Gestione carrello con più fornitori e ordine

Versione JavaScript

Fumagalli Stefano, Ferro Lara, Rigamonti Davide

ANALISI DEI DATI

Legenda

- entità
- attributi
- relazioni

Testo

Un'applicazione di commercio elettronico consente all'utente (acquirente) di visualizzare un catalogo di prodotti venduti da diversi fornitori, inserire prodotti in un carrello della spesa e creare un ordine di acquisto a partire dal contenuto del carrello.

Un prodotto ha un codice (campo chiave), un nome, una descrizione, una categoria merceologica e una foto. Lo stesso prodotto (cioè codice prodotto) può essere venduto da più fornitori a prezzi differenti.

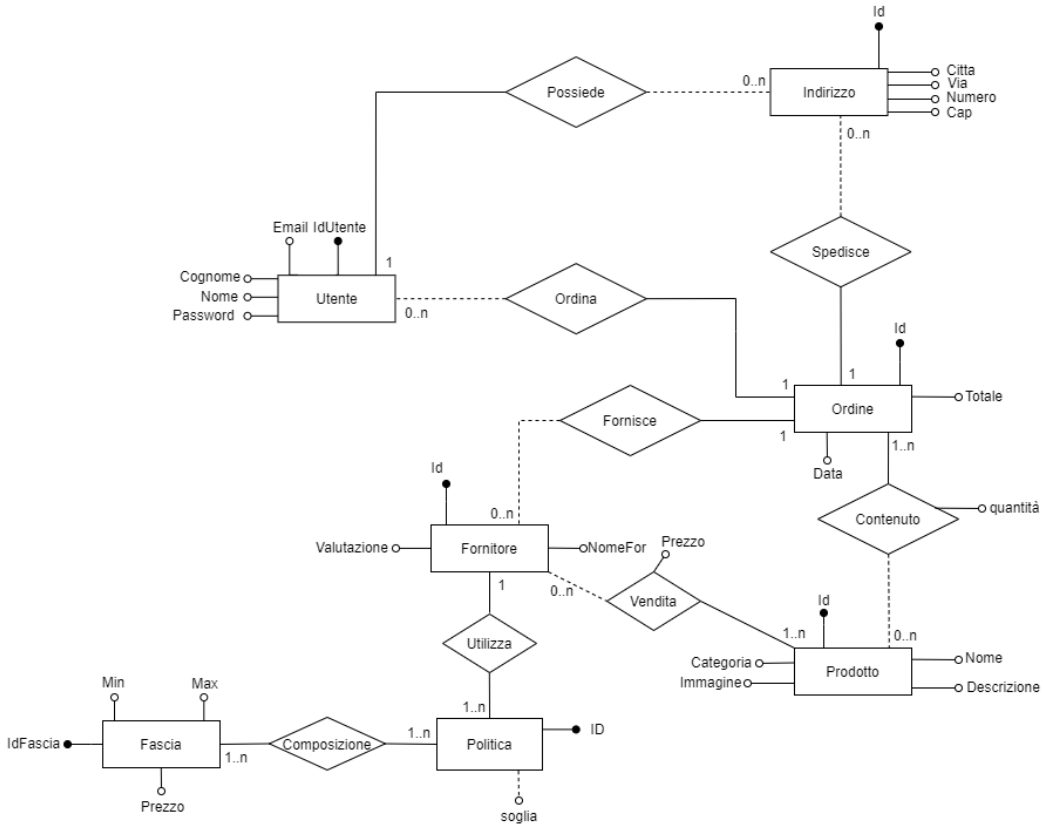
Un fornitore ha un codice, un nome, una valutazione da 1 a 5 stelle e una politica di spedizione.

Un utente ha un nome, un cognome, un'e-mail, una password e un indirizzo di spedizione.

La politica di spedizione precisa il prezzo della spedizione in base al numero di articoli ordinati. Ogni fornitore è libero di definire fasce di spesa. Una fascia di spesa ha un numero minimo, un numero massimo e un prezzo. Ad esempio: da 1 a 3 articoli 15€, da 4 a 10 articoli 20€, oltre a 10 articoli, ecc. Oltre alla fascia di spesa, il fornitore può anche indicare un importo in euro oltre al quale la spedizione è gratuita.

Un ordine ha un codice, il nome del fornitore, l'elenco dei prodotti, un valore totale composto dalla somma del valore dei prodotti e delle spese di spedizione, una data di spedizione e l'indirizzo di spedizione dell'utente.

