UNIVERSITÀ DEGLI STUDI DI TORINO
DIPARTIMENTO DI INFORMATICA

TECNOLOGIE WEB

Progetto: **e-commerce di libri**

Studente: **Fagioli Paolo**
Matricola: **859461**

Anno Accademico 2019/2020

Tema del sito

Attraverso TheBookstore si è voluto realizzare un sito e-commerce per l'acquisto di libri. Per poter accedere al sito è necessaria l'iscrizione, successivamente si può iniziare ad aggiungere libri al carrello o nella propria wishlist. Ogni utente può cercare nella libreria online il libro che desidera, cliccarci sopra e vedere tutte le informazioni, come la valutazione media fornita dagli utenti, la trama ed eventuali commenti rilasciati, utili per farsi un'idea sul prodotto che andrà ad acquistare.

Sezioni principali

- **Autenticazione**
  La sezione di autenticazione permette all'utente di accedere/registrarsi al sito (è prevista la successiva possibilità di uscire/disiscriversi dal sito).

- **Home**
  La pagina principale dove l'utente, una volta autenticato, può iniziare a cercare un libro all'interno della libreria online. Per rendere la ricerca più efficiente è stata introdotta la possibilità di ordinare i libri in base ad alcuni criteri come titolo, autore, prezzo.

- **Book (Sezione libro)**
  La sezione libro contiene tutte le informazioni riguardo il prodotto che si desidera acquistare, è possibile l'aggiunta al carrello, fornire una valutazione su una scala da 1 a 5, inserire dei commenti e nel caso in cui non lo si voglia acquistare subito è prevista la possibilità di aggiungerlo nella propria lista dei desideri.

- **Profile (Sezione utente)**
  Questa sezione prevede la pagina personale dell'utente, dove possono essere visualizzati i libri presenti nella propria lista dei desideri e quelli acquistati.
  Inoltre, da qui, è possibile accedere a due micro-sezioni quali:
  - **pagamenti**: per l'aggiunta del metodo di pagamento (per rendere successivamente l'acquisto più veloce)
  - **impostazioni**: qui l'utente può:
    1. aggiungere/cambiare la propria foto profilo
    2. cambiare la password
    3. disiscriversi dal sito

- **Cart (Carrello)**
  in questa sezione è possibile visualizzare tutti i prodotti aggiunti al carrello, modificare le quantità, eliminare un prodotto, svuotare il carrello o procedere al pagamento

Funzionalità

- ### Login

  I dati inseriti vengono validati lato front-end utilizzando gli attributi di HTML5 **pattern** e **required**, in supporto vi è anche la funzione JS **escapeInput().** Il file **loginSignup.js**, invia i dati al server **login_ajax.php** in modo asincrono tramite una richiesta AJAX di tipo POST. i dati lato back-end vengono sanificati attraverso la funzione filter_input(). In caso di errori viene mostrato un feedback all'utente con il tipo specifico di errore (es. email does not exist in our databases, password incorrect). In caso contrario viene inizializzata la sessione ($_SESSION['email']) e l'utente verrà reindirizzato alla pagina principale.

- ### Registrazione

  Anche in questo caso i dati vengono controllati sia lato front-end che lato back-end. **loginSignup.js** invia i dati a **signup_ajax.php** in modo asincrono tramite una richiesta AJAX di tipo POST. In questo caso potrebbe succedere che un utente tenti di registrarsi al sito con una email già registrata, allora verrà restituito un feedback (errore) del tipo ("you may be already registered as "email"), in caso contrario si procede analogamente come nel caso del login.

- ### Logout

  Il sito utilizza le sessioni PHP, poiché HTTP è un protocollo stateless e non sarebbe possibile mantenere l'utente loggato. Una volta eseguito il logout vengono liberate tutte le variabili di sessione session_unset() e distrutti i dati relativi a quest'ultima session_destroy(). successivamente si viene reindirizzati alla pagina di accesso al sito.

- ### Gestione del contenuto generato dall'utente

  L'utente ha la possibilità di svolgere diverse azioni:

  - aggiungere/rimuovere un libro al/dal carrello
  - modificare la quantità nel carrello
  - aggiungere/rimuovere un libro alla/dalla wishlist
  - valutare e commentare un libro
  - eliminare un commento
  - inserire/modificare/rimuovere l'immagine del profilo
  - aggiornare la password
  - disiscriversi

  Tutte queste azioni sono implementate attraverso richieste AJAX di tipo POST.
  Una volta arrivati al server, i dati vengono processati ed in questo momento entra in gioco il database per rendere persistenti le modifiche.

Caratteristiche

- ## Usabilità

  Per assicurare un'ottima leggibilità, durante la progettazione è stato deciso di usare testo nero su sfondo bianco. Il sito si basa sul principio del "**visibility and feedback**":
  È dotato di un design minimal in modo da aiutare l'utente a compiere il suo obiettivo nel modo più veloce ed efficiente possibile.

