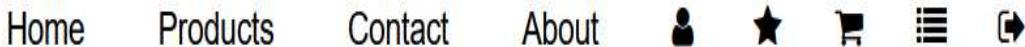


Tecnologie Web (6 CFU) – MFN0634

SHOP SHOES

Il sito “Shop Shoes” è uno store online che si occupa della vendita di scarpe da calcio per tutti i tipi di terreni. L’obiettivo del sito è quello di fornire all’utente un ventaglio, quanto più ampio possibile, di scelte e guidarlo verso la scelta migliore. Le sezioni principali del sito sono quattro e sono visualizzabili sulla pagina home solo dopo aver eseguito il login o la registrazione dell’utente al sito:



- La sezione Home mostra un’immagine stilizzata che funge da presentazione del sito;
- La sezione Products mostra l’intero catalogo dei prodotti in vendita sul sito con relative informazioni principali eventualmente espandibili per avere informazioni dettagliate;
- La sezione Contact mostra un form che l’utente deve compilare per inviare un feedback all’azienda che può contenere pareri/idee/suggerimenti;
- La sezione About che presenta molto in breve il creatore dell’azienda e il motivo della creazione.

FUNZIONALITA’

L’utente accede al sito tramite la pagina index.php la quale offre la possibilità di effettuare il login per utenti già iscritti al sito o la registrazione ai nuovi utenti:

- **LOGIN:** Attraverso la pagina login.php l’utente può inserire username e password nell’apposito form e accedere così alla pagina home.php(cuore del sito) se i dati inseriti sono corretti. Se i dati non risultano essere corretti l’utente viene opportunamente avvisato dell’errore e invitato a riprovare. È impossibile inviare il form finché i dati immessi non rispettano specifici pattern sia per l’username che per la password.
- **REGISTRATION:** Attraverso la pagina registration.php l’utente può, a fronte di un inserimento corretto dei dati necessari, effettuare la registrazione al sito. Eventuali errori nell’inserimento vengono notificati all’utente che viene invitato a riprovare. È impossibile inviare il form finché i dati immessi non rispettano i relativi pattern mostrati all’utente in caso di errore o in caso di passaggio del mouse sul relativo campo.
- **LOGOUT:** Nella pagina home.php è presente un’icona che rappresenta il logout e che al click esegue il logout corretto dell’utente rimandandolo alla pagina index.php e eliminando le variabili di sessione dello stesso.

Il sito prevede due ruoli: il root (gestore del sito) e l’user (normale utente).

Il root non può registrarsi ma è registrato a priori e oltre alle funzionalità precedentemente presentate può in esclusiva:

1. Aggiungere un prodotto allo store attraverso la pagina insertProduct in seguito alla compilazione di un opportuno form. Anche qui il root è guidato nell’inserimento dei dati. È impossibile inserire dati mal formattati o prodotti duplicati.
2. Rimuovere un prodotto fra quelli presenti nello store. Il root sceglie uno dei prodotti nella pagina removeProduct o seleziona l’icona “minus” sulla card relativa al singolo prodotto in

home. Verrà eliminato un prodotto in caso di scelta valida di un prodotto altrimenti non verrà eseguita nessuna azione. In ogni caso verrà fornito un riscontro al root user.

3. Visualizzare tutti gli ordini effettuati dagli utenti attraverso la pagina order.php. Per ogni ordine vengono mostrati username dell'utente, prezzo totale e articoli che lo compongono.
4. Visualizzare, per ogni prodotto in vendita, le informazioni dettagliate e le reviews ricevute attraverso la pagina singleProduct.php. Per ogni review il root può decidere di eliminarlo.

L'user, oltre alle funzionalità comuni al root, può:

