**SYSTEM DESIGN DOCUMENT**  
VERSIONE 1.0 13/12/2020  
2020/2021

**PROGETTO A CURA DI:  
  
COPPOLA PIETRO: 0512103934  
CARLO VITALE**: **0512103034**   


1° Versione: 29/10/2020

2° Versione: 10/11/2020

3° Versione: 24/11/2020

4° Versione: 28/01/2021

1. Introduzione  
   1.1 Scopo del sistema  
   1.2 Obiettivi del design  
   1.3 Definizioni, acronimi e abbreviazioni   
   1.4 Riferimenti  
   1.5 Panoramica
2. Architettura del software corrente
3. Architettura del software proposto  
   3.1 Panoramica  
   3.2 Decomposizione del sistema  
   3.3 Mapping Hardware / Software  
   3.4 Gestione dei dati persistenti  
   3.5 Controllo degli accessi e della sicurezza  
   3.6 Controllo del software globale  
   3.7 Condizioni Boundary  
    3.7.1 Avvio del sistema  
    3.7.2 Terminazione del sistema  
    3.7.3 Fallimento del sistema
4. Servizi dei sottosistemi
5. Glossario  
     
     
     
     
     
     
     
     
     
     
     
     
     
     
     
     
     
     
     
     
     
     
   **1. Introduzione  
     
   1.1 Scopo del sistema**Un produttore di riproduzioni di armi bianche, dopo anni di vendita su siti come e-bay, intende creare un proprio portale e-commerce per vendere i propri prodotti, arrivando a crearsi un'identità sul web.  
   A tal fine questo sito deve essere “immediato” per i suoi clienti, magari rimandando nella struttura a siti di ampio utilizzo ed estremamente user-friendly come Amazon.   
   Ne consegue la presenza di un catalogo di ampio respiro accompagnato da immagini e descrizione dei prodotti, e una categorizzazione dei suddetti prodotti in diverse, appunto, categorie. Inoltre la possibilità di potersi iscrivere creando e gestendo una propria pagina personale “Utente” con la quale consultare lo storico dei propri acquisti, ma anche la possibilità di sfogliare il catalogo e vedere i prodotti senza effettuare l’accesso al sito.  
   Lo scopo consiste dunque nel creare un portale di vendita di riproduzioni di armi bianche al grande pubblico tramite un sito di facile utilizzo.  
     
   **1.2 Obiettivi del design**L’obiettivo è creare un portale di vendita di facile utilizzo, per definirlo un sito di successo bisogna che:  
   - Gestita il profilo di un Utente e di un Admin in modo diverso;  
   - Permetta di visualizzare lo storico dei propri acquisti e le relative fatture;  
   - Divida gli articoli in vendita per categorie;  
   - Sia di facile da usare (User-Friendly);  
   - Tenga conto in modo corretto della disponibilità degli oggetti tramite il Database a lui collegato;

**1.2.1 Criteri di Performance**

**Tempo di risposta**

Il Tizzone Ardente deve essere abbastanza reattivo per tutte le operazioni più immediate. Per operazioni massive, come la sincronizzazione dal database, il sistema deve garantire tempi di risposta ragionevoli nell’ordine delle decine di secondi, ma, essendo un sistema web, molto dipenderà dalla qualità della connessione e dalla congestione della rete sul sistema online.

**Throughput**

Il sito dovrà essere in grado di gestire contemporaneamente le richieste di diversi utenti. Attualmente, non è possibile avere una stima precisa e concreta del carico, ma devono essere gestiti dal sistema senza rallentamenti, garantendo fluidità e una latenza molto bassa. Il sistema deve garantire consistenza in tutte le operazioni che includono chiamate al database, specialmente nei momenti di maggiore carico.

**Memoria**

Il sistema utilizza un database relazionale per memorizzare tutti i dati. La mole dei dati non rappresenterà un problema di performance del sistema.

**1.2.2 Criteri di Affidabilità**

**Robustezza**

Dobbiamo notificare all’utente un messaggio di errore nel caso di immissione di dati non validi senza interrompere il funzionamento del sito. Esso deve essere in grado di poter mantenere i propri dati anche in caso di guasti dell'hardware.