DESIGN DEL DATABASE



SCHEMA DEL DATABASE

```
CREATE TABLE `utente` (
	`IdUtente` int NOT NULL AUTO_INCREMENT,
	`Nome` varchar(16) NOT NULL,
	`Cognome` varchar(16) NOT NULL,
	`Email` varchar(64) NOT NULL,
	`Password` varchar(16) NOT NULL,
	`IdIndirizzo` int NOT NULL,
	PRIMARY KEY (`IdUtente`),
	 KEY `fk_indirizzo_idx` (`IdIndirizzo`),
	CONSTRAINT `fk_indirizzo` FOREIGN KEY (`IdIndirizzo`) REFERENCES `indirizzo` (`Id`)
)

CREATE TABLE `indirizzo` (
	`Id` int NOT NULL AUTO_INCREMENT,
	`Citta` varchar(32) NOT NULL,
	`Via` varchar(32) NOT NULL,
	`Cap` char(5) NOT NULL,
	`Numero` int NOT NULL,
	PRIMARY KEY (`Id`)
)

CREATE TABLE `fascia` (
	`IdFascia` int NOT NULL AUTO_INCREMENT,
	`Prezzo` float NOT NULL,
	`Min` int NOT NULL,
	`Max` int NOT NULL,
	PRIMARY KEY (`IdFascia`)
)
```

```
CREATE TABLE `politica` (
	`Id` int NOT NULL AUTO_INCREMENT,
	`Soglia` int DEFAULT NULL,
	PRIMARY KEY (`Id`)
)

CREATE TABLE `fornitore` (
	`Id` int NOT NULL AUTO_INCREMENT,
	`NomeFor` varchar(16) NOT NULL,
	`Valutazione` varchar(8) NOT NULL,
	`IdPoliticaForn` int NOT NULL,
	PRIMARY KEY (`Id`),
	KEY `fk_politica_fornitore_idx` (`IdPoliticaForn`),
	CONSTRAINT `fk_politica_fornitore` FOREIGN KEY (`IdPoliticaForn`) REFERENCES `politica` (`Id`)
		ON DELETE CASCADE
		ON UPDATE CASCADE
)

CREATE TABLE `prodotto` (
	`Id` int NOT NULL AUTO_INCREMENT,
	`Nome` varchar(32) NOT NULL,
	`Descrizione` varchar(1024) NOT NULL,
	`Categoria` varchar(16) NOT NULL,
	`Immagine` longblob,
	PRIMARY KEY (`Id`)
)

CREATE TABLE `contenuto` (
	`IdOrdine` int NOT NULL,
	`IdProdotto` int NOT NULL,
	`Quantita` int NOT NULL,
	PRIMARY KEY (`IdOrdine`,`IdProdotto`),
	KEY `fk_prodotto_contenuto_idx` (`IdProdotto`),
	CONSTRAINT `fk_ordine_contenuto` FOREIGN KEY (`IdOrdine`) REFERENCES `ordine` (`Id`)
		ON DELETE CASCADE
		ON UPDATE CASCADE,
	CONSTRAINT `fk_prodotto_contenuto` FOREIGN KEY (`IdProdotto`) REFERENCES `prodotto` (`Id`)
		ON DELETE CASCADE
		ON UPDATE CASCADE
)

CREATE TABLE `ordine` (
	`Id` int NOT NULL AUTO_INCREMENT,
	`Totale` float NOT NULL,
	`Data`timestamp DEFAULT NULL,
	`IdIndirizzo` int NOT NULL,
	`IdUtente` int NOT NULL,
	`IdFornitore` int NOT NULL,
	PRIMARY KEY (`Id`),
	KEY `fk_indirizzo_ordine_idx` (`IdIndirizzo`),
	KEY `fk_utente_ordine_idx` (`IdUtente`),
	KEY `fk_fornitore_ordine_idx` (`IdFornitore`),
	CONSTRAINT `fk_fornitore_ordine` FOREIGN KEY (`IdFornitore`) REFERENCES `fornitore` (`Id`)
		ON DELETE CASCADE
		ON UPDATE CASCADE,
	CONSTRAINT `fk_indirizzo_ordine` FOREIGN KEY (`IdIndirizzo`) REFERENCES `indirizzo` (`Id`)
		ON DELETE CASCADE
		ON UPDATE CASCADE,
	CONSTRAINT `fk_utente_ordine` FOREIGN KEY (`IdUtente`) REFERENCES `utente` (`IdUtente`)
		ON DELETE CASCADE
		ON UPDATE CASCADE
)
```

```
CREATE TABLE `composizione` (
		`IdPoliticaComp` int NOT NULL,
		`IdFasceComp` int NOT NULL,
		PRIMARY KEY (`IdPoliticaComp`,`IdFasceComp`),
		KEY `fk_range_composizione_idx` (`IdFasceComp`),
		CONSTRAINT `fk_politica_composizione` FOREIGN KEY (`IdPoliticaComp`) REFERENCES `politica` (`Id`)
			ON DELETE CASCADE
			ON UPDATE CASCADE,
		CONSTRAINT `fk_range_composizione` FOREIGN KEY (`IdFasceComp`) REFERENCES `fascia` (`IdFascia`)
			ON DELETE CASCADE
			ON UPDATE CASCADE
)

CREATE TABLE `vendita` (
		`IdFornitore` int NOT NULL,
		`IdProdotto` int NOT NULL,
		`Prezzo` float NOT NULL,
		PRIMARY KEY (`IdFornitore`,`IdProdotto`),
		KEY `fk_prodotto_vendita_idx` (`IdProdotto`),
		CONSTRAINT `fk_fornitore_vendita` FOREIGN KEY (`IdFornitore`) REFERENCES `fornitore` (`Id`),
		CONSTRAINT `fk_prodotto_vendita` FOREIGN KEY (`IdProdotto`) REFERENCES `prodotto` (`Id`)
			ON DELETE CASCADE
			ON UPDATE CASCADE
)
```

ANALISI DEI REQUIREMENT

Legenda

- pages (views)
- view components
- events
- actions

Versione HTML pura

Un'applicazione di commercio elettronico consente all'utente (acquirente) di visualizzare un catalogo di prodotti venduti da diversi fornitori, inserire prodotti in un carrello della spesa e creare un ordine di acquisto a partire dal contenuto del carrello. Un prodotto ha un codice (campo chiave), un nome, una descrizione, una categoria merceologica e una foto. Lo stesso prodotto(cioè codice prodotto) può essere venduto da più fornitori a prezzi differenti. Un fornitore ha un codice, un nome, una valutazione da 1 a 5 stelle e una politica di spedizione. Un utente ha un nome, un cognome, un'e-mail, una password e un indirizzo di spedizione. La politica di spedizione precisa il prezzo della spedizione in base al numero di articoli ordinati. Ogni fornitore è libero di definire fasce di spesa. Una fascia di spesa ha un numero minimo, un numero massimo e un prezzo. Ad esempio: da 1 a 3 articoli 15€, da 4 a 10 articoli 20€, oltre a 10 articoli, ecc. Oltre alla fascia di spesa, il fornitore può anche indicare un importo in euro oltre al quale la spedizione è gratuita. Se il totale supera la soglia per la gratuità della spedizione, la spedizione è gratuita indipendentemente dal numero di articoli. Dopo il login, l'utente accede a una pagina HOME che mostra (come tutte le altre pagine) un menù con i link HOME, CARRELLO, ORDINI, un campo di ricerca e una lista degli ultimi cinque prodotti visualizzati dall'utente. Se l'utente non ha visualizzato almeno cinque prodotti, la lista è completata con prodotti in offerta scelti a caso in una categoria di default. L'utente può inserire una parola chiave di ricerca nel campo di input e premere INVIO. A seguito dell'invio compare una pagina