1. Decidere di aggiungere o rimuovere uno o più prodotti dalla/alla sua lista dei preferiti. In seguito può accedere a tale lista attraverso la stella presente sulla navigation bar. Qui può rimuovere articoli dalla lista, aggiungerli direttamente al carrello o visualizzare informazioni dettagliate sui singoli prodotti.
2. Decidere di aggiungere o rimuovere uno o più prodotti al/dal suo carrello. In seguito può accedere al carrello attraverso il carrello presente sulla navigation bar. Qui può: scoprire il costo totale degli articoli nel carrello, completare l'ordine, rimuovere elementi o ottenere informazioni dettagliate sui singoli prodotti.
3. Visualizzare lo storico di tutti gli ordini che ha completato tramite l'icona lista presente sulla navigation bar. Accederà con un click sulla stessa ad una pagina dedicata nella quale visualizzerà informazioni dettagliate per i singoli ordini, se ce ne sono.
4. Visualizzare, attraverso l'icona informazioni sulle card dei prodotti, le informazioni relative al singolo prodotto e decidere di aggiungere il prodotto al carrello se non già presente o condividerlo con altre persone. Sempre sulla stessa pagina l'utente può visualizzare le review di altri utenti per l'articolo e eventualmente aggiungerne uno.

CARATTERISTICHE

- **Usabilità:** L'intero sito rispetta i principi del web design e in particolare:
 - Ogni funzione è facilmente accessibile e comprensibile dall'utente. Ogni funzione è auto esplicativa poiché sono state utilizzate delle icone significative per rappresentarle ma se ciò non bastasse al passaggio del mouse è presente, per ogni singola funzionalità, la descrizione in breve della stessa che appare al passaggio del mouse e scompare successivamente.
 - Ogni operazione dell'utente è seguita da opportuni riscontri positivi o negativi che siano rispettivamente con diversi colori (red o green) per renderli visibili e significativi.
 - E' stato mantenuto un costante livello di contrasto all'interno del sito per rendere la navigazione all'utente fluida, chiara e quanto più possibile priva di fastidi. I colori principalmente utilizzati sono stati: black, white, lightblue e gold. Il font utilizzato invece è il sans-serif con font-size differenti a seconda delle esigenze di presentazione ma che garantiscono sempre una facile comprensione del contenuto.
- **Interazione/animazione:** Tutte le interazioni vengono gestite mediante javascript e le interazioni che generano contenuto nuovo apportano modifiche alla vista della pagina corrente senza che ci sia bisogno di un refresh della stessa. Ad esempio, l'inserimento di un commento da parte di un utente è visualizzato immediatamente nella sezione reviews senza refresh attraverso jquery. Il sito presenta animazioni su alcuni buttoni che al click mostrano una finta simulazione (loop circolare) creata attraverso animazione bounce. Inoltre nella sezione prodotti, l'utente può visualizzare attraverso delle icone sulle singole card se il prodotto stesso è già presente in wishlist o nel carrello ed eventualmente aggiungerlo o rimuoverlo rispettivamente. Un'icona di colore gold indica la presenza del prodotto nel carrello o nella wishlist rispettivamente, di colore bianco invece l'assenza. L'animazione

principale che l'utente può però utilizzare è il drag and drop (definito in onLoad.js). Essa permette di inserire nel carrello un elemento trascinandolo nella speciale card situata alla fine della presentazione dei prodotti. Se si trascina dentro un prodotto già nel carrello, esso viene rimosso dal carrello stesso. In ogni caso viene generato un riscontro per l'utente.

