Luca Antognarelli
Alessio Panizzieri

WEB SHOP

Indice

Descrizione della piattaforma

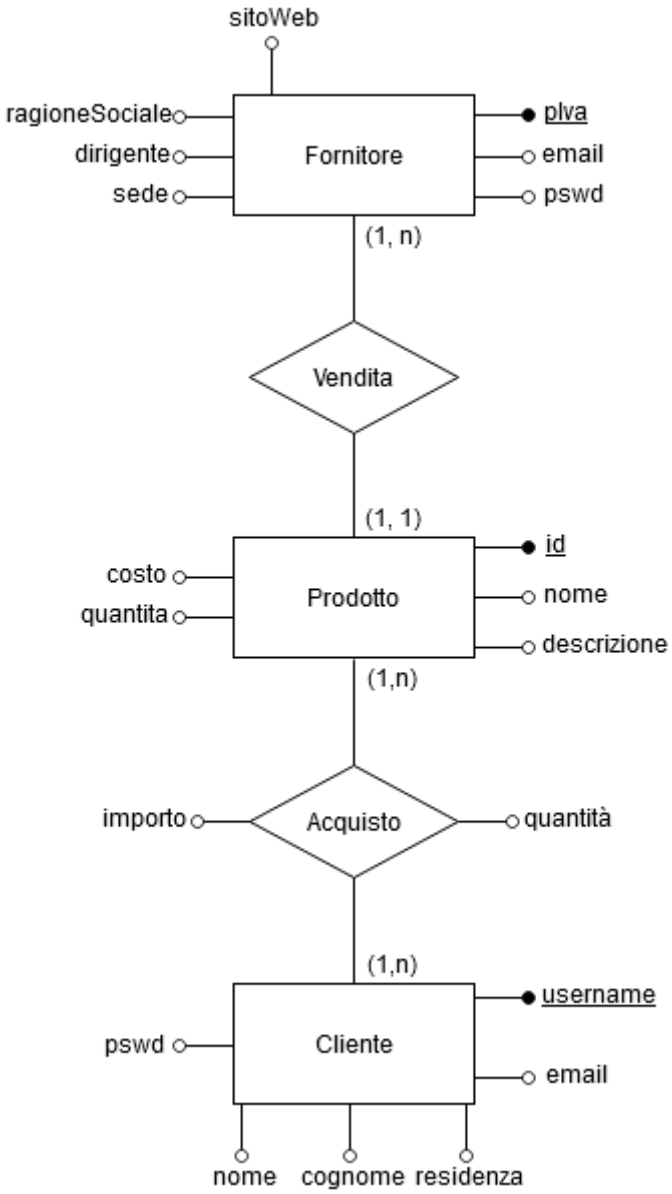
L'idea che c'è dietro alla piattaforma 'A&P web shop' è quella di realizzare una piattaforma di compravendita tra fornitori e clienti.
Ogni fornitore, identificato univocamente dalla propria Partita IVA, produce una certa quantità di prodotti e li mette in vendita sulla piattaforma ad un determinato prezzo.
Ogni cliente, identificato univocamente dal proprio username utilizzato al momento della registrazione, può consultare la piattaforma visualizzando tutti i prodotti disponibili e decidere di acquistarne uno o più in maniera semplice e veloce.
Clienti e fornitori per poter usufruire dei servizi offerti dalla piattaforma dovranno registrarsi mediante email (che non potrà essere la stessa per più clienti e/o fornitori) e password; i clienti dovranno inoltre scegliere un username univoco, indicare il proprio nome, cognome e la loro residenza; i fornitori dovranno invece indicare, oltre alla loro email e password, la ragione sociale, la sede legale, il sito web ed il nominativo del dirigente.
Grazie alla piattaforma sarà inoltre possibile, sia per i clienti che per i fornitori, tenere traccia dei propri acquisti (o vendite nel caso dei fornitori) consultando la pagina relativa al proprio profilo, nel quale saranno contenute quindi tutte le fatture.

