Diario di lavoro

Luogo	SAMT
Data	02.04.2019

Lavori svolti

Oggi ho iniziato a lavorare all'aggiunta, la diminuzione e l'eliminazione dei prodotti direttamente dal carrello.
Per prima cosa ho dovuto aggiungere dei simboli per farlo all'interno del carrello, quindi nella funzione che va ad inserirci i dati ho aggiunto 2 parti dove veniva inserita la x che elimina completamente il prodotto.

```
577         var aProd  = document.createElement( tagName: "a");
578         aProd.setAttribute( qualifiedName: "name",  value: obj['nome_prodotto'] +"."+ obj['prezzo_prodotto'] +"."+ obj['quantita_prodotto']
579         aProd.setAttribute( qualifiedName: "onclick",  value: "addToCart(this)");
580         var span = document.createElement( tagName: "span");
581         span.setAttribute( qualifiedName: "class",  value: "product-remove mr-5");
582         var i = document.createElement( tagName: "i");
583         i.setAttribute( qualifiedName: "class",  value: "fa fa-plus");
584         i.setAttribute( qualifiedName: "aria-hidden",  value: "ture");
585         span.appendChild(i);
586         aProd.appendChild(span);
587         div.appendChild(aProd);
588
589         //Inserisco il bottone di diminunzione
590         var aProd  = document.createElement( tagName: "a");
591         aProd.setAttribute( qualifiedName: "name",  value: obj['nome_prodotto'] +"."+ obj['prezzo_prodotto'] +"."+ obj['quantita_prodotto']
592         aProd.setAttribute( qualifiedName: "onclick",  value: "delOneFromCart(this)");
593         var span = document.createElement( tagName: "span");
594         span.setAttribute( qualifiedName: "class",  value: "product-remove mr-4");
595         var i = document.createElement( tagName: "i");
596         i.setAttribute( qualifiedName: "class",  value: "fa fa-minus");
597         i.setAttribute( qualifiedName: "aria-hidden",  value: "ture");
598         span.appendChild(i);
599         aProd.appendChild(span);
600         div.appendChild(aProd);
601
602         //Inserisco il bottone di eliminazione
603         var aProd  = document.createElement( tagName: "a");
604         aProd.setAttribute( qualifiedName: "name",  value: obj['nome_prodotto'] +"."+ obj['prezzo_prodotto'] +"."+ obj['quantita_prodotto']
605         aProd.setAttribute( qualifiedName: "onclick",  value: "delFromCart(this)");
606         var span = document.createElement( tagName: "span");
607         span.setAttribute( qualifiedName: "class",  value: "product-remove");
608         var i = document.createElement( tagName: "i");
609         i.setAttribute( qualifiedName: "class",  value: "fa fa-close");
610         i.setAttribute( qualifiedName: "aria-hidden",  value: "ture");
611         span.appendChild(i);
612         aProd.appendChild(span);
613         div.appendChild(aProd);
```

Figura 1 inserimento bottoni aggiunta diminuzione eliminazione prodotti

Tutti i contenitori delle immagini dei prodotti sono all'interno di un link che ha come nome le chiavi, concatenate, del prodotto in questione, e quando vengono cliccati aprono ognuno il metodo apposito per ciò che deve fare.
Il metodo per aggiungere il prodotto è già creato, ho solo dovuto fare quelli dell'eliminazione di un pezzo del prodotto e del prodotto completo.
Le funzioni implementate in JavaScript sono le seguenti.

```
415  function delOneFromCart(obj){
416
417      //Divido la chiave composta nei valori
418      var keys = decode(obj.name);
419
420      //Se il prodotto è disponibile
421      if(keys[2] > 0) {
422          xhttp.onreadystatechange = function () {
423              if (this.readyState === 4 && this.status === 200) {
424
425                  //Diminuisco di uno il valore delle quantità a tutti i nomi richiesti
426                  var name = encode(keys[0], keys[1], keys[2]);
427                  var x = document.getElementsByName(name);
428                  keys[2]++;
429                  for(var i = x.length - 1; i >= 0; i--) {
430                      x[i].name = encode(keys[0], keys[1], keys[2]);
431                  }
432
433                  //Modifico la variabile del numero di oggetti
434                  cartObjects--;
435                  totPrice -= parseInt(keys[1]);
436                  String(keys[1]);
437
438                  //Prendo i valori che vanno mostrati nel carrello
439                  getCartProduct(keys[0], keys[1], keys[2]);
440              }
441          }
442          xhttp.open( method: "POST",  url: "/gestionevendita2018/customer/modifyCart",  async: true);
443          xhttp.setRequestHeader( name: "Content-Type",  value: "application/x-www-form-urlencoded");
444          xhttp.send( body: "&name=" + keys[0] + "&price=" + keys[1] + "&quantity=" + keys[2] + "&action=delOne")
445      }
446  }
```