- **Sessioni:** Le sessioni vengono inizializzate ad ogni login o registrazione dell'utente e vengono utilizzate per molteplici scopi: le variabili di sessione sono utilizzate per assicurare l'accesso ai contenuti ai soli utenti loggati attraverso opportuni controlli sulle stesse. In caso di controlli negativi l'utente viene avvisato e rimandato alla pagina login o registration. Fra le variabili di sessione vi è anche _SESSION["typeof"] che indica il tipo di utente (root o user) e permette o nega l'accesso agli utenti non root a determinate sezioni critiche del sito. È inoltre utilizzata una variabile _SESSION["error"] per trasmettere messaggi di errore fra una pagina e un'altra dopo una redirect. La sessione con relative variabili viene distrutta nel momento del logout o nel momento in cui si verifica un errore. Per ogni diverso utente, viene quindi creata, gestita e infine cancellata una sessione diversa .
- **Validazione dati in input:** Tutti i dati dei form quindi immessi da un utente vengono validati attraverso regular expression sia in HTML che in JS oltre che in php attraverso funzioni di filtraggio. Le espressioni regolari in HTML/JS utilizzate sono: "[a-z]{4,16}" per l'username; "[A-Z]{1}[a-z]{7}[0-9]{1}[^1]" per la password; "[a-z0-9._%+-]+@[a-z0-9.-]+\.[a-z]{2,4}\$" per l'email; "[A-Z]{1}[a-z]{1,16}|[A-Z]{1}[a-z]{1,16}\s[A-Z][a-z]{1,14}" per il nome; "[a-z]{0,1}'{0,1}[A-Z]{1}[a-z]{1,32}" per il cognome; "\d{1,2}\.\d{1,2}" per l'altezza dei tacchetti nei nuovi prodotti ; "\d{1,6}\.\d{1,2}" per il prezzo dei nuovi prodotti; /^[A-Z][a-z]{3,9}\s?\w{0,5}\$/ per controllare l'inserimento di un terreno corretto per un prodotto; /^[A-Z][a-z]{6,8}\s?\w{0,7}\$/ per controllare un valore corretto per il materiale di un prodotto; infine attraverso appositi controlli viene verificato che il file inserito sia un'immagine (in insert.js).
- **Sicurezza:** Sono state adottate tutte le misure per evitare falle nella sicurezza. Ogni password viene inserita nel database dopo essere cifrata tramite algoritmo MD5. Tutti gli input di tutti i form subiscono un triplice controllo (HTML, JS, PHP) e vengono sempre "quotati" attraverso la funzione quote prima di ogni utilizzo nelle query. Queste misure garantiscono una sicurezza tale da evitare attacchi XSS ed HTML/SQL Injection.
- **Presentazione:** Tutte le componenti delle pagine sono state pensate per essere chiare ed esplicative anche in seguito ad una ridimensione della pagina stessa. Sono state utilizzate maggiormente misure percentuali e si è utilizzato un display flessibile che rende le componenti responsive ai ridimensionamenti. Si è scelta una single page per la home del sito e la si è divisa in 4 sezioni principali racchiuse fra navigation bar e footer. Per tutte le altre funzionalità invece si sono utilizzate pagine diverse a seconda della richiesta. Si è inoltre scelta una posizione sticky per la navigation bar così da lasciare i controlli all'utente sempre facilmente raggiungibili e utilizzabili.
- **Interrogazione del database:** Le interrogazioni al database sono molteplici e per molteplici scopi e sono tutte mediate da AJAX. Tutte le funzioni utilizzate sono racchiuse nel file functions.php . Si accede alle singole funzioni attraverso un filtro mediato dal campo action inizializzato in _POST quando viene generata la chiamata AJAX. Per tale motivo ogni singola richiesta AJAX sfrutta il metodo POST. Tutte le funzioni presenti in questo file sfruttano la funzione dbConnect() che esegue la connessione al database, setta l'attributo ERRMODE EXCEPTION e restituisce il PDO relativo alla connessione. Ogni query è quindi racchiusa in un costrutto try/catch che cattura le PDOException quando avvengono e le gestisce di conseguenza.

FRONT-END

-Separazione presentazione/contenuto/comportamento: In tutto il codice è stato adottato uno stile unobtrusive e in particolare: i file html sono privi di codice css e privi per la quasi totalità di codice PHP. Quasi perché PHP viene utilizzato per: creare/gestire le sessioni e quindi necessario in minima parte all'interno del codice html; gestire le inclusioni di file che si differenziano a seconda che la richiesta di caricamento della pagina venga eseguita dal root o da un user. Il codice javascript è slegato totalmente dal codice html/php e non presenta regole grafiche se non l'aggiunta e la rimozione dinamica di classi di stile per alcuni elementi necessarie a modifiche visive o animazioni.

-Soluzioni cross-platform: Come accennato in precedenza tutte le componenti del sito sono state definite in maniera tale da non perdere chiarezza e ordine in seguito ad una ridimensione. Per ottenere questo risultato si sono usate per la maggior parte misure percentuali così da garantire un riadattamento delle stesse a seconda della dimensione della pagina e si sono definite regole @media nei file css per le componenti principali.