Progettazione del database

Specifiche

Al fine della realizzazione, e successiva implementazione, della piattaforma risulta evidente la necessità di memorizzare le informazioni relative ai: clienti, fornitori e prodotti; ed individuarne le relazioni che li legano: clienti → acquisto ← prodotti e prodotti → vendita ← fornitori.
Risulta inoltre necessario memorizzare permanentemente le informazioni relative alla compravendita di un prodotto, quali: l'intestatario → chi ha effettuato l'acquisto, l'emittente→ chi ha venduto un determinato prodotto, il prodotto acquistato, in che quantità, l'importo totale relativo (costo del prodotto * quantità acquistata) e la data in cui è avvenuto l'atto.

Schema concettuale



Schema logico

fornitori(<u>plva</u>, email, pswd, ragioneSociale, sede, dirigente, sitoWeb)

prodotti(<u>id</u>, fkPIvaFornitore, nome, descrizione, costo, quantita)

clienti(<u>username</u>, email, pswd, nome, cognome, residenza)

fatture(<u>id</u>, fkIntestatario, fkIdProdotto, fkPIvaFornitore, emissione, quantitaProdotto, importo)

Funzionamento della piattaforma e query

La piattaforma 'A&P web shop' prevede al suo interno l'esistenza di due tipologie di profili, profondamente diversi tra loro: profilo cliente e profilo fornitore.
Profilo cliente: al momento della registrazione è richiesto l'inserimento: di un username che non sia già in uso, email, password ed alcuni dati personali quali: nome, cognome e residenza. Una volta effettuato il login si ha accesso alla home page della piattaforma nella quale sono presenti tutti i prodotti per cui è possibile effettuarne l'acquisto (quantità disponibile > 0) scegliendone la quantità, se dovesse essere inserita una quantità non valida verrà notificato il problema ed annullata la procedura d'acquisto. Nella pagina relativa al profilo cliente sono presenti le informazioni del cliente, con la possibilità di visualizzare tutte le fatture degli acquisti effettuati.
Profilo fornitore: al momento della registrazione è richiesto l'inserimento: della partita iva, email, password ed alcuni dati relativi al fornitore quali: ragione sociale, collocamento della sede legale, nominativo del dirigente e sito web. Una volta effettuato il login si ha accesso alla home page della piattaforma nella quale sono presenti tutti i prodotti messi in vendita, con la possibilità di modificarne: nome, descrizione, costo e quantità disponibile. Nella pagina relativa al profilo fornitore sono presenti le informazioni del fornitore, con la possibilità di visualizzare tutte le fatture delle vendite effettuate.

Data base e viste

Il data base della piattaforma è stato creato seguendo le decisioni prese nello schema logico, aggiungendo alle tabelle aventi riferimenti esterni politiche di reazione di tipo *cascade* per le azioni di update e delete.
Sono state inoltre create due viste: *vetrina_prodotti* e *magazzino*. La prima è stata creata con lo scopo di semplificare, ed evitare la ripetizione, della query mediante la quale vengono mostrati i prodotti disponibili ai clienti nella home page mostrando solamente i dati utili ed interesse.
La seconda è stata creata con lo scopo di ottimizzare e semplificare l'ottenimento dei dati sui prodotti di ciascun fornitore da visualizzare nella home page, così da permettere una gestione più facile.
Seguono le due query mediante le quali sono state create le viste:

```
/*Vista che forisce la lista dei prodotti disponibili associati ai rispettivi venditori*/
CREATE VIEW vetrina_prodotti AS (SELECT p.nome AS prodotto, p.descrizione AS descrizione, p.costo AS costo, p.quantita AS quantita, f.
ragioneSociale AS fornitore, p.id AS id_prodotto, f.pIva AS pIva_fornitore
                                 FROM fornitori f, prodotti p
                                 WHERE f.pIva = p.fkPIvaFornitore AND p.quantita > 0
                                 ORDER BY p.nome);

/*Vista riservata ai fornitori, forisce la lista dei prodotti associati alla partita IVA del fornitore*/
CREATE VIEW magazzini AS (SELECT f.pIva AS PIva, p.id AS id_prodotto, p.nome AS prodotto, p.descrizione AS descrizione, p.costo AS costo, p.
quantita AS quantita
                          FROM fornitori f, prodotti p
                          WHERE f.pIva = p.fkPIvaFornitore
                          ORDER BY f.pIva);
```