Figura 2 delOneFromCart elimina un pezzo del prodotto

Le funzioni richiamano tutte lo stesso controller, h è lo stesso della funzione di aggiunta dei prodotti nel carrello. La funzione dell'eliminazione completa del prodotto non è ancora completata, il prodotto iene cancellato, ma non viene mostrato direttamente l'eliminazione dal carrello, bisogna ricaricare la pagina.
Il controller della funzione ha subito a sua volta dei cambiamenti, per essere utilizzata da tutte e 3 la funzioni, prendendo l'ultimo elemento inviato con il post, che raffigura l'azione da svolgere.

```php
public function modifyCart()
{
    //Se la sessione è aperta apro le pagine altrimeenti no
    if(isset($_SESSION['customer'])) {

        //Prendo la classe model
        require_once 'application/models/buy.php';
        $buy = new BuyModel();

        //Prendo le variabili passate dal POST
        $name = isset($_POST["name"])? $_POST["name"] : null;
        $price = isset($_POST["price"])? $_POST["price"] : null;
        $quantity = isset($_POST["quantity"])? $_POST["quantity"] : null;
        $action = isset($_POST["action"])? $_POST["action"] : null;

        //Se entrambi i campi non sono vuoti
        if ($name != null && $price != null && $quantity != null && $action != null) {
            $buy->modifyData($name, $price, $quantity, $_SESSION['customer'], $action)
        }
    //Altrimenti
    }else{

    }
}
```

Figura 3 modifyCart modifica il carrello aggiungendo diminuendo o eliminando i prodotti

L'unica differenza della funzione, oltre al nome che cambia da "addToCart" a "modifyCart" è che prende il valore dell'azione e lo passa al model che lavora con il database.

```php
switch ($action) {
    case "add":
        $quantity = $this->addQuantity($buyCount);

        //Imposto la nuova quantità del prodotto
        require_once 'application/models/product.php';
        $product = new ProductModel();
        $product->setQuantity($prodName, $prodPrice, $prodQuantity, $prodQuantity - 1);
        $prodQuantity--;
        break;
    case "delOne":
        $quantity = $this->removeQuantity($buyCount);

        //Imposto la nuova quantità del prodotto
        require_once 'application/models/product.php';
        $product = new ProductModel();
        $product->setQuantity($prodName, $prodPrice, $prodQuantity, $prodQuantity + 1);
        $prodQuantity++;
        break;
    default:

        //Imposto la nuova quantità del prodotto
        require_once 'application/models/product.php';
        $product = new ProductModel();
        $newQuantity = isset($buyCount[0]['quantita_richiesta'])? $prodQuantity + $buyCount[0]['quantita_richiesta
            : $prodQuantity + $buyCount['quantita_richiesta'];
        $product->setQuantity($prodName, $prodPrice, $prodQuantity, $newQuantity);
}
```

Figura 4 switch controllo action

Nel model dopo il controllo del numero di oggetti c'è uno switch che controlla l'azione e a dipendenza di qual è svolge un'azione diversa, nei primi 2 casi vengono richiamate funzioni che modificano la quantità, mentre nell'ultimo caso, ovvero quello in cui viene eliminato completamente il prodotto, viene impostata una nuova quantità che è la vecchia più tutta la quantità richiesta.
Le funzioni richiamate esternamente sono le seguenti.

```php
    public function addQuantity($buyCount){

        //Setto la nuova quantità e modifico il campo nella tabella
        if(is_array($buyCount[0]))
            $quantity = $buyCount[0]['quantita_richiesta'] + 1;
        else
            $quantity = $buyCount['quantita_richiesta'] + 1;

        return $quantity;
    }

    public function removeQuantity($buyCount){

        //Setto la nuova quantità e modifico il campo nella tabella
        if(is_array($buyCount[0]))
            $quantity = $buyCount[0]['quantita_richiesta'] - 1;
        else
            $quantity = $buyCount['quantita_richiesta'] - 1;

        return $quantity;
    }
```

