Progetto Database

Libreria Online

Studente:

Thilina Perera P.A [514553]

Il progetto consiste nella creazione di un e-commerce per la vendita di prodotti editoriali di nome Libriperera.

I clienti hanno la possibilità di salvare e effettuare ordini che verranno spediti all’indirizzo di spedizione specificato. Il cliente effettua il pagamento tramite carta di credito o account paypal.

Il sito web è accessibile ai clienti che si registrano e accedono tramite un nome utente e una password. Il cliente ha la possibilità di cercare i prodotti da aggiungere al carrello da un catalogo degli articoli, completo di informazioni come descrizione, prezzo e disponibilità del prodotto. Una volta aggiunti al carrello, gli articoli rimarranno al suo interno anche dopo la disconnessione del cliente, e verranno rimossi/variati solo nel caso in cui:

- l’ordine viene effettuato.

-gli articoli vengono rimossi manualmente dall’utente.

-articoli rimossi per una variazione dello stato degli articoli nel magazzino (articoli non più disponibili o se è disponibile una quantità inferiore).

Il cliente è un utente, sono presenti altri due tipi di utente, l’Admin e il Super-admin.

- L'admin oltre ad essere un cliente, può inserire, rimuovere e modificare tutte le informazioni del catalogo degli articoli e ha accesso alla pagina per la registrazione di nuovi admin.

- Il super-admin è un admin che può assegnare e revocare i permessi di amministratore e gestire lo stato dei pagamenti/ordini dei clienti.

**Glossario**:

|  |  |  |  |
| --- | --- | --- | --- |
| ***Temine*** | ***Descrizione*** | ***Sinonimi*** | ***Collegamenti*** |
| **Cliente** | User iscritto al sito, ha la possibilità di acquistare articoli e quindi effettuare ordini | User | Indirizzo  Carta di credito  Carrello  Piattaforma cliente |
| **Admin** | User iscritto al sito, Gestisce gli articoli nel catalogo | Gestore sito | Piattaforma admin  articolo |
| **Super-admin** | Utente che può accedere al database, Gestisce gli articoli nel catalogo, gli ordini dei clienti, lo stato dei pagamenti e ordini. | Gestore sito | Ordine  pagamento |
| **Carta di credito/paypal** | Un metodo di pagamento utilizzato dai clienti per effettuare ordini | Metodo di pagamento | Cliente |
| Articolo | Prodotto editoriale disponibile per l’acquisto. | Lista di articoli | Carrello  Ordine |
| Ordine | Lista di Articoli acquistati da un cliente pronto per essere spedito. | Acquisto | Indirizzo  Carta di credito |
| Indirizzo | Insieme di informazioni necessarie per la spedizione ad un domicilio | domicilio | Cliente |
| Carrello | Lista di articoli salvati dall’utente per un eventuale acquisto | Articoli desiderati | Cliente  Articolo |
| Piattaforma cliente | Sito web lato per i clienti,  dà la possibilità di acquistare articoli e quindi effettuare ordini | Sito web | cliente |
| Piattaforma admin | Sito web lato per i admin,  dà la possibilità di gestire il catalogo degli articoli | Interfaccia di gestione | admin |

**Ristrutturazione schema ER**:

Per aumentare l’efficienza del database, sono state eliminate alcune informazioni ridondanti. Le tre generalizzazioni presenti sono state risolte combinando la tabella genitore delle generalizzazioni con quelle figlie.

In particolare ho trovato ridondante la generalizzazione di Persona in 3 utenti differenti(cliente, admin, superadmin), nella ristrutturazione ho scelto di avere una tabella “Utente” che comprende tutti i tipi di utenti che vengono differenziati dalla colonna “ruolo”.

**Lista delle operazioni:**

**-login**: accesso dell’utente tramite username e password.

**-logout:** disconessione dall’account dell’user.

**-add to cart:** aggiunta di un articolo al carrello dopo aver specificato la quantità desiderata.

**-remove:** rimozione di un articolo dal carrello del cliente.

**-ricerca articolo**: ricerca di articoli tramite una parola chiave digitata su barra di ricerca.

**-filtro catalogo:** ricerca degli articoli filtrati per tipologia di prodotto editoriale.

**-i tuoi articoli:** lista degli ordini del singolo utente con informazioni relative all’ordine.

**-acquisto tramite paypal:** pagamento effettuato dal cliente tramite servizio paypal.

**-acquisto tramite carta di credito:** pagamento effettuato dal cliente tramite carta di credito.

operazioni riservati agli Admin

**-edit articolo:** modificare informazioni relative ad un articolo.

**-delete articolo:** rimuovere un articolo da catalogo.

**-add articolo:** aggiungere un nuovo articolo dal catalogo.

**-mostra articolo:** mostra informazioni legate ad un singolo articolo

**-registrare nuovo Admin:** registrare sul sito un nuovo admin.

**Traduzione verso il modello relazionale**:

utente (**IDcliente**, Nome, Cognome, Mail, Password)

articolo (**ISBN**, Nome, Autore, Editore, Descrizione, Prezzo, Quantità, Tipologia, image1, image2, image3, Data\_inserimento)

carrello (**id\_utente\*,** **ISBN\*** ,articolo, quantità)

indirizzo(**id\_indirizzo,** **id\_cliente\***, ship\_address1, admin\_area1, admin\_area2, countrycode, postalcode)

ordine(**id\_ordine**, **id\_cliente\***, lista\_prodotti, Data\_inserimento, costo, id\_pagamento, status\_ordine, **id\_indirizzo**\*)

pagamento (**id\_pagamento**, **id\_transazione\***, **id\_cliente**\*, payment\_status, payer\_email, costo)

**Trigger database**:

Per questo progetto ho pensato di rendere il carrello dei clienti gestito dal database piuttosto di renderlo utilizzabile solo tramite sessione cliente, perciò una volta che il cliente scegli di effettuare il logout dal sito, gli articoli salvati nel carrello rimarranno salvati nel database in modo tale che l’utente possa effettuare l’ordine in un altro momento senza dover rimettere gli articoli nel carrello.

Ho dovuto effettuare più controlli per mantenere questo approccio e in particolare volevo che ogni cliente avesse un carrello con articoli e quantità degli articoli coerenti con la disponibilità del negozio

Per fare ciò oltre a controlli server side con PHP, dal punto di vista del Database è presente un Trigger nominato update\_quantity, che mostro qui sotto

CREATE TRIGGER `update\_quantity` AFTER UPDATE ON `articolo`

FOR EACH ROW BEGIN

UPDATE carrello

SET quantità =

CASE

WHEN NEW.Quantità < quantità THEN NEW.Quantità

ELSE quantità

END

WHERE ISBN = NEW.ISBN;

END

Un cliente non potrà mai inserire una quantità maggiore alla disponibilità. Ma un eventualità molto probabile è: che l’ articolo è stato inserito in precedenza nel carrello per essere salvato senza effettuare l’ordine, successivamente il numero degli articolo disponibile diminuisce per via della modifica da parte di un admin o semplicemente per l’acquisto di altri utenti, la quantità verrà modificata e abbassata alla quantità disponibile o tolta se non è più disponibile proprio grazie a update\_quantity.

Un'altra eventualità è quella dell’eliminazione da parte dell’admin di un articolo dal database (operazione meno frequente visto che è preferibile tenere un articolo nel database per un eventuale restock successivo)

mentre l’articolo è salvato nel carrello. Per risolvere ciò viene fatto un eliminazione a cascata da tutti i carrelli alla cancellazione dell’articolo dal catalogo.