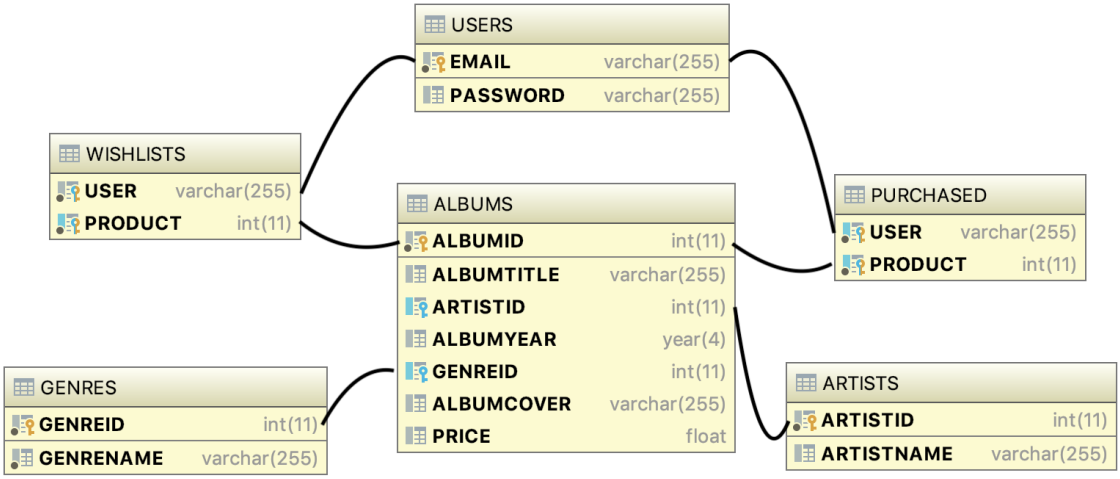
  Si è fatto affidamento alle funzionalità di jQuery e di Bootstrap 4 per risolvere i problemi di compatibilità con eventuali browser più datati.

- **Organizzazione file e cartelle di progetto**: il progetto è suddiviso nelle seguenti cartelle:

  - `css`: questa cartella contiene il solo file `style.css`.
  - `files`: questa cartella contiene le immagini di copertina degli album presenti nel database.
  - `js`: questa cartella contiene i file JavaScript, ciascuno dei quali ha il nome della pagina web che deve eseguirne il codice.
  - `librerie`: questa cartella contiene i file dei framework jQuery, Bootstrap (sia CSS che JS) e Popper.js (richiesta da Bootstrap).
  - `php-sections`: questa cartella contiene i file `php` delle porzioni di codice che vengono riutilizzate più volte dalle pagine (`head`, `footer`, e `navbar`).
  - `php-server`: questa cartella contiene i file delle funzioni `php` che vengono eseguite dal server, compreso il file `database.php` che include tutte le funzioni di accesso al database.
  - `php-views`: questa cartella contiene i file `php` delle pagine del sito.

6 Back-end

- **Architettura generale classi/funzioni PHP**: le funzioni PHP vengono richiamate mediante chiamate JavaScript/AJAX e sono inserite all'interno di file che ne portano il nome.
  Fanno eccezione le funzioni che hanno il compito di interrogare il database, inserite nel file `database.php`
- **Schema del database**:



