Progetto di Tecnologie Web

Federico Serra, matricola 898925

Indice

1 Informazioni generali **2**
1.1 Tema del sito . . . 2
1.2 Sezioni principali . . . 2
1.2.1 User (login/signup) . . . 2
1.2.2 Index (home page) . . . 2
1.2.3 About (biografia/contatti) . . . 2
1.2.4 Shop (e-commerce) . . . 2
1.2.5 Cart (carrello) . . . 2
1.2.6 Purchase (acquisto) . . . 2
1.2.7 Orders (ordini in corso) . . . 2

2 Funzionalità **2**
2.1 Registrazione, login e logout . . . 2
2.2 Gestione del contenuto generato dall'utente . . . 3

3 Caratteristiche **3**
3.1 Usabilità . . . 3
3.2 Interazione/animazione . . . 3
3.3 Sessioni . . . 3
3.4 Interrogazione del db . . . 3
3.5 Validazione dei dati di input . . . 3
3.6 Sicurezza . . . 3
3.7 Presentazione . . . 3

4 Front-End **4**
4.1 Separazione presentazione/contenuto/comportamento . . . 4
4.2 Soluzioni cross-platform . . . 4
4.3 Organizzazione file e cartelle di progetto . . . 4
4.4 Soluzioni html/css/javascript degne di nota . . . 4

5 Back-End e comunicazione Back-End/Front-End **4**
5.1 Architettura generale e descrizione funzioni remote, funzioni di callback . . . 4
5.1.1 Funzioni comuni - cartella: `model/` . . . 4
5.1.2 User - cartella: `model/user/` . . . 5
5.1.3 Home page/index - cartella: `model/` . . . 5
5.1.4 Shop - cartella: `model/shop/` . . . 5
5.1.5 Cart - cartella: `model/cart` . . . 5
5.1.6 Orders - cartella: `model/orders/` . . . 6
5.1.7 Purchase - cartella: `model/purchase/` . . . 6
5.2 Schema del db . . . 6

1 Informazioni generali

1.1 Tema del sito

BB Ceramics è il sito di e-commerce di una ceramista. Il sito ha per l'artista il duplice scopo di essere fonte di vendite e allo stesso tempo "biglietto da visita" sempre accessibile.

1.2 Sezioni principali

1.2.1 User (login/signup)

Il sito sarà fruibile esclusivamente agli utenti registrati, pertanto in questa pagina verrà fornita l'opzione di login e registrazione. Tentando l'accesso alle altre pagine si verrà quindi reindirizzati a questa.

1.2.2 Index (home page)

È la pagina di benvenuto: l'obiettivo è quello di suscitare curiosità agli occhi del visitatore.

1.2.3 About (biografia/contatti)

In questa sezione si potranno leggere: formazione, biografia ed esperienze lavorative dell'artista. Sarà inoltre possibile tramite un `form` digitare una mail da inviare per entrare in contatto con quest'ultima.

1.2.4 Shop (e-commerce)

Lo shop presenta gli articoli nel catalogo e ne fornisce le informazioni al click sulle schede della griglia. Sarà poi possibile aggiungere al carrello l'oggetto desiderato.

1.2.5 Cart (carrello)

Qui sarà possibile visualizzare gli articoli attualmente presenti nel carrello, modificarne le quantità e, eventualmente, procedere all'acquisto.

1.2.6 Purchase (acquisto)

Accessibile solo da `cart.php`, presenta il `form` da cui è possibile pagare e confermare l'ordine.

1.2.7 Orders (ordini in corso)

La pagina presenta gli ordini non ancora evasi e pertanto annullabili tramite drag&drop nell'apposito cestino.

2 Funzionalità

2.1 Registrazione, login e logout

Login e registrazione sono gestiti nella pagina `user.php`. Di default verrà mostrato il login, ma si potrà passare alla registrazione tramite un click.
In entrambi i casi verranno passati i dati inseriti dall'utente ad una chiamata `post` di Ajax, che a sua volta restituirà l'esito dell'operazione tramite il formato `json`.
Per la registrazione, lato server, viene verificata l'assenza di una mail già associata ad un altro account; in caso di successo la pagina verrà ricaricata, mostrando la sezione di login.
Il login invece, se avviene con successo, reindirizza l'utente all'home page del sito (`index.php`).
Per quanto riguarda il logout (`logout.php`), viene effettuata una richiesta Ajax di tipo `get`, senza l'invio di dati, e dopo la distruzione della sessione (`session_unset()` e `session_destroy()`) si viene reindirizzati alla pagina di login, `user.php`.