RISULTATI con prodotti che contengono la chiave di ricerca nel nome o nella descrizione. L'elenco mostra solo il codice, il nome del prodotto e il prezzo minimo di vendita del prodotto da parte dei fornitori che lo vendono (lo stesso prodotto può essere venduto da diversi fornitori a prezzi diversi e l'elenco mostra il minimo valore di tali prezzi). L'elenco è ordinato in modo crescente in base al prezzo minimo di vendita del prodotto da parte dei fornitori che lo offrono. L'utente può selezionare mediante un click un elemento dell'elenco e visualizzare nella stessa pagina i dati completi e l'elenco dei fornitori che lo vendono a vari prezzi (questa azione rende il prodotto "visualizzato"). Per ogni fornitore in tale elenco compaiono: nome, valutazione, prezzo unitario, fasce di spesa di spedizione, importo minimo della spedizione gratuita e il numero dei prodotti e valore totale degli dei prodotti di quel fornitore che l'utente ha già messo nel carrello. Accanto all'offerta di ciascun fornitore compare un campo di input intero (quantità) e un bottone METTI NEL CARRELLO. L'inserimento nel carrello di una quantità maggiore di zero di prodotti comporta l'aggiornamento del contenuto del carrello e la visualizzazione della pagina CARRELLO. Questa mostra i prodotti inseriti, raggruppati per fornitore. Per ogni fornitore nel carrello si vedono la lista dei prodotti, il prezzo totale dei prodotti e il prezzo della spedizione calcolato in base alla politica del fornitore. Per ogni fornitore compare un bottone ORDINA. Premere il bottone comporta l'eliminazione dei prodotti del fornitore dal carrello e la creazione di un ordine corrispondente. Un ordine ha un codice, il nome del fornitore, l'elenco dei prodotti, un valore totale composto dalla somma del valore dei prodotti e delle spese di spedizione, una data di spedizione e l'indirizzo di spedizione dell'utente. I valori degli attributi di un ordine sono memorizzati esplicitamente nella base di dati indipendentemente dai dati del carrello. In ogni momento l'utente può accedere tramite il menu alla pagina HOME, ORDINI e CARRELLO. La pagina ORDINI mostra l'elenco ordinato per data decrescente degli ordini con tutti i dati associati. L'applicazione NON salva il carrello nella base di dati ma solo gli ordini.

Versione con JavaScript

Si realizzi un'applicazione client server web che estende e/o modifica le specifiche precedenti come segue:

- Dopo il login dell'utente, l'intera applicazione è realizzata con un'unica pagina.
- Ogni interazione dell'utente è gestita senza ricaricare completamente la pagina, ma produce l'invocazione asincrona del server e l'eventuale modifica del contenuto da aggiornare a seguito dell'evento.
- L'applicazione memorizza il contenuto del carrello a lato client.
- Nella pagina RISULTATI l'elenco dettagliato dei prodotti già nel carrello da parte di un fornitore compare mediante una finestra sovrapposta quando si passa con il mouse sopra il numero che indica quanti prodotti del medesimo fornitore sono già nel carrello.

COMPLETAMENTO SPECIFICHE

- La pagina iniziale è il form di login
- Tutti i dati anagrafici/di spedizione per un ordine devono essere inseriti
- La quantità di un prodotto nel carrello/ordine non può essere negativa
- Il click del pulsante ORDINA visualizza la sezione ORDINI

AGGIUNTA ALLE SPECIFICHE

- Aggiunta del messaggio personalizzato di benvenuto
- Aggiunta del bottone LOGOUT nella view HOME che, alla pressione da parte dell'utente, effettua il logout e lo riporta alla pagina di login
- Aggiunta di un evento di click ai prodotti visualizzati recentemente che comporta l'apertura della sezione RISULTATI con i dettagli relativi al prodotto selezionato
- Possibilità di selezionare un indirizzo diverso durante la fase di spedizione

REQUIREMENT NON-FUNZIONALI

- L'applicazione, a parte la pagina di login, deve essere contenuta in una sola pagina
- L'interazione dell'utente non deve comportare il refresh completo della pagina
- La validità dell'input deve essere controllata a lato client, oltre che a lato server
- L'autenticazione e autorizzazione devono avvenire in modo sicuro