**Disponibilità**

Il Tizzone Ardente deve essere attivo all’uso e a richieste da parte degli utenti, 24 ore su 24, grazie ad un server che rimane sempre attivo.

**Tolleranza all’errore**

Il sistema deve essere capace di operare durante condizioni d’errore. Ciò sarà reso possibile da tutte quelle sotto-parti del sistema che hanno un grado di accoppiamento basso, in modo tale che l’errore in un sottosistema non influisca sulle funzionalità di un altro. Inoltre, deve essere in grado di pianificare dei backup periodici dell’intero sistema.

**Sicurezza**

Le tecniche utilizzate, per garantire la sicurezza, sono basate prevalentemente su una ‘login’, la quale permette il riconoscimento dell’utente e non permette l’accesso a parti del sistema non autorizzate. Tutti gli accessi al sistema avvengono tramite un’operazione di check che controlla se l’utente che sta operando sul sistema è iscritto come un cliente oppure come amministratore. C’è, quindi, un controllo che verifica i permessi degli utenti che verranno adeguatamente distribuiti sulla base del grado di importanza e responsabilità di questi. In particolare un amministratore avrà i massimi permessi poiché ha facoltà di utilizzare qualsiasi tipo di funzione disponibile, infine un utente ha la facoltà di utilizzare qualsiasi tipo di funzione disponibile relativa alla gestione e all’organizzazione del proprio profilo.

**1.2.3 Criteri di costo**

I costi di sviluppo previsti riguardano l’affitto del server per la fase di sviluppo. Per quanto riguarda il DBMS utilizzeremo MySQL che è open source. Inoltre i costi di sviluppo prevedono la retribuzione per il lavoro dei singoli componenti del team, In aggiunta sono previsti i costi delle licenze software: Balsamiq, PhotoShop, Microsoft Office.

**1.2.4 Criteri di manutenzione**

**Estensibilità**

Il sistema dovrà essere orientato all’ aggiunta di nuove funzionalità o nuove classi. Quindi il codice dovrà essere strutturato e anche ben comprensibile per evitare ambiguità, nel momento in cui c’è uno sviluppo successivo.

**Modificabilità**

Deve essere possibile intervenire sul codice esistente per correggere eventuali bugs o implementare nuove funzionalità. Il codice deve essere leggibile per rendere agevole la modifica.

**Leggibilità**

Il codice sarà ben strutturato e facile da comprendere, in quanto ci saranno opportuni commenti (in particolare per le operazioni più inesplicabili) per semplificare eventuali interventi su di esso.

**Tracciabilità dei requisiti**

Grazie alla tracciabilità dei requisiti, sarà possibile effettuare le modifiche necessarie al corretto funzionamento del sistema, valutando correttamente i costi e i rischi che le modifiche porteranno.

