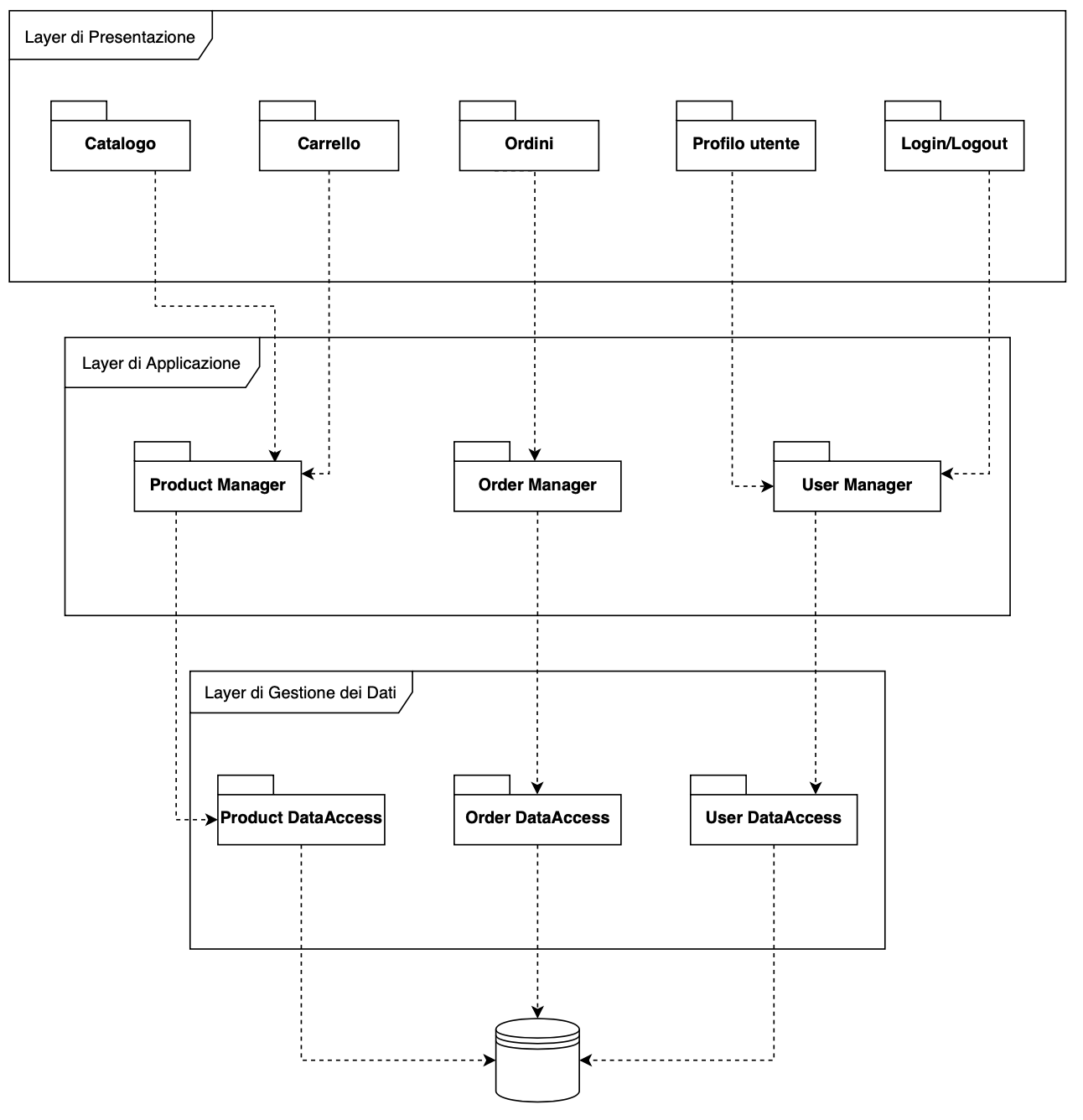
System design document

Design goals

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Priorità | Id design goal | Descrizione | Categoria | Requisito non funzionale | Trade off |
| 1 | DG-1 | Il sito dovrà essere responsive e dovrà adattarsi a diversi dispositivi e risoluzioni | Usabilità | NFR01 | Funzionalità vs usabilità  Trattandosi di un sito di e-commerce si preferisce dare priorità all’usabilità rispetto alla funzionalità |
| 1 | DG-2 | Il sistema dovrà essere in grado di gestire input errati dall’utente, cercando di far correggere l’immissione dei dati | Affidabilità | NFR02 | Affidabilità vs costo  Nonostante risulti essere più dispendioso, si preferisce utilizzare meccanismi di controllo dell’inserimento dei dati in input al fine di evitare errori |
| 1 | DG-3 | Il sistema dovrà essere in grado di gestire manomissioni tramite sql injection | Affidabilità | NFR03 |  |
| 2 | DG-4 | Il sistema utilizzerà un protocollo per la comunicazione sicura attraverso una rete di computer utilizzato su Internet (https) | Affidabilità | NFR04 |  |
| 3 | DG-5 | Il sistema provvederà a crittografie i dati sensibili dell’utente tramite crittografia MD5 | Affidabilità | NFR05 |  |
| 2 | DG-6 | Il sistema dovrà garantire un tempo di risposta relativamente basso (<4s) anche in condizioni di traffico elevato | Prestazioni | NFR06 | Prestazioni vs modificabilità  Trattandosi di una piattaforma web si preferisce dare priorità alla performance piuttosto che alla logica del software |
| 2 | DG-7 | Per inserire un nuovo prodotto l’amministratore deve impiegare meno di 3 min | Prestazioni | NFR07 |  |
| 3 | DG-8 | Il sistema dovrebbe sopportare la connessione di 300 utenti simultaneamente, senza subire crolli | Prestazioni | NFR08 |  |