SOMMARIO

View e Componenti

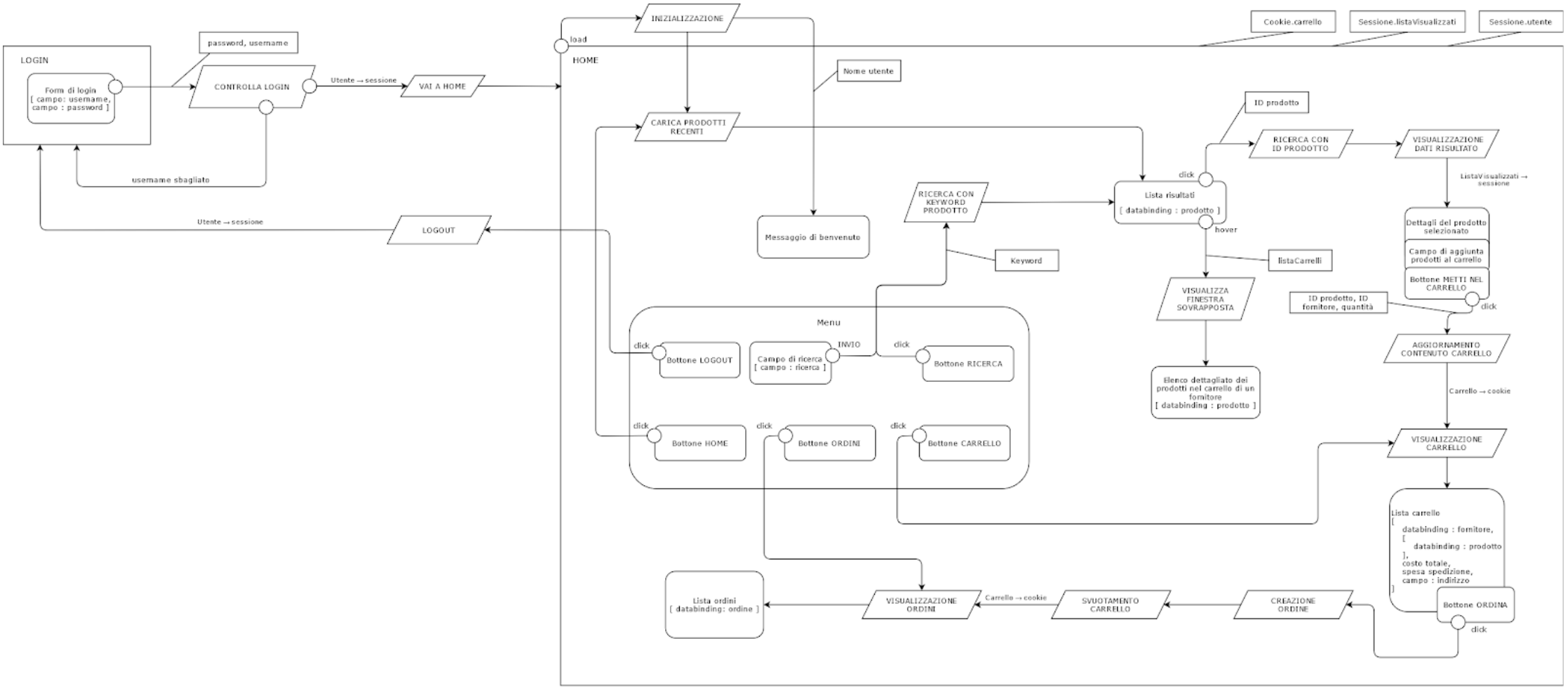
- Login
  - Form di login
- HOME
  - Menu
    - Bottone HOME
    - Bottone CARRELLO
    - Bottone ORDINI
    - Campo di ricerca
    - Bottone RICERCA
    - Bottone LOGOUT
  - Messaggio di benvenuto
  - Lista risultati ed elementi recenti
  - Campo di aggiunta prodotti al carrello
  - Bottone METTI NEL CARRELLO
  - Lista di prodotti nel carrello
  - Campi indirizzo
  - Bottone ORDINA
  - Lista ordini
  - Elenco dettagliato prodotti carrello hover

Eventi e Azioni

- Login
  - Verifica credenziali
- Pressione INVIO nel campo ricerca
  - Ricerca con KEYWORD prodotto
- Click elemento elenco risultati
  - Visualizzazione dati elemento

    - Aggiunta dell’elemento all’elenco visualizzati
- Click METTI NEL CARRELLO
    - Aggiornamento contenuto carrello
    - Visualizzazione carrello
- Click ORDINA
    - Creazione ordine
    - Svuotamento carrello
    - Visualizzazione ordini
- Click HOME
    - Carica prodotti recenti
- Click ORDINI
    - Visualizzazione ordini
- Click CARRELLO
    - Visualizzazione carrello
- Hover numero prodotti dell’elemento all’interno dei risultati
    - Visualizzazione finestra sovrapposta con dettagli prodotti
- Click LOGOUT
    - Logout

APPLICATION DESIGN



EVENTI & AZIONI

Nota

I parametri sottolineati delle richieste **POST** nelle sezioni *EVENTI & AZIONI* e *CONTROLLORI & EVENT HANDLERS* indicano il passaggio di una stringa di testo in formato JSON.

Tabella

	Client Side		**Server Side**	
	EVENTO	AZIONE	EVENTO	AZIONE
1	login → login form → submit	Login	**POST** (username, password)	Controllo delle credenziali
2	home → load	*EVENTO 4*	-	-
3	home → menu → campo di ricerca → INVIO	Ricerca con KEYWORD prodotto	**GET** (keyword)	Estrazione di una lista di prodotti data una keyword
4	home → menu → bottone HOME	Carica prodotti recenti	**POST** (listaVisualizzati)	Estrazione di una lista di 5 prodotti visti di recente dati i loro id (eventualmente utilizzando prodotti casuali dalla categoria libri nel caso ci siano meno di 5 prodotti visualizzati)
5	home → menu → bottone ORDINI	Visualizzazione ORDINI	**GET**	Estrazione degli ordini effettuati dall'utente
6	home → menu → bottone CARRELLO	Visualizzazione CARRELLO	**POST** (listaCarrelli)	Estrazione di un resoconto dei prodotti all'interno dei carrelli dati gli id dei prodotti e dei loro fornitori

7	home → elenco risultati → click elemento	- Visualizzazione dati elemento - Aggiunta all'elenco visualizzati	**GET** (idProdotto)	Estrazione delle informazioni relative ad un prodotto dato il suo id
8	home → elenco risultati → elemento → bottone METTI NEL CARRELLO	- Aggiornamento contenuto carrello - *EVENTO 6*	-	-
9	home → lista prodotti nel carrello → bottone ORDINA	- Creazione ordine - Svuotamento carrello - *EVENTO 5*	**POST** (Carrello, indirizzoSpedizione)	Inserimento del nuovo ordine all'interno del database
10	home → elenco risultati → hover sul numero di prodotti dell'elemento	Visualizzazione finestra sovrapposta con dettagli prodotti	**POST** (listaCarrelli)	Estrazione di un resoconto dei prodotti all'interno di un solo carrello dati gli id dei prodotti e dei loro fornitore
11	home → menu → bottone LOGOUT	Logout	**GET**	Logout

