Relazione Applicazioni Web

Santià Alice 20013051, Viviani Andrea 20010912

Per l’avvio:

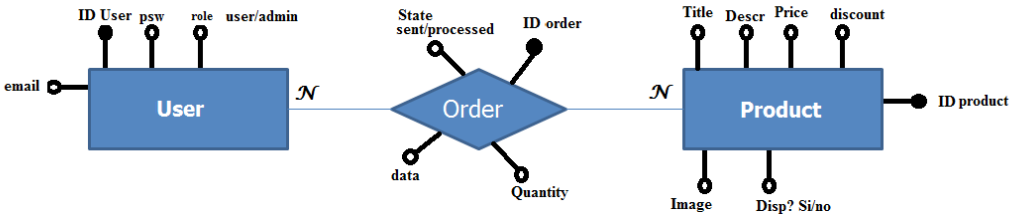
- Modificare la stringa di connessione al database in appsettings.json e aggiungere le proprie credenziali se queste fossero necessarie per accedere al database.
- Abbiamo allegato uno script SQL (ddl) per creare il database e uno script SQL (dml) per popolarlo, eseguirli in sequenza. Attenzione: prima di eseguire lo script dml modificare il path che compare dopo DECLARE @ROOT VARCHAR(500) nella prima riga, inserendo il proprio path alla cartella in cui si è salvato il progetto.
- Aprire il progetto in VisualStudio ed eseguire.
- Abbiamo già inserito alcuni prodotti, ordini e tre utenti:
    - alice@gmail.com user, password: ciao
    - andre@gmail.com user, password: ciao
    - john@gmail.com admin, password: ciao
- Gli utenti user hanno già eseguito degli ordini.

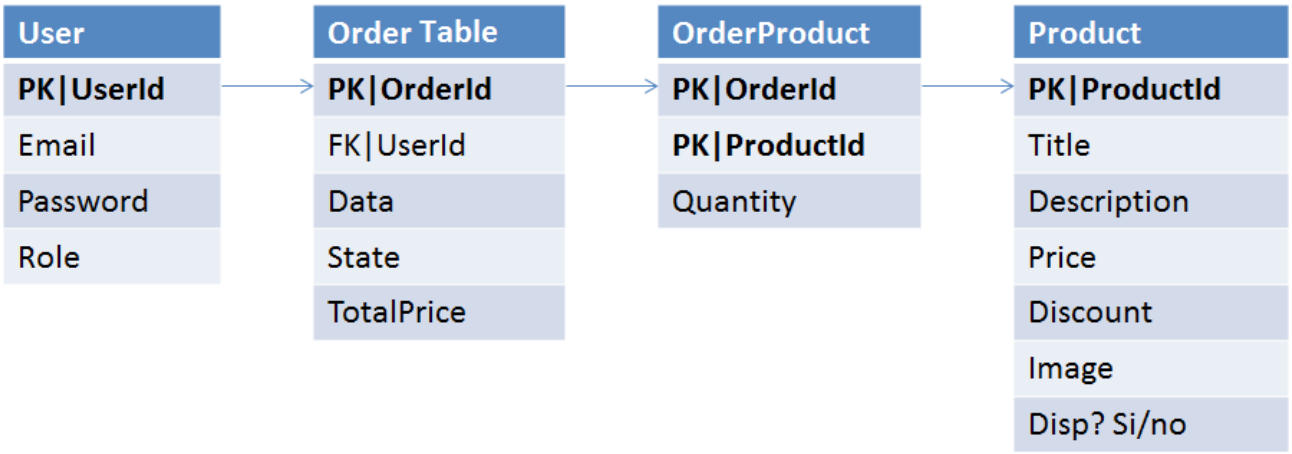
Scelte implementative

- Abbiamo usato dei bean per poter gestire l’accesso degli utenti alle loro rispettive aree riservate e per memorizzare il carrello in sessione.
- Gli utenti di tipo admin non possono fare acquisti quindi non vedono il carrello e i bottoni per aggiungervi i prodotti.
- Per aggiungere prodotti al carrello, modificare le informazioni relative ai prodotti, modificare il ruolo di un utente, modificare lo stato di un ordine abbiamo usato dei form.
- Per fare operazioni di ricerca o di filtraggio si usano dei form.
- In generale, abbiamo usato i form quando avevamo necessità di comunicare col database.
- Abbiamo fatto un filtro generico e dei filtri specifici per filtrare utenti e ordini.
- Il carrello è un bean memorizzato in sessione, quindi l’acquisto deve per forza essere terminato in sessione.

Descrizione modello relazionale

Seguono lo schema ER e UML del database che abbiamo creato.





Descrizione modello ad oggetti

Il sito è stato implementato col pattern MVC:

- Models: la cartella contiene i modelli ovvero classi che mappano le entità del database in oggetti, abbiamo una classe per ogni entità.
- Controllers: la cartella contiene i controllers, classi che hanno metodi per manipolare i modelli e restituire il risultato alle viste.
    - CartController: gestisce aggiunta/rimozione di prodotti al carrello, svuotamento del carrello, modifica della quantità di un prodotto presente nel carrello.
    - CrudController: gestisce le operazioni crud di base, i controller di User, OrderTable e Product usano i metodi in CrudController per eseguire tali operazioni.
    - HomeAdminController: non fa nulla, reindirizza soltanto alla vista per la pagina privata degli admin.
    - HomeController: mostra la homepage con la top 10 dei prodotti più venduti nell'ultimo mese.
    - OrderController: gestisce la creazione di nuovi ordini per gli utenti di tipo user e la loro modifica per gli utenti di tipo admin, permette di visualizzare gli ordini fatti.
    - ProductController: visualizza i prodotti nelle pagine di catalogo, ricerca e dettaglio del prodotto, gestisce la modifica del prodotto per gli utenti di tipo admin.

        - SmallPricesController: mostra una pagina con l’elenco di tutti i prodotti con uno sconto > 0.
        - UserController: permette la registrazione e il login per tutti gli utenti, visualizza una lista di utenti e permette di modificarne il ruolo se si è admin.
- Views: la cartella contiene viste, ovvero pagine statiche che mostrano il risultato della manipolazione dei modelli.
- Data: contiene i bean per User, Order, Cart e il contesto.
- Scr: contiene i filtri.
- I loghi e le icone usate nel sito sono in wwwroot/Image, mentre le immagini relative ai prodotti nel database sono in una cartella chiamata ImmaginiProdotti.