Piattaforma

Registrazione

La registrazione avviene separatamente per fornitori e clienti, mediante form dedicate, con la possibilità di scelta all'apertura della pagina.
A seguito dell'inserimento dei dati utili alla registrazione da parte di un cliente o un fornitore, questi ultimi vengono inviati al server con una particolare accortezza per la password che subisce inizialmente un'azione di crittografia basata sull'hashing, nello specifico SHA-256, e successivamente prima dell'inserimento dei dati nel db, al fine di evitare iniezioni di codice malevole, viene utilizzata una tecnica di query-prepare. Viene inoltre controllato che l'email inserita non sia già presente nella piattaforma, oltre che l'username nel caso di un cliente e della partita IVA nel caso di un fornitore.
Seguono due delle query utilizzate:

Query controllo unicità email:

```
$queryCheckEmail = "SELECT *
                    FROM clienti, fornitori
                    WHERE clienti.email = '".$emailIn."' OR fornitori.email = '".$emailIn."'";
```

In cui $emailIn rappresenta l'email inserita al momento della registrazione.

Query-prepare ed esecuzione query d'inserimento dati di un fornitore nel db:

```
$queryInserisciFornitore = $GLOBALS['connect']->prepare("insert into fornitori values('".$nuovoFornitore['pIva']."', '".$nuovoFornitore
['email']."', ?, '".$nuovoFornitore['ragioneSociale']."', '".$nuovoFornitore['dirigente']."', '".$nuovoFornitore['sede']."', '".
$nuovoFornitore['sitoWeb']."')");

$password = mysqli_real_escape_string($GLOBALS['connect'], $nuovoFornitore['password']);
$queryInserisciFornitore->bind_param('s', $password);

if($queryInserisciFornitore->execute() == false)
{
    echo("error_4");
    exit;
}
```

In cui: $GLOBALS['connect'] rappresenta la stringa di connessione al db, $nuovoFornitore['dato'] rappresenta un array associativo contenente i dati del fornitore da registrare nella piattaforma.
Il carattere '?' nella query serve per evitare azioni d'injection, viene ricostruita mediante le funzioni *myqli_real_escape_string* e nello specifico *bind_param* assegna il valore inserito per il quale si vuole evitare un'azione d'injection (password crittografata), successivamente alla ricostruzione della query viene eseguita.

Login

Avviene sia per i clienti che per i fornitori mediante l'email e la password, avendo entrambi i dati memorizzati in due tabelle differenti mediante una funzione controllo l'appartenenza dei dati inseriti ai clienti o ai fornitori, nel caso non vi fosse una corrispondenza l'utente viene invitato ad effettuare la registrazione di un nuovo account mediante il reindirizzamento presso la pagina di registrazione.
Come per la registrazione, al fine di garantire la sicurezza della piattaforma, la password subisce una crittografia basata sull'hashing, nello specifico SHA-256 e successivamente, precedentemente l'esecuzione della query di controllo dei dati viene eseguita un'azione di query-prepare per prevenire tentativi d'injection.
Segue la query di login:

```php
$query = $GLOBALS['connect']->prepare("
        SELECT *
        FROM ".$tabellaCheck."
        WHERE email = '".$emailIn."' AND pswd = ?
    ");

$password = mysqli_real_escape_string($GLOBALS['connect'], $passwordIn);
$query->bind_param('s', $password);
$query->execute();
$result = $query->get_result();
```

In cui: $tabellaCheck assume il valore ‘clienti’ se si deve effettuare il login di un cliente o ‘fornitori’ se si deve effettuare il login di un fornitore, $emailIn e $passwordIn sono l’email e la password (cifrata) inserite.
A seconda del tipo di account loggato viene creata una sessione contenente valori utili alla gestione delle azioni di acquisto (se clienti l’username) e modifica dei prodotti (se fornitori partita iva), oltre che il tipo di utente loggato, ‘clienti’ o ‘fornitori’, e l’email, con la quale è possibile controllare se l’utente ha già effettuato il login nel caso in cui chiuda per errore la scheda non dovrà effettuare nuovamente il login; mantenere dati utili in sessione permette inoltre di gestire il caricamento dinamico delle pagine: home e profilo.