**1.2.5 Criteri per l’utente finale**

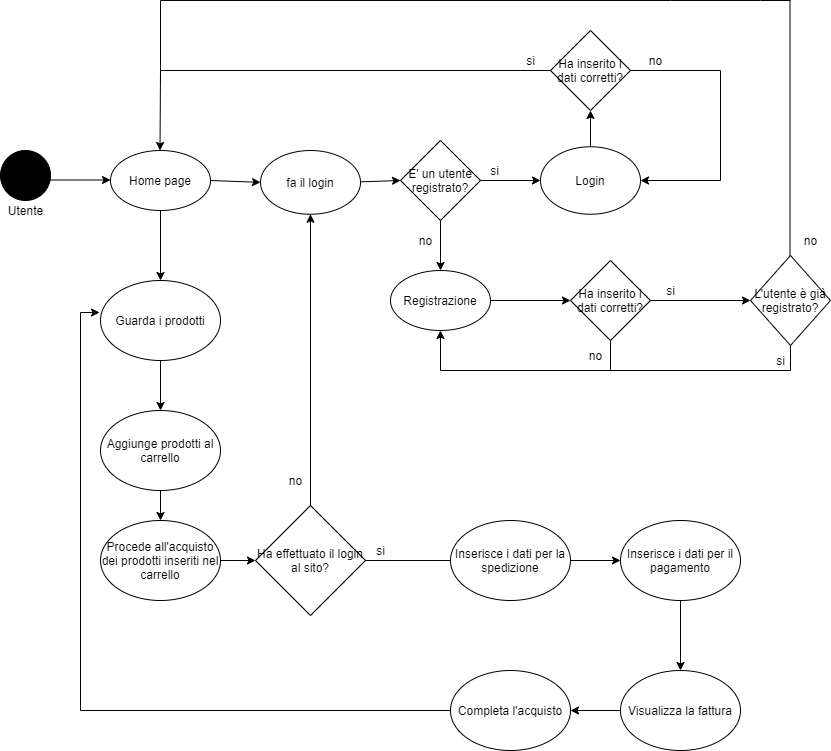
**Usabilità**

Il Tizzone Ardente deve essere:

* User-friendly (l’utente deve essere in grado di interagire con il sistema e padroneggiare le funzionalità in modo rapido);
* Dare la possibilità all’utente di capire quando ha successo nel perseguire i suoi obiettivi nel sistema o quando sta sbagliando qualcosa per poter intervenire;
* Utile (soddisfare le esigenze del cliente).

**1.3 Definizione, acronimi e abbreviazioni**- Il Tizzone Ardente: Nome del sistema.  
- Utente: Attore del sistema, può effettuare diverse operazioni come: ricerca, acquisti, login / logout e modifiche. Si divide in 2 sottocategorie,  
 - Cliente;  
 - Admin;  
- Login: Attività di accesso al sito.  
- Logout: Attività di uscita dal sito.  
- User-friendly: Termine che indica il facile utilizzo ed approccio al sito per chiunque.  
- Cliente: Attore del sistema che può effettuare operazioni tra cui: ricerca, aggiunta di prodotti al carrello, acquisti, controllare il proprio storico ecc.  
- Admin / Amministratore: Attore del sistema che può effettuare operazioni tra cui: modifica di un prodotto, aggiungere / eliminare un prodotto, ecc.  
- Database (DB): Termine che indica la banca dati affiancata al sito;  
  
**1.4 Riferimenti**- *Object Oriented Software Engineering – Using UML, Patterns and Java, Prentice Hall, 3rd edition, 2009*- *Ingegneria del software. Fondamenti e principi, edizione 2, 2004***1.5 Panoramica**Il Tizzone Ardente è un sito e-commerce che permetterà degli acquisti di riproduzioni di armi bianche in modo semplice ed immediato, il sistema consisterà:  
- Registrazione di utenti  
- Sfogliare il catalogo tramite l’interfaccia grafica  
- Sfogliare il catalogo tramite navbar  
- Esaminare prodotti nello specifico  
- Aggiungere prodotti al carrello  
- Acquistare i prodotti presenti nel carrello  
- Sfogliare lo storico e le fatture degli acquisti legati ad un account  
- Cancellare o aggiornare il proprio Account personale

**2. Architettura del software corrente**

Attualmente la gestione delle vendite di queste riproduzioni avviene tramite market digitali come E-bay, ma lasciano poca personalizzazione e limitano la possibilità di crearsi una propria identità. Nasce, perciò, a partire dai bisogni dell’utente.  
  
**3. Architettura del software proposto**  
****

**3.1 Panoramica**

Il sistema da noi proposto è un sito web e-commerce per la vendita di riproduzioni di armi bianche.

Il sistema supporta due tipi di utenti:

**Cliente**: è uno degli attori principali del sistema e interagisce con le sue principali funzionalità. Può decidere di registrarsi al sito creando un proprio account, oppure se è già registrato di effettuare il login. Visualizza i prodotti che sono venduti all’interno del sito. Aggiunge al carrello il prodotto che desidera comprare. Procede poi all’acquisto inserendo le informazioni per la spedizione, il metodo di pagamento e visualizzando la fattura. Può poi visualizzare gli ordini che ha effettuato in passato.

**Amministratore:** attore principale, il cui compito è la gestione il sistema. Può infatti aggiungere, modificare o eliminare i prodotti all’interno del sito e può visualizzare tutti gli ordini effettuati sul sito. Inoltre può utilizzare tutte le funzionalità descritte per i clienti.