2.2 Gestione del contenuto generato dall'utente

L'utente ha la possibilità di aggiungere prodotti nel proprio carrello (da `shop.php`) e di modificarne le quantità (aumento/diminuzione/rimozione, in `cart.php`). Il contenuto del carrello è gestito tramite un'apposita tabella del database: `userOrder`.
Inoltre è possibile per l'utente annullare un ordine in corso (da `orders.php`).

3 Caratteristiche

3.1 Usabilità

Si è cercato di dare la massima priorità all'usabilità del sito, con l'utilizzo di uno stile minimale e con richiami alle interfacce a cui l'utente medio è abituato (griglie, box, barra di navigazione ecc…), attraverso l'osservazione dei siti più conosciuti. Si è scelto l'utilizzo di un solo colore (al di fuori dei colori "neutrali", nero, grigio e bianco) per evitare confusione e affaticamento visivo. Per tutte le funzioni più significative si è fornito un feedback visivo, attraverso un `flash` dallo sfondo colorato, che cambia in base al contesto.

3.2 Interazione/animazione

L'interazione con il sito risulta efficace e minimale, senza però rinunciare ad animazioni, utili come ulteriore spunto visivo. In generale si è cercato un compromesso tra efficienza e facilità di utilizzo, con l'ausilio di transizioni più complesse (come il drag&drop per l'eliminazione degli ordini) insieme a semplici `transition` (i pulsanti/bottoni) e `animation` (l'icona del carrello nella navbar, che cambia e viene animata se è presente almeno un oggetto nel carrello).

3.3 Sessioni

Le sessioni sono state gestite tramite `php`, sfruttando le funzioni per l'inizio (`session_start()`), l'aggiornamento (`session_regenerate_id()`), la rimozione (`session_unset()`) e la distruzione (`session_destroy()`) delle stesse. Le variabili di sessione utilizzate sono: l'email (`$_SESSION['email']`) e il nome dell'utente (`$_SESSION['name']`), usate per l'accesso alle tabelle e al benvenuto/riconoscimento dell'utente, rispettivamente.

3.4 Interrogazione del db

Le interrogazioni del database avvengono esclusivamente tramite l'interfaccia PDO del `php`.
In particolare le funzioni usate sono: `new PDO(...), setAttribute(...), getMessage(...), prepare(...), execute(...), fetch(...), fetchAll(...)`.

3.5 Validazione dei dati di input

La validazione dei dati in input è stata effettuata sia lato client (con `javascript`) che lato server (con `php`). In particolare vengono controllati gli input della registrazione, del login e della conferma del pagamento.
In caso di tentata registrazione con altra mail o di login con account inesistente, viene fornito un feedback sfruttando il risultato delle funzioni `php` utilizzate.

3.6 Sicurezza

Per tutte le query `SQL` è stata utilizzata la funzione `prepare()`, che provvede ad applicare la funzione `quote()` automaticamente. Il characterset utilizzato nel database è `utf8mb4`.
La gestione delle password **non** è in chiaro, ma criptata. Le password degli utenti sono state salvate tramite la funzione `password_hash()`. La verifica avviene invece attraverso la funzione `password_verify()`.

3.7 Presentazione

La presentazione del sito è chiara e predilige la fruizione dei contenuti ad un pubblico il più vasto possibile.

4 Front-End

4.1 Separazione presentazione/contenuto/comportamento

Si è scelto uno stile unobstrusive per separare presentazione, contenuto e comportamento. Ciò è stato possibile tramite l'uso di JQuery e Ajax. Si è quindi evitato il mescolamento del codice `HTML` con il `javascript` e il `CSS`. Sono inoltre assenti le stampe `php` nel codice `HTML`. Si è ridotto al minimo il codice `php` incluso all'interno dell'`HTML`. Per tutte le pagine il contenuto è stato gestito dinamicamente tramite `javascript`, minimizzando il refresh delle pagine.