  - **visibility**:
    homepage con solo barra di ricerca senza rumore di sottofondo;
    barra dei menù con le pagine principali etc..
  - **feedback:**
    Sia in caso di successo che in caso di fallimento il sistema comunica un riscontro all'utente. ad esempio errori di login/registrazione, successo nell'aggiunta del metodo di pagamento, input validi evidenziati in verde.

- ## Interazione/animazione

  - **filtri e ordinamento:**
    l'utente ha la possibilità di utilizzare i filtri per visualizzare solo i libri che fanno parte di una certa categoria/genere letterario. Può anche decidere il tipo di ordinamento, che può essere in base al titolo (opzione di DEFAULT), autore, dal prezzo più basso, dal prezzo più alto.

  - **sistema di valutazione a cinque stelle** ★★★★★ **:**
    Una volta che l'utente accede alla pagina personale di un libro avrà la possibilità di conoscere la valutazione media per quel libro (in termini di stelle). Se il libro non ha ancora ricevuto nessuna valutazione appariranno cinque stelle vuote. Effettuando un mouseover l'utente illuminerà le stelle dalla prima a sinistra fino a quella posizionata sotto il puntatore. In questo momento l'utente potrà decidere di cliccare (in questo modo valuterà il libro). Il voto verrà salvato nel database ed apparirà una voce "your rate: x". Una volta effettuato il mouseleave, verrà mostrata la nuova valutazione media, che ora tiene conto del voto appena inserito.

- ## Sessioni

  La sessione viene aperta durante l'accesso al sito da parte dell'utente (tramite registrazione o login). A questo punto verrà inizializzata una variabile di sessione $_SESSION['email'] contenente l'email dell'utente. Per mantenere l'utente loggato e per recuperare i dati relativi alla sessione, ogni pagina del sito esegue le funzioni session_start() e ensure_logged_in(). Se non vi è una sessione esistente, la session_start() creerà una nuova sessione, da ensure_logged_in() risulterà l'array

associativo $_SESSION vuoto e si verrà reindirizzati nella pagina di accesso per effettuare l'autenticazione. La sessione verrà distrutta al logout.

- ## Interrogazione del db

  All'interno del sito sono presenti numerose query per l'interrogazione del database, la maggior parte di queste avviene a seguito di richieste AJAX. Sono presenti query di selezione, accodamento, aggiornamento ed eliminazione.

- ## Validazione dati input

  La validazione dei dati in input avviene sia lato client che lato server.

  - **lato client**:
    Per la validazione dei form sono stati utilizzati gli attributi **required** (per evitare di lasciare il campo vuoto) e **pattern** (per fare in modo che rispettasse le richieste) di HTML5, con supporto attraverso una funzione Javascript **escapeInput().**
  - **lato server**:
    Poiché troppo rischioso estrarre parametri direttamente da richiesta HTTP, è stata utilizzata la funzione **filter_input()**, che filtra l'input in modo robusto, In particolare utilizzata con il filtro FILTER_SANITIZE_FULL_SPECIAL_CHARS.
    Come secondo strato di validazione è stata utilizzata per le email la funzione **filter_var()** con filtro FILTER_VALIDATE_EMAIL.

- ## Sicurezza

  Come conseguenza della validazione degli input attacchi di tipo XSS e SQL Injection sono resi vani.
  Le password degli utenti prima di essere salvate su database vengono cifrate utilizzando l'algoritmo BCRYPT. Per incrementare la sicurezza si è deciso di utilizzare **password salted**, in questo modo se due utenti condividono la stessa password es "paperino", poiché le due stringhe (sale) sono diverse, le due password criptate risulteranno diverse.
  Se un avversario dovesse rubare le password, una password salted risulterà molto più difficile da indovinare rispetto ad una unsalted. Si ha in questo modo un incremento esponenziale in termini di sicurezza.

- ## Presentazione

  Il sito è dotato di un layout responsive con l'utilizzo di un hamburger menù. In diverse pagine, come il profilo, la pagina del singolo libro, le impostazioni e i pagamenti, è stato implementato uno schema a due colonne.

Front-end

- **Separazione presentazione/contenuto/comportamento (stile unobtrusive)**
  Per evitare confusioni tra contenuto e comportamento è stato adottato lo stile unobtrusive separando il codice HTML/PHP dal codice Javascript. Allo stesso modo tutti i file inerenti la presentazione si trovano in una cartella separata in modo da evitare mescolanze tra CSS e HTML/PHP (foglio di stile esterno e collegato all'HTML, anziché incorporato tramite tag <style>). Invece di inserire contenuto CSS nei file JS si è preferito utilizzare quest'ultimo per aggiungere o rimuovere classi, avendo come conseguenza il cambiamento delle regole CSS applicate.

- **Soluzioni cross-platform**
  Per rendere il sito responsive sono state usate le media rules di CSS su alcuni elementi come la barra dei menù (navbar) con uso dell'hamburger menu e la barra di ricerca. Anche lo stile a due colonne viene presentato in maniera diversa a seconda delle dimensioni del dispositivo che si utilizza, ad esempio utilizzando un computer le due colonne appariranno affiancate, mentre utilizzando un telefono verranno disposte in modo sequenziale (prima una e poi l'altra).