La divisione in classi nei vari sottosistemi sarà individuata in base alle funzionalità per poter rendere massima la coesione e minimo l’accoppiamento tra i sottosistemi in modo che i cambiamenti in un sottosistema non influiscano sugli altri

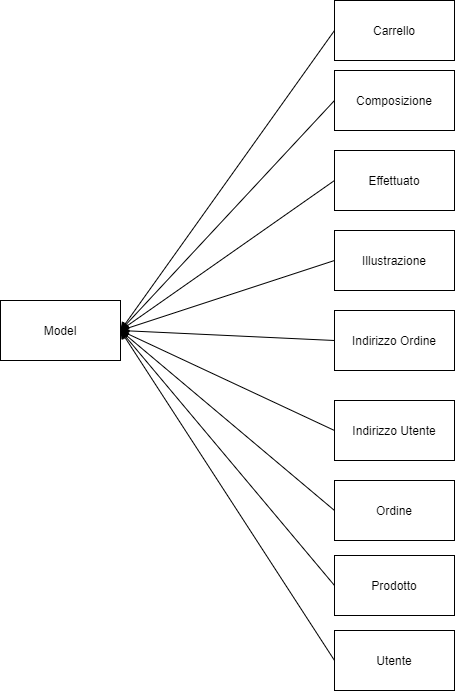
Il sistema si basa su un architettura Model-View-Controller, infatti le funzionalità saranno divise in layer logici: sistema di memorizzazione (model), presentazione (view) e business logic (controller)

**3.2 Decomposizione in sottosistemi**

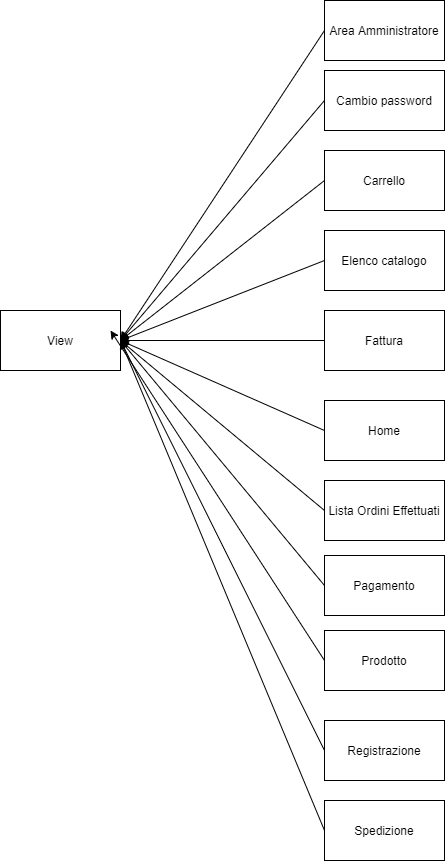
È stata utilizzata l’architettura MVC (model-view-controller) perché particolarmente adatta per applicazioni web e siti di e-commerce come il nostro. Questo stile di architettura presenta una netta divisione tra interfaccia, rappresentata dal sottosistema view, controllo della gestione dei dati persistenti, rappresentato dal sottosistema model, e controllo della logica di business rappresentato dal sistema controller.

Decomponiamo, quindi, il sistema in tre tipi che gestiscono aspetti e funzionalità differenti