Home page

Il caricamento della home page della piattaforma avviene dinamicamente attraverso uno script che utilizza la tecnologia AJAX, Asynchronous JavaScript and XML, basata su uno scambio di dati in modalità asincrona tra client e server che permette l’aggiornamento dinamico di porzioni di pagina, non interferendo quindi con eventuali altre azioni svolte dall’utente sulla pagina.
Nel caso della home page, a seconda del tipo di account loggato, indicato dalla variabile in sessione, viene gestito il caricamento dinamico della pagina:
clienti: viene visualizzato l’elenco di tutti i prodotti in vendita con le rispettive informazioni, accanto ad ognuno è presente un bottone che avvia la procedura d’acquisto, richiedendo la quantità del prodotto d’acquistare e confermando l’acquisto, se la quantità inserita è disponibile, viene notificato l’avvenuto acquisto e generata la corrispettiva fattura, che comporta la diminuzione della quantità disponibile del prodotto in accordo alla quantità acquistata.
fornitori: viene visualizzato l’elenco dei prodotti in vendita appartenenti al fornitore che ha effettuato il login, accanto ad ognuno è presente un bottone che avvia la procedura di modifica. Quest’ultima si svolge inserendo in sessione l’id del prodotto, potendo così gestirne la modifica, selezionando attraverso l’apposito form il campo da modificare ed assegnandoli un nuovo valore, potendo così comporre e generare la query di modifica.
Seguono una porzione dello script basato su AJAX che gestisce il caricamento dinamico della pagina e query di modifica:

```
var xmlhttp = new XMLHttpRequest(); //Variabile gestione interrogazioni client-server
function visualizzaMagazzino()
{
    xmlhttp.onreadystatechange =
    function()
    {
        if (xmlhttp.readyState == 4 && xmlhttp.status == 200)
        {
            if(xmlhttp.responseText == "error_0")
                alert("Errore nella ricerca dell'azione");

            if(xmlhttp.responseText == "error_6")
                alert("Errore nella visualizzazione del magazzino");

            else if(xmlhttp.responseText != null)
            {
                try
                {
                    var resultRequest = JSON.parse(xmlhttp.responseText);

                    var stringHtml = "<h3>Magazzino</h3> \
                                      <button onclick=\"window.location.href='inserisci.html'\">Aggiungi un prodotto</button> \
                                      <table> \
                                        <tr><th>ID</th><th>Prodotto</th><th>Descrizione</th><th>Costo</th><th>Quantita'</th></tr>";

                    resultRequest.forEach(element =>
                    {
                        stringHtml += "<tr><td>" + element.id_prodotto + "</td><td>" + element.prodotto + "</td><td>" + element.descrizione + "</
                        td><td>" + element.costo + "</td><td>" + element.quantita + "</td><td><button type='button' onclick='preparaModifica(\""
                        + element.id_prodotto + "\")'>Modifica</button>"+"</td></tr>";
                    });

                    stringHtml += "</table><br>";

                    //Modifica prodotto
                    stringHtml += "<div id=\"popupFrame\" style=\"display:none\"> \
                                    <form action=\"../library/controller.php\" method=\"post\"> \
                                        Campo da modificare:<input list=\"modCampo\" name=\"modCampo\" autocomplete=\"off\" /><datalist
                                        id=\"modCamp\"> \
                                            <option>Nome</option> \
                                            <option>Descrizione</option> \
                                            <option>Costo</option> \
                                            <option>Quantita</option> </datalist><br> \
                                        Nuovo valore:<input type=\"text\" id=\"modValore\" name=\"modValore\" required=\"required\"><br> \
                                        <input type=\"submit\" name=\"azione\" value=\"modifica_prodotto\" onclick=\"aggiornaPagina()\"> \
                                    </form> \
                                </div><br>";

                    document.getElementById('contenuto').innerHTML = stringHtml;

                }

                catch
                {
                    alert("Errore nell'esecuzione del parse --> visualizzaMagazzino()");
                }
            }
        }

    }

    xmlhttp.open("GET", "../library/controller.php?azione=home_page", true);
    xmlhttp.send();
}
```