CONTROLLORI & EVENT HANDLERS

	Client Side		**Server Side**	
	EVENTO	CONTROLLORE	EVENTO	CONTROLLORE
1	login → login form → submit	makeCall()	**POST** (username, password)	ControllaLogin
2	home → load	- gestorePagina.init() - *EVENTO 4*	-	-
3	home → menu → campo di ricerca → INVIO	makeCall()	**GET** (keyword)	CercaKeyword

4	home → menu → bottone HOME	makeCall()	**POST** (listaVisualizzati)	CaricaVisualizzati
5	home → menu → bottone ORDINI	makeCall()	**GET**	VisualizzaOrdini
6	home → menu → bottone CARRELLO	makeCall()	**POST** (listaCarrelli)	CaricaCarrello
7	home → elenco risultati → click elemento	- makeCall() - aggiungiVisualizzato()	**GET** (idProdotto)	CercaProdotto
8	home → elenco risultati → elemento → bottone METTI NEL CARRELLO	- aggiungiCookieProdotto() - *EVENTO 6*	-	-
9	home → lista prodotti nel carrello → bottone ORDINA	- makeCall() - cancellaCookieCarrello() - *EVENTO 5*	**POST** (Carrello, indirizzoSpedizione)	AggiungiOrdine
10	home → elenco risultati → hover sul numero di prodotti dell'elemento	makeCall()	**POST** (listaCarrelli)	CaricaCarrello
11	home → menu → bottone LOGOUT	makeCall()	**GET**	Logout

DAO & MODEL OBJECTS (SERVER)

- Model Objects (Beans)
  - Prodotto
  - Fornitore
  - Utente
  - Ordine
  - Carrello
  - Indirizzo
  - Range

- Data Access Object (Classes)
  - ProdottoDAO
    - prendiProdotti(Collection<Integer> presenti, int quantita)

        - prendiProdottiByKeyword(String parolaChiave)
        - prendiOfferteByIdProdotto(int ID)
        - prendiProdottoByIdProdottoFornitore(int idProdotto, int idFornitore)
        - prendiProdottoById(int ID)
        - esisteProdotto(int idProd)
    - FornitoreDAO
        - prendiFornitoreById(int id)
        - esisteFornitore(int idForn)
    - UtenteDAO
        - controllaCredenziali(String email, String psw)
    - OrdineDAO
        - prendiOrdiniByIdUtente(int idUtente)
        - aggiungiOrdine(float totale, int idIndirizzo, int idUtente, int idFornitore, List<Prodotto> prodotti)
        - prendiProssimoId()
    - indirizzoDAO
        - prendiIdIndirizzoByParam(String citta, String via, String cap, int numero)
        - prendiIndirizzoById(int id)
        - aggiungiIndirizzo(String citta, String via, String cap, int numero)

- Controllers (Servlets)
    - ControllaLogin
    - CaricaVisualizzati
    - CaricaCarrello
    - VisualizzaOrdini
    - CercaKeyword
    - CercaProdotto
    - AggiungiOrdine
    - Logout

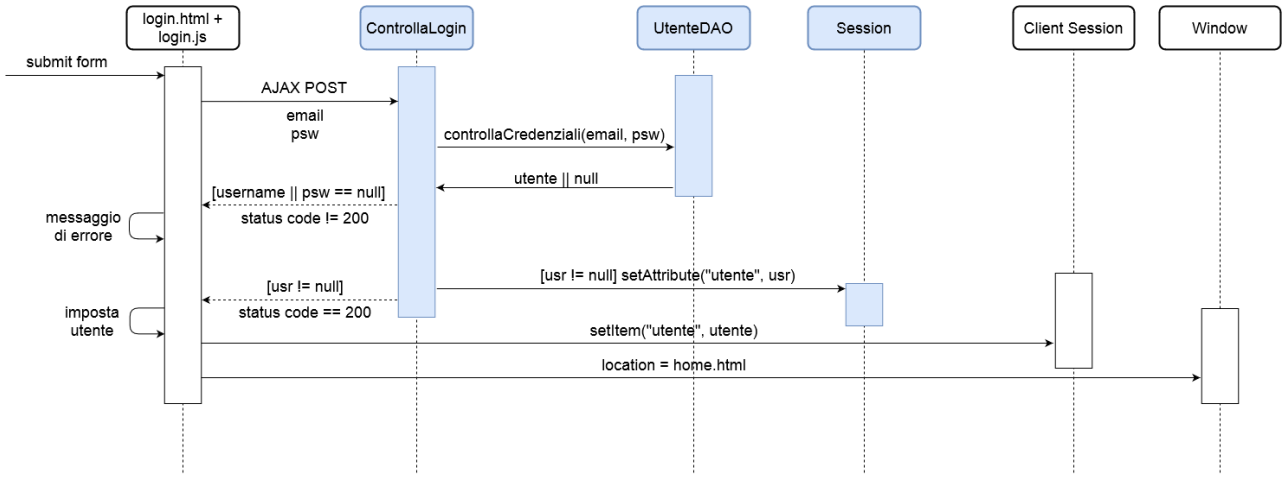
VIEW & VIEW COMPONENT (CLIENT)

- Login
    - Form di login
        - Submit del form, controllo campi e gestione riposte
- Home
    - ListaRisultati, ListaCarrello, ListaOrdini: liste contenenti un determinato tipo di elemento (Prodotto, Carrello o Ordine)
        - carica(): Funzione di caricamento della lista, definita parametricamente durante la costruzione
        - update(): Funzione di aggiornamento della lista, richiama a cascata la funzione update() su tutti gli elementi della lista
        - show(): Mostra la lista tramite l'attributo *hidden* del div associato
        - hide(): Nasconde la lista tramite l'attributo *hidden* del div associato
        - isHidden(): Ritorna *true* se la lista è visibile, *false* altrimenti
    - Prodotto, Carrello, Ordine, Offerta: componenti dinamici utilizzati per rappresentare un determinato elemento all'interno della pagina