4.2 Soluzioni cross-platform

Il sito è stato testato su Firefox e Brave, simulando la visualizzazione da PC, tablet e telefono, verificandone la responsiveness.
In caso di schermi di dimensioni ridotte viene cambiato il layout del sito, riducendo e modificando proporzionalmente gli elementi del DOM (navbar, footer, griglia in shop, presentazione home page ecc…) tramite l'uso della regola `CSS @media`.

4.3 Organizzazione file e cartelle di progetto

Il progetto è stato organizzato tramite il pattern MVC. Sotto la cartella `model` si trovano i file `php` che prevedono la connessione con i dati (sessioni e `SQL`); in `controller` si trovano invece i file `javascript` che si occupano della manipolazione e gestione dei dati; in `view` i file `html` e `css`, che presentano i dati. Sotto `model`, vi è un'ulteriore divisione in cartelle dei file `php`, le quali richiamano le funzionalità del sito. Infine troviamo una cartella `img` al cui interno si trovano i contenuti multimediali del sito.

4.4 Soluzioni html/css/javascript degne di nota

Come soluzione degna di nota si è scelto di analizzare lo shop, in particolare la gestione dei prodotti nella griglia. Al caricamento della pagina vengono fornite dal `php` solamente le informazioni essenziali del prodotto (`nome` e `id`); al click invece, viene fatta un'altra richiesta Ajax per ottenere le informazioni specifiche e complete del prodotto. Questa soluzione "lazy" alleggerisce sia lato server (evitando l'invio di dati non utilizzati) che lato client (caricamento più veloce), senza andare a scapito dell'usabilità (al click, il riscontro è immediato, non si percepisce attesa né caricamento).

5 Back-End e comunicazione Back-End/Front-End

5.1 Architettura generale e descrizione funzioni remote, funzioni di callback

Nota: per tutte le funzioni, la comunicazione server-client, cioè `php-javascript`, avviene tramite Ajax; per i dati da passare si è scelto il formato `json`; in caso di errore la funzione `error` di callback (fallimento database) e la `success` (nel caso in cui sia fallita l'operazione) è generica e restituisce nella console e/o nel `flash` un messaggio comprensibile dall'utente; in caso di successo, la `success` di fallback provvederà al feedback utente, alla costruzione/modifica del DOM o al reindirizzamento. Queste informazioni non verranno quindi esplicitate in seguito. Non saranno inoltre riportati per brevità "uso" e "output" quando ritenuti banali. Infine, viene specificato sotto ogni file o lista di file `php`, il file `javascript` che ne indica il controller o eccezionalmente la sua locazione/inclusione.

5.1.1 Funzioni comuni - cartella: `model/`

- `common.php`: reindirizza a `user.php` se l'utente non ha fatto il login

  File direttamente incluso in tutte le pagine della `view`, ad eccezione di `user.php`

- `updateBar.php`: restituisce il numero di oggetti nel carrello

  File controller associato: `common.js`

5.1.2 User - cartella: `model/user/`

- `login.php`: permette il login dell'utente
  - uso: `POST→login.php?email=prova@dom.it&password=test`
  - output: `{'status' : true} | {status : false; 'msg' : "errore..."}`
- `signup.php`: permette la registrazione
  - uso: `POST→signup.php?username=prova&email=prova@dom.it&password=test`
  - output: `{'status' : true; 'msg' : "successo..."} | {status : false; 'msg' : "errore..."}`

  File controller associato: `user.js`

- `logout.php`: funzione di logout

  File controller associato: `common.js`

- `util.php`: fa partire la sessione e fornisce la funzione di collegamento al database

  File direttamente incluso in tutte le pagine del `model`

5.1.3 Home page/index - cartella: `model/`

- `getHomePage.php`: estrae casualmente 3 `id` dalla tabella `artwork` del database
  - output: `{'name' : 'utente'; 'imgs' : array associativo con id e name dei prodotti}`