In cui la variabile xmlhttp è un oggetto XMLHttpdRequest che permette di gestire la comunicazione asincrona tra client e server. Attraverso *open* setto in GET l'azione da eseguire al server, così da avviare la procedura che mi porterà a comporre dinamicamente la pagina, attraverso *send* richiamo invio in GET la stringa con l'azione da eseguire al server e successivamente attraverso *readyState* e *status* controllo che la connessione al server sia andata a buon fine. Lato server in php compongo la risposta in formato JSON:

```
//Array che conterrà i risultati della query
$magazzino = Array();

//Faccio un ciclo in cui scorro tutte le righe ottenute come risultato e le inserisco in un array
while($row = $result->fetch_assoc())
    array_push($magazzino, $row);

//Converto in json l'array risultato ottenuto al fine di poterlo gestire in ajax
echo json_encode($magazzino);
```

e lato client attraverso *JSON.parse* decodifico la risposta al fine di ottenere i dati grazie ai quali nel ciclo successivo compongo dinamicamente la pagina.

Acquisto

L'effettuazione di un acquisto è divisa in due parti: inizialmente il cliente sceglie il prodotto da acquistare, successivamente indicando la quantità conferma l'acquisto.
Al momento della scelta del prodotto vengono inseriti in sessione l'id del prodotto selezionato ed il suo costo, utili alla successiva creazione della fattura e calcolo del costo complessivo.

```
$_SESSION['idProdotto'] = $idProdotto;
$_SESSION['costo'] = $costo;
```

Quando l'utente indica la quantità che desidera acquistare e conferma l'acquisto viene avviata una procedura che controlla inizialmente se la quantità inserita è disponibile, se così non fosse viene notificato il problema al cliente, altrimenti viene calcolato il costo complessivo (costo * quantità) e generata la fattura relativa all'atto di compravendita avvenuto. Ad acquisto avvenuto viene aggiornata la quantità residua disponibile del prodotto coinvolto.
Seguono le query di generazione della fattura ed aggiornamento della quantità di prodotto disponibile:

```
//Genero la fattura a seguito dell'acquisto avvenuto
$queryFattura = "INSERT INTO fatture(fkIntestatario, fkIdProdotto, fkPIvaFornitore, emissione, quantitaProdotto, importo)
                 VALUES('".$_SESSION['username']."', '".$_SESSION['idProdotto']."', '".$pIvaFornitore."', '".date("Y/m/d").
                 "', '".$quantita."', '".$importo."')";

//Aggiorno la quantità di prodotto disponibile a seguito dell'acquisto
$queryUpdateQuantita = "UPDATE prodotti
                        SET quantita = quantita - ".$quantita."
                        WHERE id = '".$_SESSION['idProdotto']."'";
```

In cui $importo è dato da:

```
$importo = $_SESSION['costo'] * $quantita;
```

Con la seguente query è avvenuto precedentemente il controllo della quantità richiesta che deve essere <= alla quantità disponibile, così da permettere l'avvio di tutta la procedura d'acquisto:

```
//Verifico che sia possibile effettuare l'acquisto
$queryAcquisto = "SELECT *
                  FROM vetrina_prodotti v
                  WHERE v.id_prodotto = '".$_SESSION['idProdotto']."'  AND v.quantita >= '".$quantita."'";
```

Modifica

La modifica di un prodotto da parte di un fornitore è divisa in due parti: il fornitore sceglie il prodotto da modificare, successivamente mediante l'apposito form sceglie il campo da modificare, gli assegna un nuovo valore e conferma la modifica. E' possibile modificare qualsiasi campo eccetto l'id.
Con la prima fase viene inserito in sessione l'id del prodotto da modificare:

```
$_SESSION['idProdotto'] = $idProd;
```

Con la seconda fase vengono inviati al server il campo ed il nuovo valore da assegnarlo, così da poter comporre una query dinamica per l'aggiornamento del dato del prodotto desiderato.
Segue la query mediante la quale avviene l'aggiornamento del valore:

```
$queryModifica = "UPDATE prodotti
                  SET ".$campo." = '".$valore."'
                  WHERE id = ".$_SESSION['idProdotto']."";
```