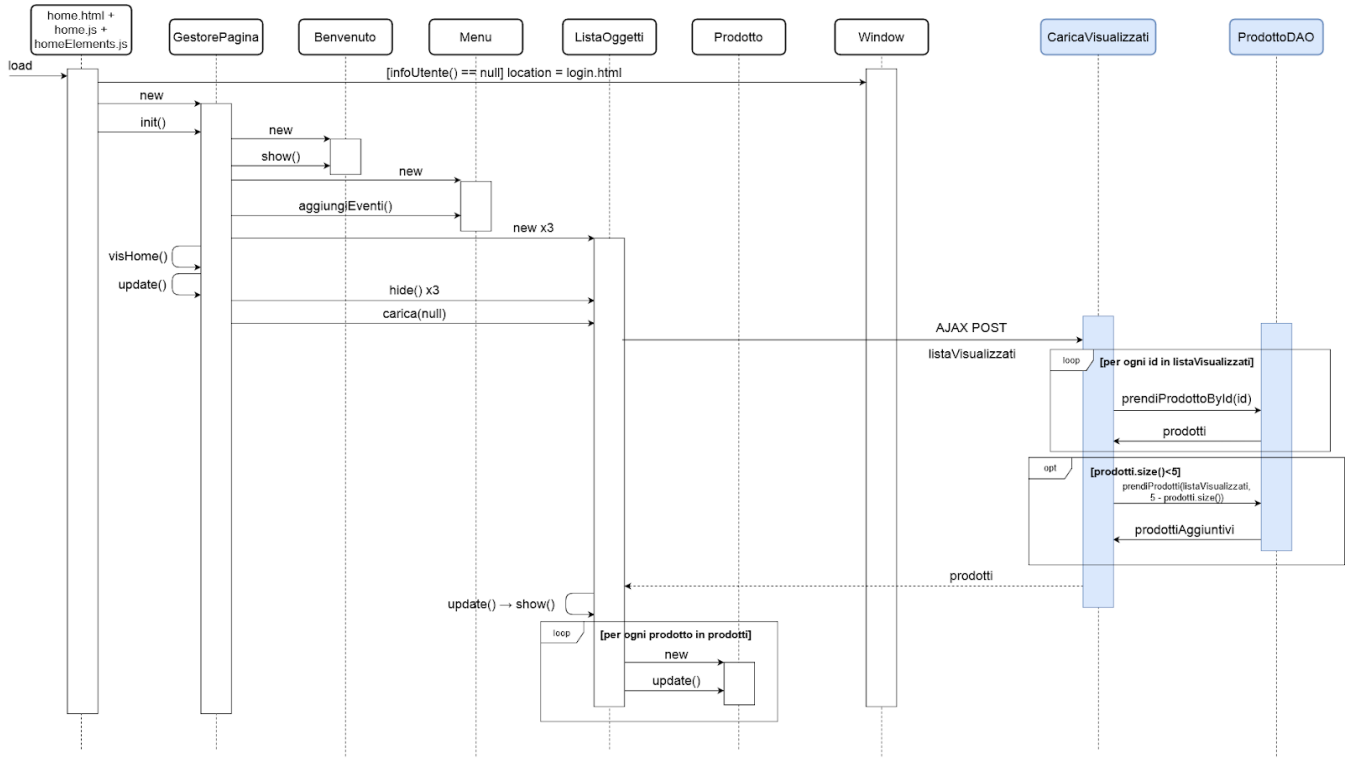
        - update(): Carica le informazioni relative all’elemento creando e relative strutture all’interno della pagina
    - Menu
        - aggiungiEventi(): Associa gli eventi dei componenti contenuti nel menu alle loro relative funzioni necessarie per una corretta navigazione
    - Benvenuto
        - show(): Carica dinamicamente il contenuto per un messaggio personalizzato di benvenuto
    - GestorePagina
        - init(): Funzione richiamata all’interno dell’evento load della pagina, responsabile dell’inizializzazione dei componenti
        - update(): Aggiornamento generico della pagina, nasconde tutti i componenti
        - visHome(), visCarrello(), visOrdini():Aggiornamenti specifici della pagina, richiamano l’aggiornamento generico caricando alcuni elementi specifici in base alla funzione (ListaRisultati, ListaCarrello o ListaOrdini)