Back-end

- **Architettura generale delle classi/funzioni php**
  - **index.php:** pagina che permette l'accesso al sito
  - **./php**: la cartella php contiene file liberi che sono le pagine del sito web, inoltre contiene altre 3 cartelle: **ajax**, **functions**, **include**.
  - **php/ajax**:
    Contiene tutti i file necessari per effettuare le risposte AJAX, questi all'interno sono suddivisi per argomento per essere più facilmente rintracciabili: ad esempio la cartella **cart** conterrà i file php coinvolti nelle risposte AJAX che interessano il carrello come add_to_cart e remove_from_cart.
  - **php/functions:**
    Nella cartella functions sono presenti tutte le funzioni comuni utilizzate da più di un file. Per i file all'interno di questa cartella è stata utilizzata una nomenclatura **common_ + "argomento"** in modo da rendere evidente il tema che accomuna le funzioni raggrupate nello stesso file.
  - **php/include**:
    La cartella include contiene i file per la connessione al database, l'header e il footer comuni a tutte le pagine del sito.

Schema del database

Il database è composto da nove tabelle:

1. **BookGenres**(genre)

2. **Books**(book_id, title, genre, trama, author, published_year, cover, price)
   Books(genre) referenzia BookGenres(genre)

3. **Cart**(insertion_id, user, item)
   Cart(user) referenzia User(email)
   Cart(item) referenzia Books(book_id)

4. **Comments**(id, user, item, comment, date)
   Comments(user) referenzia Users(email)
   Comments(item) referenzia Books(book_id)

5. **Payments**(user, card_number, card_type, expiry_date, cvv, card_holder)
   Payments(user) referenzia Users(email)

6. **Purchases**(id, user, item, quantity)
   Purchases(user) referenzia Users(email)
   Purchases(item) referenzia Books(book_id)

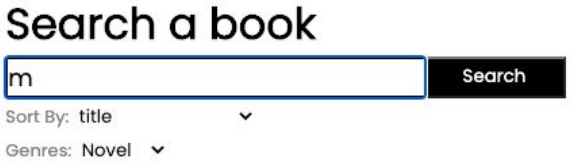
7. **Rating**(item, user, rate)
   Rating(item) referenzia Books(book_id)
   Rating(user) referenzia Users(email)

8. **Users**(email, name, surname, pwd, image*)

9. **Wishlist**(user, item)
   Wishlist(user) referenzia Users(email)
   Wishlist(item) referenzia Books(book_id)

Descrizione delle funzioni remote (API)

Esempio di funzione remota descritto: è la funzione **search_book.php** che recupera tutti i libri che sono conformi allo standard definito dalla query scritta dall'utente nella barra di ricerca della homepage (home.php).



- **Modo di utilizzo**:
  Metodo **POST** su localhost:8888/tweb/php/ajax/book/search_book.php

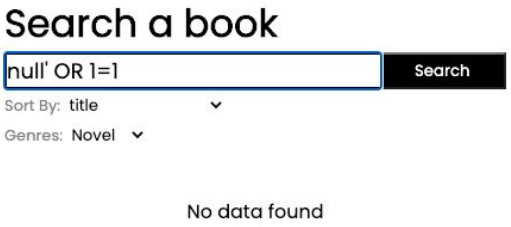
- **Sintassi output**:
  Le informazioni prodotte sono successivamente codificate in stringa JSON dalla funzione json_encode


```
{
   "success": 1,
   "data": [
       {
           "book_id": "2",
           "title": "Frankenstein",
           "genre": "Novel",
           "trama": "I saw the pale student ...",
           "author": "Mary Shelley",
           "published_year": "1994",
           "cover": "../images/book_covers/frankenstein.jpg",
           "price": "6.00"
       },
       {
           "book_id": "7",
           "title": "Moby Dick",
           "genre": "Novel",
           "trama": "At the heart of Moby-Dick ...",
           "author": "Herman Melville",
           "published_year": "2013",
           "cover": "../images/book_covers/moby_dick.jpg",
           "price": "5.47"
       }
   ]
}
```


- **Condizioni di errore**:
  Verrà mostrato all'utente un feedback, Javascript farà comparire la stringa "We've got a problem with our servers! Try again later.".

  Nel caso in cui la ricerca non dovesse dare nessun risultato, la sintassi dell'output sarà del seguente tipo


```
{
   "success": 0,
   "error": "No data found"
}
```

- **Funzioni di Callback**:
  Nel caso in cui response.success = 1 verrà eseguita la funzione **showBooks(p),** passando response.data (ovvero l'array di libri) come parametro. Questa si occuperà di mostrare i libri inserendo del codice HTML nel DOM.
  Il caso soprastante in cui response.success = 0 Javascript si occuperà di restituire il feedback all'utente.