- **Descrizione delle funzioni remote**:
  - `add_to_purchased.php`: viene invocata quando l'utente acquista un prodotto.
    Riceve in input la variabile `$_POST["album"]`, che contiene l'id dell'album da inserire tra gli acquisti e utilizza la variabile `$_SESSION["user"]` (se questa non è impostata significa che l'utente non è loggato, quindi si viene reindirizzati alla pagina di accesso).
    Invia una richiesta al database che inserisce il prodotto tra gli acquistati e restituisce (in formato JSON), in caso di successo, il codice 200 alla richiesta AJAX, in caso di fallimento , il codice 404 con un messaggio di errore.
  - `add_to_wishlist.php`: viene invocata quando l'utente inserisce un prodotto nella sua wishlist.
    Riceve in input la variabile `$_POST["album"]`, che contiene l'id dell'album e utilizza la variabile `$_SESSION["user"]` (se questa non è impostata significa che l'utente non è loggato, quindi si viene reindirizzati alla pagina di accesso).
    Invia una richiesta al database che inserisce il prodotto nella wishlist e restituisce (in formato JSON), in caso di successo, il codice 200, in caso di fallimento , il codice 404 con un messaggio di errore.
  - `delete_from_wishlist.php`: viene invocata quando l'utente clicca sul bottone `Delete` dalla pagina che mostra la wishlist.
    Riceve in input la variabile `$_POST["album"]`, che contiene l'id dell'album e utilizza la variabile `$_SESSION["user"]` (se questa non è impostata significa che l'utente non è loggato, quindi si viene reindirizzati alla pagina di accesso).
    Invia una richiesta al database che rimuove il prodotto dalla wishlist e restituisce (in formato JSON), in caso di successo, il codice 200 e la copia della wishlist aggiornata, in caso di fallimento , il codice 404 con un messaggio di errore.
  - `get_album.php`: viene invocata quando l'utente clicca su un album per visualizzarne i dettagli.
    Riceve in input la variabile `$_POST["album"]`, che contiene l'id dell'album (se questa non è impostata viene restituito un messaggio di errore). Il server interroga il database per ottenere tutti i dettagli sull'album (viene

generato un messaggio di errore anche nel caso in cui l'album selezionato non sia presente) e restituisce le informazioni ricavate (in formato JSON) nel caso in cui la ricerca sia andata a buon fine.

- `get_bestsellers.php`: questa funzione restituisce, in formato JSON, gli album più venduti presenti nel database.
  In caso di errore, viene generato un messaggio che informa l'utente.
- `get_by_genre.php`: riceve in input la variabile `$_POST["genre"]`, che contiene l'id del genere musicale selezionato dall'utente (se questa non è impostata viene restituito un messaggio di errore) e, interrogando il database, restituisce la lista degli album di tale genere in formato JSON.
  Nel caso in cui si verifichi un errore, viene generato un messaggio che informa l'utente.
- `get_genres.php`: questa funzione restituisce una lista, in formato JSON, di tutti i generi presenti nel database.
  Nel caso in cui si verifichi un errore, viene generato un messaggio che informa l'utente.
- `get_purchased.php`: utilizzando la variabile `$_SESSION["user"]`, resituisce la lista degli album acquistati dall'utente.
  Se la variabile `$_SESSION["user"]` non è impostata significa che l'utente non è loggato, quindi si viene reindirizzati alla pagina di accesso.
  Nel caso in cui si verifichi un errore, viene generato un messaggio che informa l'utente.
- `get_wishlist.php`: utilizzando la variabile `$_SESSION["user"]`, resituisce la lista degli album che l'utente ha inserito nella sua wishlist.
  Se la variabile `$_SESSION["user"]` non è impostata significa che l'utente non è loggato, quindi si viene reindirizzati alla pagina di accesso.
  Nel caso in cui si verifichi un errore, viene generato un messaggio che informa l'utente.
- `login.php`: questa funzione riceve in input le variabili `$_POST["email"]` e `$_POST["password"]` e ne verifica la validità.
  Nel caso in cui si verifichi un errore, viene generato un messaggio che informa l'utente.
- `logout.php`: questa funzione riceve in input la variabile `$_SESSION["user"]`, distrugge le variabili di sessione e reindirizza l'utente alla pagina di accesso al sito.
  Nel caso in cui si verifichi un errore, viene generato un messaggio che informa l'utente.
- `subscribe.php`: questa funzione riceve in input le variabili `$_POST["email"]` e `$_POST["password"]`, ne verifica la validità e verifica che l'utente non sia presente nel database.
  Nel caso in cui si verifichi un errore, viene generato un messaggio che informa l'utente.