SEQUENCE DIAGRAMS

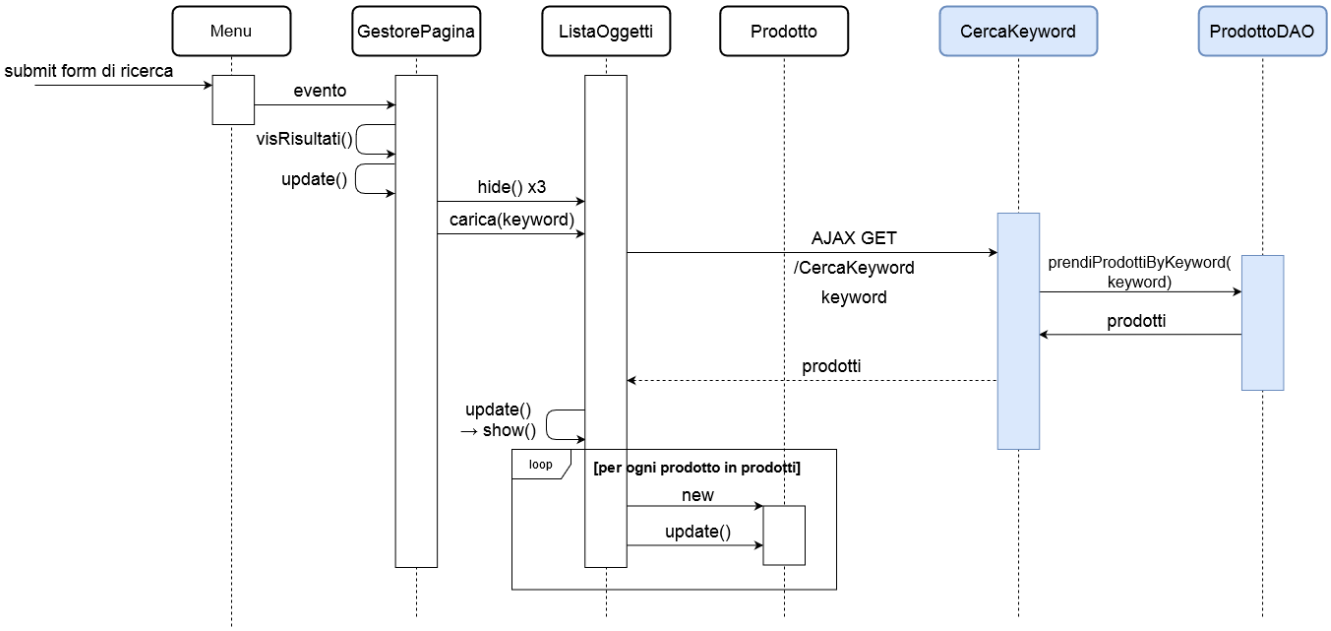
- login



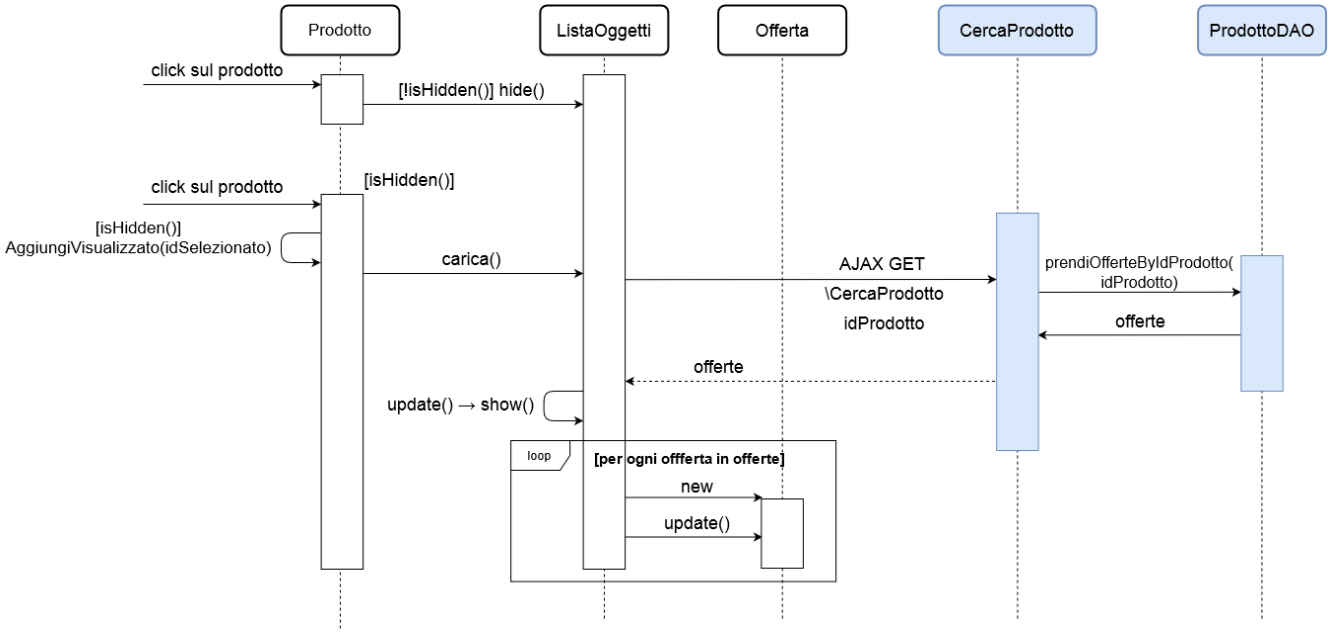
- caricamento home



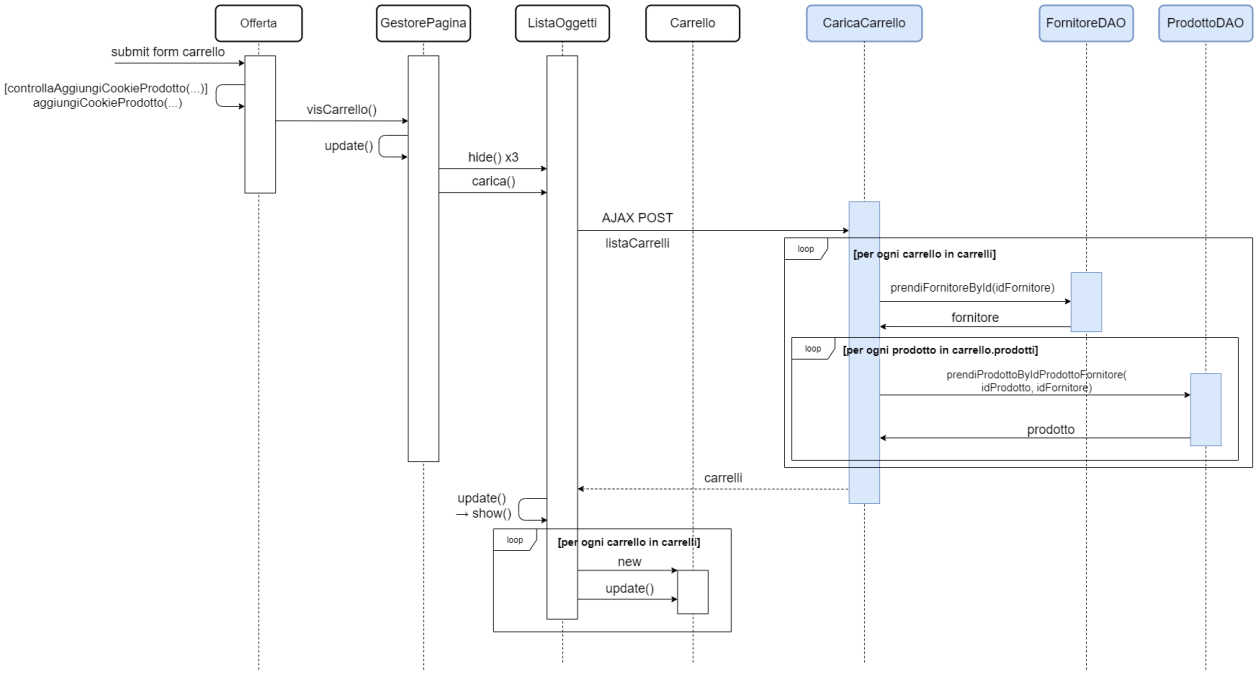
- ricerca con keyword



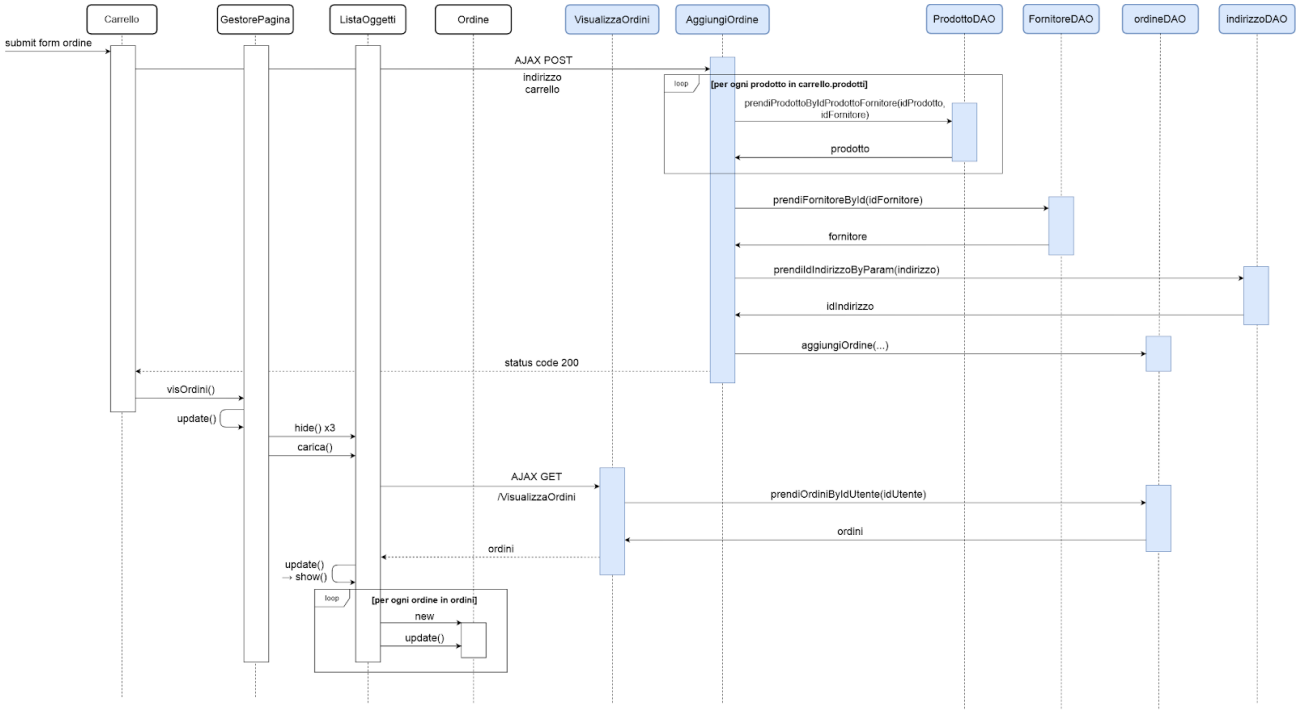
- seleziona prodotto