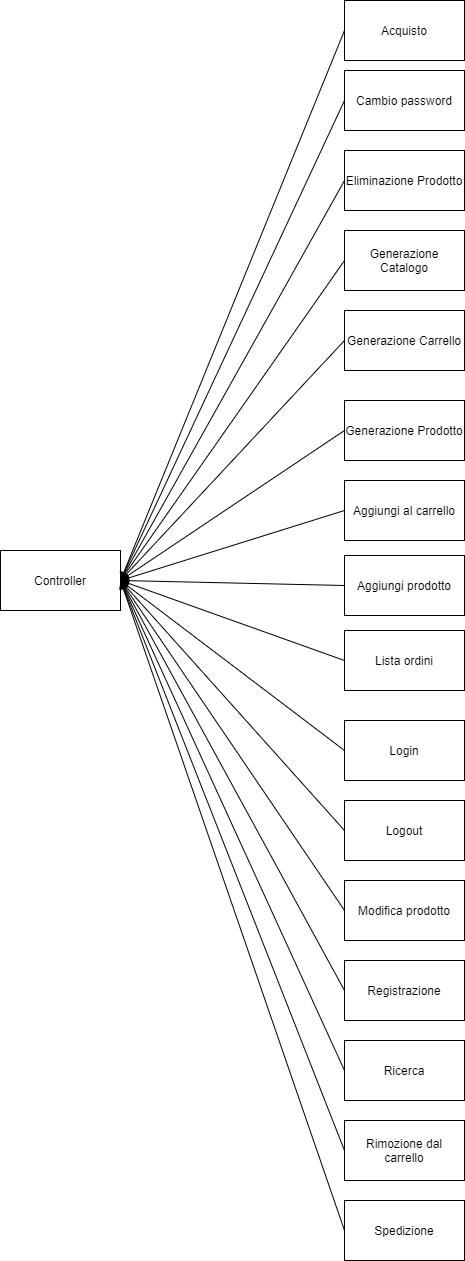
* Sottosistema Model: si occupa della gestione e dello scambio dei dati tra i sottosistemi. In pratica fornisce le operazioni per accedere ai dati necessari al funzionamento del sito. Nel nostro sottosistema abbiamo identificato i seguenti model:



* Sottosistema View: gestisce l’interfaccia grafica e l’interazione con gli utenti. Nel sottosistemi abbiamo identificato le seguenti view:



* Sottosistema Controller: responsabile della sequenza di interazioni con l’utente (riceve i comandi dell’utente attraverso le view e li attua modificando lo stato delle altre due componenti). In pratica si occupa della gestione della logica del sistema. Nel sottosistema abbiamo identificato i seguenti controller



**3.3 Mapping Hardware/Software**

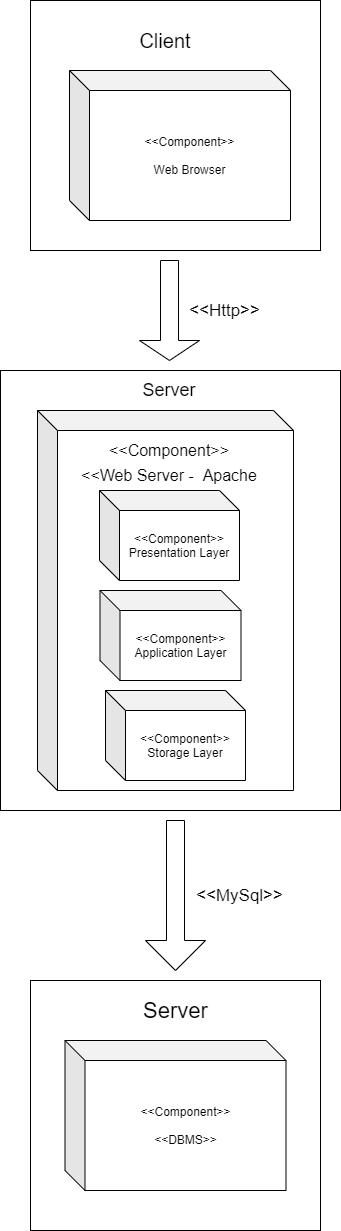
La struttura hardware proposta è costituita da un server centrale e dai client, cioè un qualsiasi computer al quale un utente può collegarsi per sfruttare il browser per accedere al Tizzone Ardente. Al server si collegano i client ed il database, il tipo di utente è determinato in fase di autenticazione controllando nel database il tipo di utente che corrisponde all’email con la quale si autentica. I client dovranno effettuare richieste al server per eseguire le operazioni legate alle loro rispettive funzionalità. II client ed il server saranno connesse tramite una rete che utilizzerà il protocollo TCP/IP.

**Presentation** **layer** : Ha come obiettivo quello di trasformare i [dati](https://it.wikipedia.org/wiki/Dato) forniti dal [livello di applicazione](https://it.wikipedia.org/wiki/Livello_di_applicazione) in un formato standard. In altre parole include i Boundary Object che permettono l’interazione con l’utente.

**Application layer** : Include la logica di business. Si riferisce a tutta quella logica applicativa che rende operativo un sito. Assieme al lato DB rappresenta dunque il back-end del sito. Il codice risultante viene inviato al browser del client.