  File controller associato: `index.js`

5.1.4 Shop - cartella: `model/shop/`

- `addToCart.php`: aggiunge un prodotto al carrello
  - uso: `POST→addToCart.php?id=1`
  - output:`{'status' : true; 'msg' : "successo..."} | {status : false; 'msg' : "errore..."}`
- `filter.php`: restituisce gli `id` e `name` da `artwork` in base ai filtri inseriti nel `form`
  - uso: `GET→filter.php?price=&color=verde&category=ciotola`
  - output: `{array associativo con id e name dei prodotti}`
- `getArtWorks.php`: restituisce `id` e `name` da tutti i pezzi in `artwork`; fornisce tutti i colori e le categorie di `artwork`
  - output: `{'imgs' : array associativo con id e name dei prodotti; 'colors' : array con i colori; 'categories' : array con le categorie}`
- `getProductInfo.php`: restituisce `name, material, width, height, depth, isEdible, price` di un singolo oggetto
  - uso: `GET→addToCart.php?id=1`
  - output: `{array associativo con le caratteristiche del prodotto}`

  File controller associato: `shop.js`

5.1.5 Cart - cartella: `model/cart`

- `getCart.php`: restituisce gli oggetti nel carrello e il prezzo totale in euro
  - output: `{'products' : array associativo con nome, prezzo, id, quantità dei prodotti; 'total' : 100}`
- `removeProduct.php`: rimuove un oggetto dal carrello
  - uso: `GET→removeProduct.php?id=1`
  - output: `{"Prodotto rimosso dal carrello"}`
- `updateQuantity.php`: aumenta o diminuisce di 1 la `quantity` di un oggetto
  - uso: `GET→updateQuantity.php?id=1&op=1`
  - output: `{'msg' : "successo..."; 'total' : 10}`

  File controller associato: `cart.js`

- `util.php`: fornisce le funzioni per ottenere gli oggetti nel carrello e il loro prezzo complessivo

5.1.6 Orders - cartella: `model/orders/`

- `getOrders.php`: estrae dal database gli ordini non ancora evasi (`ongoing==TRUE`)
  - output: `{array associativo con, per ogni ordine: via, città e provincia di spedizione, totale in euro, data ordine}`
- `removeOrder.php`: annulla un ordine non ancora evaso (`ongoing==TRUE`)
  - uso: `GET→removeOrder.php?date=2021-01-21%2018:48:14`
  - output: `{"L'ordine è stato annullato!"}`

File controller associato: `orders.js`

5.1.7 Purchase - cartella: `model/purchase/`

- `confirmPurchase.php`: rimuove gli oggetti dal carrello e aggiunge l'ordine alla tabella `orders`
  - uso: `POST→confirmPurchase.php?street=Via%20Garibaldi%2040&prov=Torino&city=Lanzo`
  - output: `{'status' : true} | {'status' : false}`
- `getResume.php`: restituisce il totale in euro degli oggetti nel carrello
  - output: `{'user' : "utente"; 'total' : 100}`

File controller associato: `purchase.js`

5.2 Schema del db

Lo schema del database è semplice ma funzionale all'utilizzo del sito.
Dell'utente si memorizza solamente lo stretto necessario, senza *invadenti* richieste di dati aggiuntivi.
I pezzi di ceramica (`artwork`) hanno svariati dettagli tecnici e sono identificati da un id univoco (usato anche nella scelta della foto da mostrare, vd. `./img/artwork/`).
Nel carrello viene associata l'email dell'utente (la chiave primaria di `user`) all'id del pezzo, consentendo la presenza di una sola entry per la coppia `user-artwork` (la quantità, `quantity`, è tuttavia modificabile). Gli ordini sono invece memorizzati in `userOrder`. Osservando questa tabella si nota come si è scelto di richiedere i dati della spedizione ad ogni ordine e non, ad esempio, in fase di registrazione.

> Nota: si è deciso di non gestire il pagamento con carta di credito, nonostante nel `form` in `purchase.php` siano presenti i box che richiedono numero di carta e CVV, in quanto non ritenuto strettamente collegato al corso di Tecnologie Web.