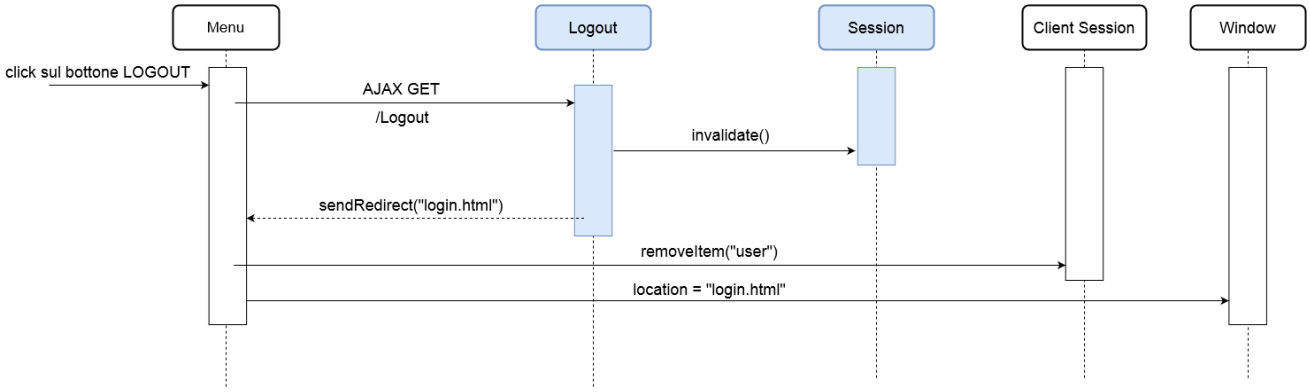
- aggiungi al carrello



- aggiungi ordine



- logout


Menu
Logout
Session
Client Session
Window
click sul bottone LOGOUT
AJAX GET
/Logout
invalidate()
sendRedirect("login.html")
removeItem("user")
location = "login.html"