-Organizzazione file e cartelle di progetto:

- 1.Cartella css: contiene tutti i file css relativi alle singole pagine inclusi al caricamento delle pagine stesse e in più un file denominato common.css per definire regole comuni fra tutte le pagine
- 2.Cartella html: contiene tutti i file html che definiscono la struttura delle pagine comuni e non e vengono inclusi successivamente nel codice php attraverso la funzione include.
- 3.Cartella javascript: contiene i file javascript relativi al comportamento delle singole pagine. Ogni pagina ha un file javascript associato e per le pagine utilizzate sia dal root che dall'user ci sono due file diversi: ad esempio per una pagina comune denominata pagina1 esistono pagina1.js per l'user e pagina1Root.js per l'user root. È in più presente un file denominato commonFunctions che racchiude le funzioni comuni a più file javascript. Ogni file viene opportunamente incluso dopo attenti controlli mediati da PHP.
- 4.Cartella php: contiene i file php che definiscono le pagine e gestiscono le sessioni. Contiene inoltre il file functions che contiene tutte le funzioni utilizzate per comunicare con il database utilizzabili attraverso chiamate ajax.
- 5.Cartelle image e new_image: contengono rispettivamente le immagini utili alla presentazione e stilistica del sito una e le immagini dei prodotti in vendita l'altra.
- 6.jquery-ui-1.13.2: libreria esterna utilizzata per il drag and drop.

-Soluzioni html/css/javascript degne di nota: utilizzo di icone del gruppo fontawesome; implementazione di una funzione javascript che alterna la visualizzazione e la non visualizzazione della password in fase di login o registrazione con un semplice click sull'icona a forma di occhio locata al margine destro dell'input field; utilizzo del DOM attraverso Jquery per aggiornare la vista dopo l'interazione con l'utente che può comportare eliminazione, modifica o aggiunta del contenuto; implementazione del drag and drop per aggiungere/rimuovere elementi al/dal carrello trascinando la singola immagine. Al caricamento dei prodotti nella pagina home, quelli rispettivamente presenti nel carrello o nella wishlist dell'utente vengono marcati con un'icona dorata anziché bianca (default); questo avviene tramite un controllo sul carrello e sulla wishlist dell'utente che genera modifiche alla vista. Tutto ciò per rendere l'interazione dell'utente più fluida e comprensibile.

BACK-END E COMUNICAZIONE FRONT/BACK-END

-Back-end: il file shop_shoes.sql contiene l'intero database con annessi vincoli di chiavi esterne definiti fra le varie tabelle. L'accesso al database è eseguito da tutte le funzioni nel file functions che vengono a loro volta richiamate dai file javascript mediante richieste ajax di tipo post. L'output delle

funzioni causa il corretto caricamento a schermo della pagina e consente attraverso Jquery le modifiche dinamiche della vista senza utilizzo del click-wait-refresh.

-Schema del database:

```
users (username, name, surname, email, password, typeof)  
products (id, product_name, price, field, height_cleats, material, image)  
cart (id, username, insert_date)  
wishlist (id, username, insert_date)  
reviews (id, username, insert_date, text)  
boughth (id, username, product_name, price, total_price, date_of)
```

-Comunicazione client-server e definizione delle funzioni remote:

La comunicazione client-server avviene mediante chiamate ajax di javascript che utilizzano tutte il metodo post essendo lo stesso più sicuro del metodo get. Ogni richiesta viene quindi inviata al server che la processa e restituisce un risultato in ogni caso. Il risultato è quasi sempre un JSON ma può essere anche una stringa in base alla quale la funzione di callback javascript esegue azioni in risposta. I risultati JSON hanno tutti il formato {"name1": value1; "name2": value2; ...}. Ogni richiesta ajax che prevede un risultato di tipo JSON, inizializza il datatype come json mentre la funzione che soddisfa la richiesta prima di restituire il json setta il contentType (ossia il tipo del contenuto che si sta inviando) come json.

Le funzioni remote sono abbastanza numerose e sono le seguenti (utilizzano tutte il metodo post):