System Decomposition



[Navigation Path Utente Generico -> NPUG]

[Navigation Path Admin -> NPA]

[Navigation Path Gestore Catalogo -> NPGC]

[Navigation Path Gestore Ordini-> NPGO]

Decomposizione del sistema in sottosistemi utilizzando i 3 layer forniti dall’architettura MVC.

1. **Layer di Presentazione**: livello che si occupa dell’interazione tra utente e sistema attraverso componenti grafiche che rappresentano le funzionalità del sistema
   1. Catalogo: Sottosistema che gestisce le interfacce grafiche, che permette:
      * La visualizzazione dei prodotti nel catalogo da parte del guest o cliente; [NPUG -> Home]
      * La visualizzazione del prodotto singolo da parte del guest o cliente; [NPUG -> Prodotto]
      * Ricerca dei prodotti; [NPUG -> Cerca]
      * Gestione da parte del gestore del catalogo della lista di prodotti;  
        [NPGC -> Prodotti]
   2. Carrello: Sottosistema che gestisce le interfacce grafiche, che permette la visualizzazione, la modfica della quantità, la cancellazione e l’inserimento dei prodotti scelti dal guest o cliente;   
      [NPUG -> Carrello]
   3. Ordine: Sottosistema che gestisce le interfacce grafiche, che gestisce:
      * La visualizzazione degli ordini effettuati dal cliente; [NPUG->Ordini]
      * Visualizzare il dettaglio di un ordine; [NPUG->Dettaglio ordine]
      * Stampare la fattura di un ordine; [NPUG->Fattura]
      * Gestione da parte del gestore degli ordini della lista ordini;   
        [NPGO->Ordini]
   4. Profilo utente: Sottosistema che gestisce le interfacce grafiche, che permette:
      * Di gestire i dati del cliente e carte di credito e indirizzi di spedizione;  
        [NPUG->Profilo utente]
   5. Login/Logout: Sottosistema che gestisce le interfacce grafiche, che mostra le funzionalità per effettuare:
      * Login e logout; [NPUG->Accedi]
      * Recupero password; [NPUG->Accedi]
      * Registrazione di un nuovo cliente; [NPUG->Registrati]
      * Registrazione di un nuovo gestore del catalogo da parte di un admin;  
        [NPA->form inserimento nuovo gestore del catalogo]
      * Registrazione di un nuovo gestore degli ordini da parte di un admin;  
        [NPA->form inserimento nuovo gestore degli ordini]
2. **Layer di Applicazione**: livello che si occupa della gestione della logica di business e al fine di produrre i risultati da inoltrare al Layer di Presentazione
   1. Product manager: sottosistema che permette:
      * La gestione dei prodotti;
      * La visualizzazione dei prodotti;
   2. Order manager: sottosistema che permette:
      * La gestione degli ordini;
      * La visualizzazione degli ordini;
   3. User manager: sottosistema che permette:
      * Il login e logout;
      * La registrazione di un nuovo cliente;
      * La registrazione di un nuovo gestore ordini e gestore catalogo;
      * Recupero e cambio password;
      * Visualizzare i dati personali del cliente;
      * Gestione dei dati personali del cliente;
3. **Layer di Gestione dei Dati**: Livello che gestisce i dati persistenti necessari al funzionamento di tutto il sistema
   1. Product DataAccess: Sottosistema che ha il ruolo di interagire con la base di dati. DAO:
      * Prodotto
        + doSave(product);
        + updateProduct(product)
        + getProduct(int id);
        + deleteProduct(int id)
        + updateQuantità(int id, int q);
        + doRetriveByTipo\_Annata(tipo, annata);
        + doRetriveByTipo(tipo);
        + doRetriveByAnnata(annata);
        + doRetiveByCategory(string cat);
   2. Order DataAccess: Sottosistema che ha il ruolo di interagire con la base di dati. DAO:
      * Ordini
        + doRetiveAllOrderByClient(int id);
        + doRetriveOrderByData(data1, data2);
        + doRetriveOrderByData\_Cliente(data1, data2, int idCliente);
        + doSave(order);
      * Voce Ordine
        + getFattura(int idOrder);
        + getDettaglioOrdine(int idOrder);
        + doSave(order);
   3. User DataAccess: Sottosistema che ha il ruolo di interagire con la base di dati. DAO:
      * Cliente
        + newCliente(cliente);
        + login();
        + logout();
        + deleteAccount(int idCliente);
        + passwordRecovery(string email);
        + confirmAccount(int idCliente);
      * Admin
        + newGestoreCatalogo();
        + newGestoreOrdini();
      * Carte di credito
        + deleteCreditCard(creditCard);
        + newCreditCard(creditCard);
      * Indirizzi di spedizione
        + newIndirizzoSpedizione(indirizzoSpedizione);
        + deleteIndirizzoSpedizione(int id);
        + updateIndirizzoSpedizione(indirizzoSpedizione);