**Storage layer** : Rappresenta il collegamento con il server da parte del sistema e si occupa di tutte le richieste di accesso e modifiche sui dati permanenti presenti nel database.

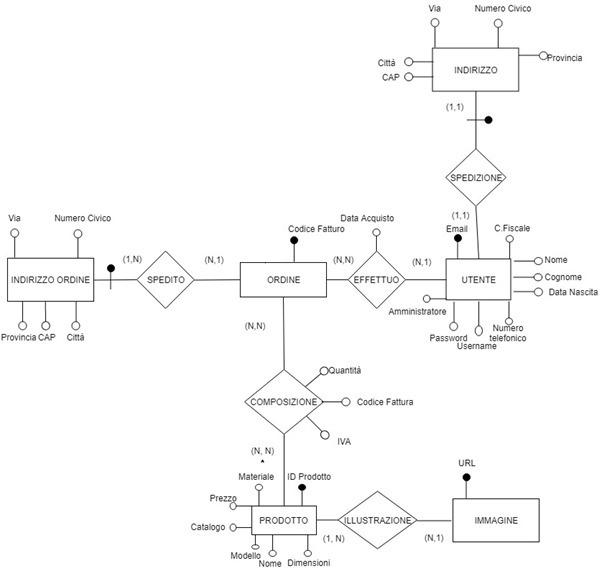
**Database Server** : Il DBMS usato è MySQL il quale presenta molte API che permettono l’interazione tra sistema e database.



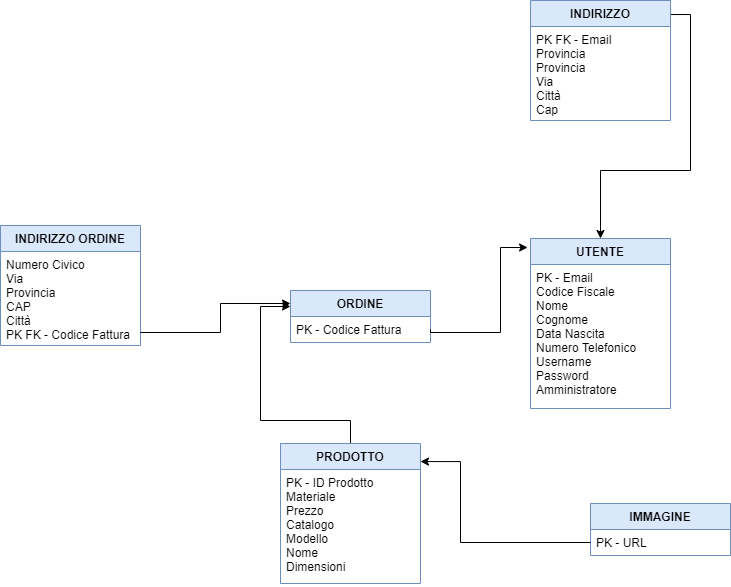
**3.4 Gestione dei dati persistenti**

Il Tizzone Ardente deve effettuare delle interrogazioni sui dati, inoltre si ha il bisogno di gestire la memorizzazione persistente delle entità. Il Tizzone Ardente utilizza un Database di tipo relazionale. Il Database utilizzato è il DBMS relazionale MySql. Di seguito riportiamo lo schema E-R (Entity-Relationship), la traduzione nel modello relazionale e la struttura delle varie tabelle.

**3.4.1 Schema E-R (Ristrutturato)**

****

**3.4.2 Modello Relazionale**

****

**3.4.3 Struttura Tabelle**

**Composizione**

|  |  |  |  |
| --- | --- | --- | --- |
| CAMPO | TIPO | NULL | KEY |
| CodiceFattura | varchar(20) | not null | foreign key |
| IDProdotto | varchar(20) | not null | foreign key |
| IVA | int(11) | not null |  |
| PrezzoPagato | double | not null |  |
| QuantitàComposizione | int(11) | not null |  |

**Effettuare**

|  |  |  |  |
| --- | --- | --- | --- |
| CAMPO | TIPO | NULL | KEY |
| CodiceFattura | varchar(20) | not null | foreign key |
| DataDiAcquisto | date | not null |  |
| Email | varchar(50) | not null | foreign key |

