Diario di lavoro

Luogo	SAMT
Data	20.02.2019

Lavori svolti

Questa mattina volevo iniziare a lavorare al carrello, ma prima ho dovuto mettere a posto alcuni problemi legati al cambiamento fatto alla fine della giornata di ieri al database come spiegato nella sezione che segue.
Dopo averlo fatto mi sono accorto che mancava la mostra dei dettagli del prodotto quando si clicca su di esso, allora ho optato per cercare un modal da potergli inserire, una volta fatto l'ho inserito nella pagina e ho continuato a lavorare al carrello senza implementarlo, perché non mi sembrava una priorità assoluta.
Una volta fatto ho cambiato la funzione "getProducts" che prende tutti i prodotti, facendo in modo che prenda i valori solo dei prodotti, perché in quel caso era inutile che prendeva tutti i dati anche dei venditori, i dati completi servono quando prende il prodotto singolo per mostrarlo o inserirlo nel carrello, quindi ho semplicemente cambiato la query, come di seguito.

```
SELECT * from prodotto
```

La select precedente non è stata eliminata ma spostata in una nuova funzione, "getProduct", che prende i dati dei prodotti singoli.

```
43    public function getProduct($name, $price, $quantity){
44        //Connetto al database
45        $conn = $this->connection->sqlConnection();
46
47        //Prendo i dati dell'utente in base alla mail
48        $sql = $conn->prepare("SELECT * from prodotto p
49                               inner join vende v on v.nome_prodotto = p.nome
50                               inner join negozio n on n.nome = v.nome_negozio and n.indirizzo = v.indirizzo_negozio and n.citta = v.citta_negozio
51                               WHERE p.nome LIKE :nome AND p.prezzo LIKE :prezzo AND p.quantita LIKE :quantita");
52        $sql->bindParam(':nome', $name, PDO::PARAM_STR);
53        $sql->bindParam(':prezzo', $price, PDO::PARAM_INT);
54        $sql->bindParam(':quantita', $quantity, PDO::PARAM_STR);
55
56        $dataArray = array();
57        //Eseguo la query
58        $sql->execute();
59
60        // Ciclo tutti i valori
61        while ($row = $sql->fetch()) {
62            array_push( &array: $dataArray, $row);
63        }
64        $conn = null;
65        return $dataArray;
66    }
```

Figura 1 model select prodotti singoli

La funzione riceve i valori che compongono la chiave dell'elemento, e fa la ricerca, prendendo tutti i valori, in base a quelli passati, dopodiché inserisce tutto in un array e lo ritorna.
Dopo aver modificato questo sono andato avanti con il file JavaScript, essendo che se non era completo quello non si può andare avanti.
Prima di tutto ho fatto in modo che alla creazione il link passi il valore del suo name, il link sarebbe quello da cliccare per inserire i valori dentro il carrello, e il valore passato è una concatenazione delle chiavi dell'oggetto in questo formato "nome.prezzo.quatità".
Una volta che questo valore arriva alla funzione viene decodificata da un'altra che ritorna i 3 valori in un array, come mostrato sotto.

```
295     function decode(value){
296         return value.split(".");
297     }
```

Figura 2 funzione di decodifica

Una volta decodificato la funzione richiamata va, grazie ad ajax, ad inserire il valore nel database.

```
303 function addToCart(codedKeys){
304     //Divido la chiave composta nei valori
305     var keys = decode(codedKeys);
306
307     xhttp.onreadystatechange = function () {
308         if (this.readyState === 4 && this.status === 200) {
309             console.log(xhttp.responseText);
310         }
311     }
312     xhttp.open( method: "POST", url: "/gestionevendita2018/customer/addToCart", async: true);
313     xhttp.setRequestHeader( name: "Content-Type", value: "application/x-www-form-urlencoded");
314     xhttp.send( body: "&name="+ keys[0] +"&price="+ keys[1] +"&quantity="+ keys[2]);
315
316
317 }
```

Figura 3 funzione JavaScript di aggiunta al carrello

La funzione richiama un controller, del cliente, che prende i valori che gli vengono passati, ovvero nome prezzo e quantità de prodotto (riga 314) e poi inseriscono i dati richiesti all'interno della tabella ponte "compra" che contiene tutti i dati dei prodotti acquistati, o semplicemente messi nel carrello, dell'utente.

```
22      public function addToCart()
23      {
24          //Se la sessione è aperta apro le pagine altrimeenti no
25          if(isset($_SESSION['customer'])) {
26
27              //Prendo la classe model
28              require_once 'application/models/product.php';
29              require_once 'application/models/buy.php';
30              $product = new ProductModel();
31              $buy = new BuyModel();
32
33              //Prendo le variabili passate dal POST
34              $name = isset($_POST["name"])? $_POST["name"] : null;
35              $price = isset($_POST["price"])? $_POST["price"] : null;
36              $quantity = isset($_POST["quantity"])? $_POST["quantity"] : null;
37              $prod = "";
38
39              //Se entrambi i campi non sono vuoti
40              if ($name != null && $price != null&& $quantity != null) {
41                  $buy->insertData($name, $_SESSION['customer']);
42              }
43          //Altrimenti
44          }else{
45
46          }
47      }
```

Figura 4 controller aggiunta al carrello.

Come si può notare, la funzione aggiunge al carrello solo se ha fatto il login un utente, in caso contrario non funziona, questo perché la parte in cui un utente a caso inserisce i valori nel carrello non l'ho ancora implementato.
La funzione model che viene richiamata si occupa semplicemente di inserire i dati all'interno del database nella tabella apposita.

```php
public function insertData($prodName, $custMail){

    //Connetto al database
    $conn = $this->connection->sqlConnection();

    //echo "Connected successfully";
    $sql = $conn->prepare("INSERT INTO compra (nome_prodotto, email_cliente, data)
    VALUES (:prodName, :custMail, now())");
    $sql->bindParam(':prodName', $prodName, PDO::PARAM_STR);
    $sql->bindParam(':custMail', $custMail, PDO::PARAM_STR);

    //Eseguo la query
    $sql->execute();

    //Se ci sono dei valori
    if($sql->rowCount() > 0){
        $conn = null;
        return true;
    } else {
        $conn = null;
        return false;
    }
}
```

Figura 5 model inserimento dati tabella "compra"

Come si può notare la parte del carrello non è ancora completata, mancano ancora la segnalazione degli elementi nel carrello, la mostra di essi, e tutta la parte dell'utente che non ha fatto il login. Un'altra cosa che devo implementare è il fatto che venga segnalato se il prodotto è effettivamente stato inserito o meno nel database.

Problemi riscontrati e soluzioni adottate

Il primo problema che ho avuto era con la mostra dei prodotti, perché ieri avevo eliminato tutto il contenuto della tabella ponte "vende" per modificare la chiave dei prodotti, ed essendo che la query per prendere i prodotti va riferimento anche a quella tabella per prendere i dati in modo completo allora ho dovuto riempirla nuovamente per poter mostrare i prodotti.

Punto della situazione rispetto alla pianificazione

Ho implementato parte del carrello.
Devo completare in modo ottimale la parte d'inserimento dei prodotti, ovvero inserendo anche quale negozio e venditore lo imposta.

Programma di massima per la prossima giornata di lavoro

Completare l'implementazione del carrello.