Inserimento

L'inserimento di un nuovo prodotto da parte di un fornitore avviene mediante una pagina ed un form dedicati. Viene richiesta la compilazione di tutti i campi previsti ed attraverso il click sul bottone viene inserito il nuovo prodotto all'interno del db.
Segue la query d'inserimento:

```
$queryInserimento = "INSERT INTO prodotti(fkPIvaFornitore, nome, descrizione, costo, quantita) VALUES('".$_SESSION['pIva']."', '".
$nome."', '".$descrizione."', '".$costo."', '".$quantita."')";
```

Profilo

La pagina del profilo viene caricata dinamicamente a seconda della tipologia di account loggato mediante uno script in AJAX. Vengono visualizzate le informazioni relative all'account, con la possibilità inoltre di visualizzare le fatture associate all'account mediante una funzione AJAX che ne permette l'ottenimento ed inserimento all'interno della pagina.
Segue la query d'ottenimento delle informazioni di un cliente e la composizione dei dati in formato JSON da inviare al client:

```
$queryInfoCliente = "SELECT c.username, c.email, c.nome, c.cognome, c.residenza
                     FROM clienti c
                     WHERE c.username = '".$_SESSION['username']."' AND c.email = '".$_SESSION['email']."'";
```

```
//Array che conterrà i risultati della query
$infoCliente = Array();

//Faccio un ciclo in cui scorro tutte le righe ottenute come risultato e le inserisco in un array
while($row = $result->fetch_assoc())
    array_push($infoCliente, $row);

//Converto in json l'array risultato ottenuto al fine di poterlo gestire in ajax
echo json_encode($infoCliente);
```

Segue una frazione della funzione AJAX che si occupa del composizione delle informazioni da visualizzare nella pagina del profilo:

```
var accountInfo = JSON.parse(xmlhttp.responseText);
var stringHtml = "<h3>Profilo</h3><table>";
accountInfo.forEach(element =>
{
    stringHtml += "<tr><td>Username</td><td>" + element.username + "</td></tr> \
                  <tr><td>Email</td><td>" + element.email + "</td></tr> \
                  <tr><td>Nome</td><td>" + element.nome + "</td></tr> \
                  <tr><td>Cognome</td><td>" + element.cognome + "</td></tr> \
                  <tr><td>Residenza</td><td>" + element.residenza + "</td></tr> ";
});

stringHtml += "</table><br>";
stringHtml += "<button onclick=\"visualizzaFatture()\">Visualizza fatture</button>";

document.getElementById('contenuto').innerHTML = stringHtml;
```

Errori

Gli errori vengono gestiti mediante uno specifico codice identificativo, la cui sintassi è la seguente: `error_[numero | lettera]`.
Ogni funzione lato server in caso di errore o situazione inaspettata comunica un codice d'errore, questo può notificare all'utente un errato utilizzo della piattaforma o allo sviluppatore un mal funzionamento.
Segue l'elenco degli errori codificati per lo sviluppo della piattaforma:

```
error_0 -> errore nel controller, parametro non valido
error_1 -> errore nella fase della registrazione, email già presente nel DB
error_2 -> errore nella fase della registrazione, tipo di account passato non valido
error_3 -> errore nella fase delle registrazione, nell'inserimento del cliente
error_4 -> errore nella fase delle registrazione, nell'inserimento del fornitore
error_5 -> errore nella query sulla vista vetrina_prodotti, non c'è la vista o non ci sono prodotti
error_6 -> errore nella query sulla vista magazzini, non c'è la vista o non ci sono prodotti o non ci sono fornitori
error_7 -> errore nella query dell'acquisto, non è possibile effettuatlo o non c'è un prodotto con quel ID o
           non ce n'è una quantità sufficiente
error_8 -> errore nella query d'inserimento, valori non validi
error_9 -> errore nella query per info cliente, valori errati o sessione inesistente
error_10 -> errore nella query per info fornitore, valori errati o sessione inesistente
error_11 -> errore nella query delle fatture, valori errati o sessione inesistente

error_l -> errore nella visualizzazioni di informaizoni, l'utente deve prima effettuare il login
```