**Illustrazione**

|  |  |  |  |
| --- | --- | --- | --- |
| CAMPO | TIPO | NULL | KEY |
| IDProdotto | varchar(10) | not null | foreign key |
| URL | varchar(100) | not null |  |

**Indirizzo Ordine**

|  |  |  |  |
| --- | --- | --- | --- |
| CAMPO | TIPO | NULL | KEY |
| CAP | varchar(5) | not null |  |
| Città | varchar(20) | not null |  |
| CodiceFattura | varchar(20) | not null | foreign key |
| Indirizzo | varchar(20) | not null |  |
| NumeroCivico | varchar(3) | not null |  |
| Provincia | varchar(20) | not null |  |

**Indirizzo Utente**

|  |  |  |  |
| --- | --- | --- | --- |
| CAMPO | TIPO | NULL | KEY |
| CAP | varchar(5) | not null |  |
| Città | varchar(20) | not null |  |
| Email | varchar(50) | not null | foreign key |
| Indirizzo | varchar(20) | not null |  |
| NumeroCivico | int(3) | not null |  |
| Provincia | varchar(20) | not null |  |

**Ordine**

|  |  |  |  |
| --- | --- | --- | --- |
| CAMPO | TIPO | NULL | KEY |
| CodiceFattura | varchar(20) | not null | primary key |

**Prodotto**

|  |  |  |  |
| --- | --- | --- | --- |
| CAMPO | TIPO | NULL | KEY |
| Catalogo | varchar(20) | not null |  |
| Descrizione | varchar(200) |  |  |
| IDProdotto | varchar(20) | not null | primary key |
| Materiale | varchar(50) | not null |  |
| Modello | varchar(50) | not null |  |
| Nome | varchar 50 | not null |  |
| Peso | double | not null |  |
| Prezzo | double | not null |  |
| Quantità | int(11) | not null |  |
| Sconto | int(11) |  |  |

**Utente**

|  |  |  |  |
| --- | --- | --- | --- |
| CAMPO | TIPO | NULL | KEY |
| Amministratore | tinyint(1) | not null |  |
| Cognome | varchar(20) | not null |  |
| Email | varchar(50) | not null | primary key |
| Nome | varchar(20) | not null |  |
| NumeroDiTelefono | varchar(10) | not null |  |
| Password | varchar(20) | not null |  |

**3.5 Controllo degli accessi e della sicurezza**

Il controllo degli accessi è garantito tramite l’utilizzo di username e password per il cliente che per l’amministratore.

La sicurezza sui dati sensibili degli utenti è garantita dall’accesso controllato in quanto solo l’utente stesso può visualizzare i suoi dati sensibili

Le operazioni che l’utente della Dashboard può fare sono:

|  |  |  |
| --- | --- | --- |
| Attore/Sottosistema | **Cliente** | **Amministratore** |
| **Gestione Accesso** | -Login  -Logout  -Registrazione | -Login  -Logout |
| **Gestione Acquisto** | -Aggiungere prodotti al carrello  -Rimuovere prodotti al carrello  -Procedere all’acquisto  -Inserire le informazioni di spedizione  -Inserire le informazioni per il pagamento  -Visualizzare la fattura e confermare l’acquisto | -Aggiungere prodotti al carrello  -Rimuovere prodotti al carrello  -Procedere all’acquisto  -Inserire le informazioni di spedizione  -Inserire le informazioni per il pagamento  -Visualizzare la fattura e confermare l’acquisto |
| **Gestione Funzionalità Amminsitratore** |  | -Aggiungere un prodotto per la vendita  -Modificare un prodotto presente nel sistema  -Rimuovere un prodotto presente nel sistema |
| **Gestione Visualizzazione Ordini** | -Visualizzare i suoi ordini passati | -Visualizzare i suoi ordini passati  -Visualizzare tutti gli ordini effettuati sul sito |
| **Gestione Account** | -Cambio password | -Cambio password |

**3.6 Controllo del software globale**

Il Tizzone Ardente è un sistema che utilizza un flusso guida dagli eventi (event-driven). Quindi, è l’utente a scegliere di seguito l’operazione da eseguire.