- **Gestione delle condizioni di errore**: le condizioni di errore vengono gestite come segue:
  - Errore nella connessione al server o nella chiamata AJAX: l'utente viene reindirizzato alla pagina di accesso al sito, dove visualizza un messaggio di errore che lo invita a riprovare.
  - Errore nell'input dei dati: viene visualizzato un messaggio all'interno del form che spiega l'errore commesso.

- **Funzioni di callback lato Javascript/AJAX**:
  - `add_to_purchased`: questa funzione (richiesta di tipo `GET`) viene invocata nel momento in cui l'utente trascina la copertina dell'album da acquistare nell'apposita area, e richiede al server l'aggiunta dell'album agli acquisti.
    Il server risponde indicando il successo o il fallimento dell'operazione.
  - `add_to_wishlist`: questa funzione (richiesta di tipo `GET`) viene invocata nel momento in cui l'utente trascina la copertina dell'album da acquistare nell'apposita area, e richiede al server l'aggiunta dell'album alla wishlist.
    Il server risponde indicando il successo o il fallimento dell'operazione.
  - `delete_from_wishlist`: questa funzione viene invocata nel momento in cui l'utente clicca sul pulsante `Delete` di un elemento della wishlist.
    Viene inviata una richiesta al server (di tipo `POST`), includendo l'ID dell'album da cancellare, e il server risponde con una copia della wishlist aggiornata, che viene ricaricata su schermo.
    In caso di errore, l'utente viene informato con un messaggio.

- `get_album`: questa funzione permette di visualizzare i dettagli di album.
  Viene inviata una richiesta al server (di tipo `POST`), includendo l'ID dell'album e il server risponde con le informazioni richieste.
  In caso di errore, l'utente viene informato con un messaggio.
- `get_bestsellers`: questa funzione permette di visualizzare gli album più venduti.
  Viene inviata una richiesta al server (di tipo `GET`), e il server risponde (in caso di successo) con le informazioni richieste.
  In caso di errore, l'utente viene informato con un messaggio.
- `get_by_genre`: questa funzione permette di visualizzare tutti gli album di un dato genere.
  Viene inviata una richiesta al server (di tipo `POST`), inviando l'ID del genere selezionato, e il server risponde (in caso di successo) con la lista richiesta.
  In caso di errore, l'utente viene informato con un messaggio.
- `get_genres`: questa funzione viene invocata nel momento in cui viene caricata la barra di navigazione e permette di popolare il menu dropdown con i generi presenti nel database.
  Si tratta di una richiesta di tipo `GET`, che restituisce la lista in caso di successo e un messaggio di errore in caso di fallimento.
- `get_purchased`: questa richiesta (di tipo `GET`) riceve la lista degli album acquistati da un dato utente, in caso di successo, mentre in caso di fallimento restituisce un messaggio di errore.
- `get_wishlist`:questa richiesta (di tipo `GET`) riceve la wishlist di un dato utente, in caso di successo, mentre in caso di fallimento restituisce un messaggio di errore.
- `login`: questa richiesta invia al server (mediante `POST`) l'email e la password inseriti dall'utente: in caso queste non siano valide (non esiste alcun utente con tale email o la password non è quella giusta per quell'utente), il server risponde con un messaggio di errore. Nel caso, invece, la richiesta vada a buon fine, l'utente viene reindirizzato alla home page del sito.
- `subscribe`: questa richiesta invia al server (mediante `POST`) l'email e la password inseriti dall'utente: in caso queste non siano valide (la password non è valida oppure esiste già un utente con tale indirizzo email), il server risponde con un messaggio di errore. Nel caso, invece, la richiesta vada a buon fine, l'utente viene reindirizzato alla home page del sito.