- load(): funzione priva di parametri che restituisce tutte le informazioni di tutti gli oggetti in vendita sotto forma di json per permetterne una visualizzazione all'utente;
- checkInWishlist(), checkInCart(): funzioni prive di parametri che restituiscono per l'utente, i product_name dei prodotti nella wishlist/nel carrello sotto forma di json.
- addToWishlist(), addToCart(): funzioni che prendono come parametro il product_name del prodotto da inserire e dopo averne trovato l'ID con apposita query, lo aggiungono alla wishlist/al carrello dell'utente se non c'è e lo rimuovono altrimenti.
- loadWishlist(), loadCart(): funzioni prive di parametri che restituiscono per un dato utente tutte le informazioni relative ai prodotti presenti nella wishlist/nel carrello in formato json.
- checkCart(): funzione che prende come parametro l'id di un prodotto e restituisce "already present" se il prodotto è già nel carrello per quel dato utente, "not present" altrimenti.
- completeOrder(): funzione che prende come parametro il total_price dell'ordine effettivo e inserisce l'ordine all'interno della tabella bought recuperando tutti gli elementi nel carrello dell'utente con apposita query. Il carrello ovviamente è costantemente aggiornato. La funzione ritorna "complete" in caso di successo, "not complete" altrimenti.
- removeAllCartElement(): funzione che non prende alcun parametro e che viene utilizzata per eliminare dal carrello dell'utente tutti gli articoli presenti. La funzione torna "empty" in caso di successo, "full" altrimenti.
- loadOrder(): funzione priva di parametri che permette di recuperare date_of e total_price degli ordini del singolo utente restituendoli sotto forma di json.
- retrieveItems(): funzione che prende come parametro date_of restituito da loadOrder() per trovare product_name e price relativi ad ogni singolo prodotto nell'ordine referenziato da date_of e username dell'utente. Gli articoli vengono poi restituiti sotto forma di json.
- allOrder(): funzione eseguibile dal solo root e priva di parametri che restituisce sotto forma di json, (date_of, username, total_price) per ogni ordine presente nella tabella bought.

- `retrieveAllItems()`: funzione eseguibile dal solo root che prende come parametro `username` e `date_of` di ogni ordine e restituisce `product_name` e `price` degli articoli presenti nei singoli ordini sotto forma di json.
- `findProduct()`: funzione che prende come parametro un `product_name` e ritorna tutte le informazioni di quel prodotto sotto forma di json, “not find” altrimenti (necessario per i prodotti precedentemente acquistati ma poi rimossi dal sito).
- `findComments()`: funzione che prende come parametro l’id di un prodotto e tutti i commenti relativi al prodotto stesso oltre che l’email dell’autore sotto forma di json.
- `addComment()`: funzione che prende come parametri il `text` del commento e il `product_name` del prodotto per il quale si è scritto il commento. A partire dal `product_name`, trova l’id del prodotto, recupera data e ora attuali e infine inserisce il commento nella tabella reviews. In caso di successo restituisce “`inserted`”, “`not inserted`” altrimenti.
- `findLastComment()`: funzione che prende come parametro `id` del prodotto e `date` in cui si è inserito il commento e restituisce email dell’utente e tutte le informazioni del commento racchiuse in formato json.
- `removeComment()`: funzione eseguibile dal solo root che prende in input `id` del prodotto, `username` dell’autore e `data` del commento e rimuove il commento dalla tabella reviews. Restituisce “`removed`” in caso di successo, “`not removed`” altrimenti.
- `toRemove()`: funzione priva di parametri che restituisce, in formato json, `product_name` e `id` di tutti i prodotti in vendita sul sito.
- `removeProduct()`: funzione eseguibile dal solo root che prende come parametro l’id di un prodotto e lo elimina dalla tabella products. Ritorna “`removed`” in caso di successo, “`not removed`” altrimenti.
- `insertProduct()`: funzione eseguibile dal solo root che prende come parametro `height_cleats`, `field`, `material`, `price` e `photo` (`product_name.formatoFoto`) e inserisce il nuovo elemento all’interno del database. Ritorna “`inserted`” in caso di successo, “`already`” se esiste già un prodotto con quel nome, “`not inserted`” altrimenti.
- `loginInitial()`: funzione che prende in input `username` e `password` e ritorna “`find`” se esiste un utente con quelle credenziali, “`not find`” altrimenti.
- `registrationInitial()`: funzione che prende in input `name`, `surname`, `username`, `email` e `password` di un nuovo utente e lo inserisce nella tabella users con tipo “`user`”. Ritorna “`added`” in caso di successo, “`username`” in caso di `username` già in utilizzo e “`error`” in caso di mancato inserimento.

Note: ogni qualvolta si parla di utente nelle descrizioni delle funzioni, si intende l’utente loggato le cui informazioni sono contenute nelle variabili di sessione `in_SESSION` e in particolare `_SESSION[“username”]`.