L’architettura software è composta da un web server che rimane in ascolto in attesa di una richiesta da parte di un web browser. Il controllo del flusso software viene gestito da classi che interagendo con il client, il quale si interfaccia tramite un web browser, svolgono le varie operazioni. Il server smista ogni nuova richiesta alla classe adeguata, inoltrando poi la risposta al client.

**3.7 Condizioni Boundary**

Le condizioni limite riguardano l’accensione e lo spegnimento del sistema per quanto riguarda il lato Server. Dal lato Client si riferiscono agli errori di connessione al server.

**3.7.1 Avvio del sistema**

Il sistema dopo essersi avviato presenta un’interfaccia ai client. Dopo aver effettuato l’autenticazione ogni utente può accedere alle funzionalità disponibili.

**3.7.2 Fallimento del sistema**

Nel caso si verifichi un errore dovuto all’hardware o al software si cercherà di ripristinare una configurazione del sistema precedente allo stato d’errore. Poiché i dati sono gestiti dal DBMS non c’è alcun rischio di perderli. Tuttavia, non è da escludere la perdita dei dati, se si verifica un guasto al supporto di memorizzazione dei dati nel database-server. Per minimizzare questo rischio, si eseguiranno periodicamente dei backup del database del sistema e periodicamente il sistema hardware verrà sottoposto a controlli.

**4. Servizi dei sottosistemi**

**4.1 Gestione accesso**

|  |  |
| --- | --- |
| **Nome** | **Descrizione** |
| Accedi | Permette ad ogni tipo di utente, previa registrazione al sistema, di accedere al sito |
| Logout | Permette a ogni tipo di utente, previo accesso al sistema, di poter effettuare la disconnessione dal sistema |
| Registrazione | Offre agli utenti, la possibilità di registrarsi al sistema per poter usufruire dei servizi. |

**4.1 Gestione Acquisto**

|  |  |
| --- | --- |
| **Nome** | **Descrizione** |
| Carrello | Permette a ogni tipo di utente di poter aggiungere al carrello uno o più prodotti. Inoltre permette anche di rimuovere uno o più prodotti dal carrello. |
| Acquisto | Permette a ogni tipo di utente, che ha almeno un prodotto all’interno del carrello, di poter effettuare le operazioni necessarie all’acquisto. |
| Spedizione | Permette all’utente di inserire le informazioni per la spedizione dei prodotti. |
| Pagamento | Permette all’utente, previo l’inserimento dei dati di spedizione, di inserire i dati per il pagamento. |
| Fattura | Permette all’utente, previo l’inserimento del metodo di pagamento, di visualizzare la fattura in modo da poter confermare l’acquisto. |

**4.3 Gestione Funzionalità Amministratore**

|  |  |
| --- | --- |
| **Nome** | **Descrizione** |
| Aggiunta prodotto | Permette all’amministratore di poter aggiungere al sito un nuovo prodotto disponibile per la vendita. |
| Modifica prodotto | Permette all’amministratore di poter modificare i dati di un prodotto all’interno del sito |
| Rimozione prodotto | Permette all’amministratore di poter rimuovere un prodotto dal sito. |

**4.4 Gestione Visualizzazione Ordini**

|  |  |
| --- | --- |
| **Nome** | **Descrizione** |
| Visualizza ordini cliente | Permette ad ogni tipo di utente, previo accesso al sito, di visualizzare gli acquisti che ha effettuato sul sito |
| Visualizza ordini amministratore | Permette all’amministratore di visualizzare tutti gli ordini che sono stati effettuati sul sito |

**4.5 Gestione Account**

|  |  |
| --- | --- |
| **Nome** | **Descrizione** |
| Cambio password | Permette a ogni tipo di utente, previo accesso al sito, di poter cambiare la password del proprio account |

**5. Glossario**

**Amministratore**: il termine identifica un attore del sistema che gode dei privilegi di amministratore

**Database**: insieme organizzato di dati persistenti

**Prodotto**: il termine identifica un oggetto che può essere acquistato

**Tizzone Ardente**: nome del sistema che verrà sviluppato

**Utente**: il termine identifica un attore del sistema che può usufruire dei servizi offerti