Figura 5 funzioni modifica quantità

La prima funzione si occupa di aumentare la quantità richiesta da inserire e la seconda di diminuirla.
Ho utilizzato questo metodo perché altrimenti avrei dovuto creare 2 funzioni con lo stesso codice per sql ma con la differenza minima mostrata nelle funzioni sopra, quindi è molto più efficiente fare in questo modo.
L'ultimo modifica che ho fatto è il controllo se la quantità richiesta viene portata a 0, perché in quel caso non bisogna semplicemente riscriverla ma eliminare direttamente l'istanza dal database, perché risulterebbe un dato inutile essendo inutilizzato.
Di conseguenza l'ho gestito come mostrato.

```php
        if($quantity == 0){
            $sql = $conn->prepare("DELETE FROM compra WHERE nome_prodotto LIKE :prodName
                AND prezzo_prodotto LIKE :prodPrice AND
                quantita_prodotto LIKE :prodQuantity AND email_cliente LIKE :custMail");
            $sql->bindParam(':prodName', $prodName, PDO::PARAM_STR);
            $sql->bindParam(':prodPrice', $prodPrice);
            $sql->bindParam(':prodQuantity', $prodQuantity, PDO::PARAM_INT);
            $sql->bindParam(':custMail', $custMail, PDO::PARAM_STR);
        }else {
            $sql = $conn->prepare("UPDATE compra SET quantita_richiesta = :quantity, data = now() WHERE nome_prodotto LIKE :prodNa
                AND prezzo_prodotto LIKE :prodPrice AND
                quantita_prodotto LIKE :prodQuantity AND email_cliente LIKE :custMail");
            $sql->bindParam(':quantity', $quantity, PDO::PARAM_INT);
            $sql->bindParam(':prodName', $prodName, PDO::PARAM_STR);
            $sql->bindParam(':prodPrice', $prodPrice);
            $sql->bindParam(':prodQuantity', $prodQuantity, PDO::PARAM_INT);
            $sql->bindParam(':custMail', $custMail, PDO::PARAM_STR);
        }
    }else{
```

Figura 6 scelta modifica o eliminazione prodotto dal carrello

Come si può vedere c'è un controllo che nel caso in cui la quantità sia pari a 0 viene eliminato altrimenti modificato il dato.

Problemi riscontrati e soluzioni adottate

Ho avuto alcuni problemi dopo l'inserimento dell'eliminazione dei dati dalla tabella "compra" perché ho dovuto modificare diverse parti di codice per riadattarla, perché a quel punto avevo anche cambiato il nome della funzione.
Un altro problema l'ho riscontrato con i bottoni, perché inizialmente avevo strutturato nel modo errato l'inserimento da parte di JavaScript di essi, ma come mostrato nell'immagine della sezione precedente è corretto. In più c'era il problema che quando premevo uno dei bottoni veniva modificato solo il nome di quello, e non andava bene essendo che nel nome dei bottoni c'è la chiave primaria del prodotto, quindi ho dovuto fare in modo che quando ne premevo uno si modificavano tutti inserendo la chiave con i dati corretti, come mostrato dalla figura sotto.

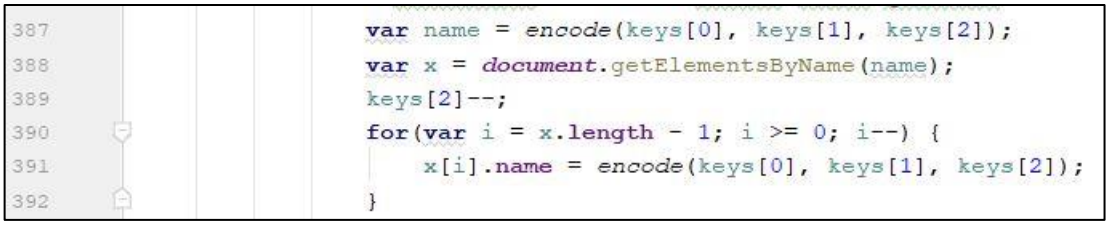

Figura 7 modifica chiavi prodotto

Il lavoro che viene eseguito è il seguente, prendo tutti i campi con il nome vecchio, modifico la quantità della chiave, dopodiché faccio passare tutti i dati dall'ultimo, essendo che dopo la modifica spariscono dall'array essendo cambiato il nome, e poi cambio il nome di quel tag. Ho avuto un problema con questa parte che mi ha reso diverso tempo perché inizialmente facevo passare i valori dal primo ma giustamente una volta cambiato il nome l'array diminuiva e quindi andava in conflitto, ma in questo modo funziona correttamente.

Punto della situazione rispetto alla pianificazione

Devo completare la modifica della quantità dei prodotti, non è ancora completa al 100%, dopodiché inizierò il checkout.
Devo ancora gestire il caso in cui 2 negozi facciano riferimento allo stesso prodotto e in uno dei 2 diminuisca la quantità.

Programma di massima per la prossima giornata di lavoro

Completare modifica quantità prodotti nel carrello, implementando il checkout.