Gestione carrello con più fornitori e ordine

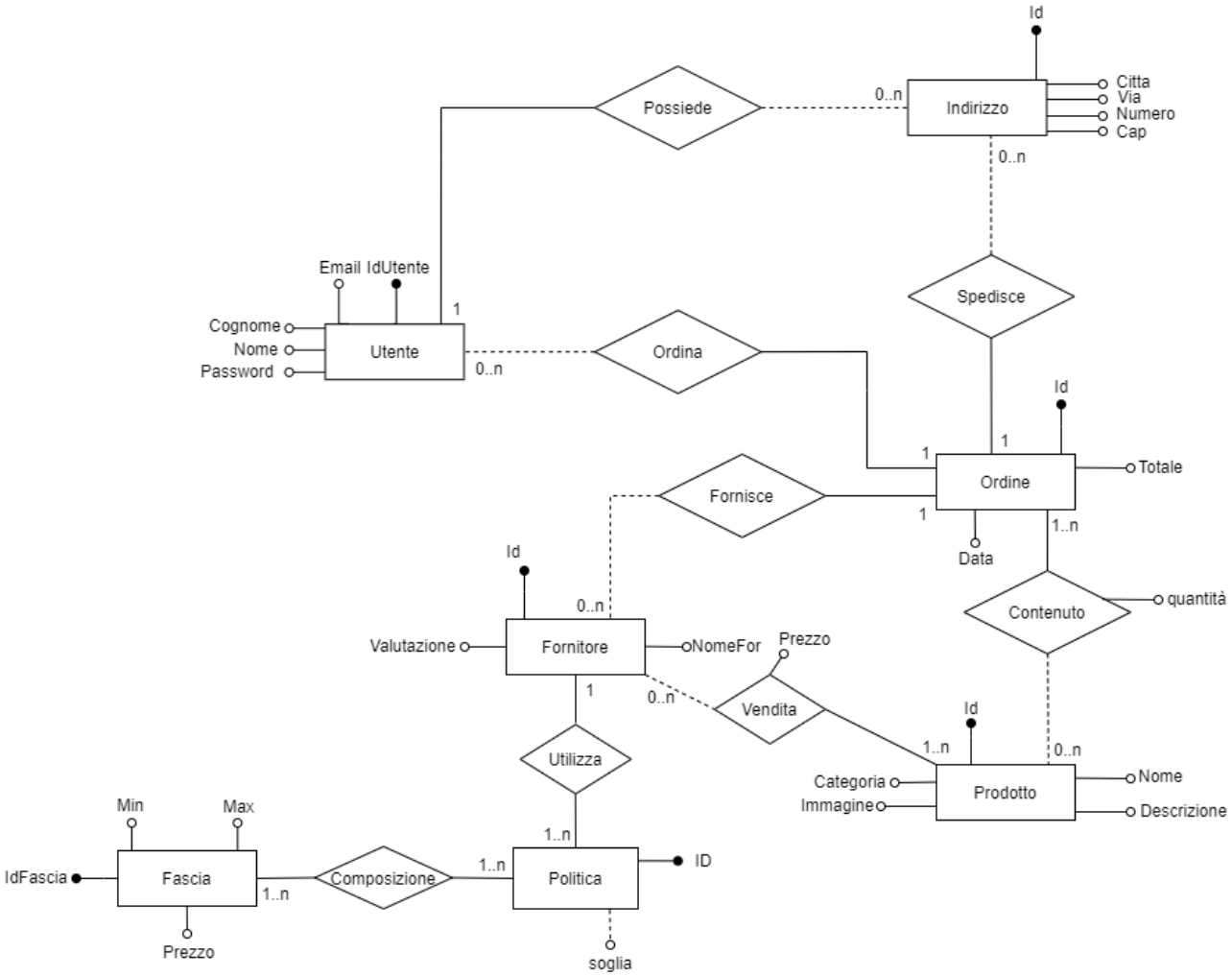
Versione HTML pura

Fumagalli Stefano, Ferro Lara, Rigamonti Davide

- **Analisi dei dati** (**Entities**, **attributes**, **relationships**)

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.

- **Database**



- **Schema**

```
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 )
```

● Application requirements analysis (Pages, view components, events, actions)

Dopo il login (*pagina di 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. 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. 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.

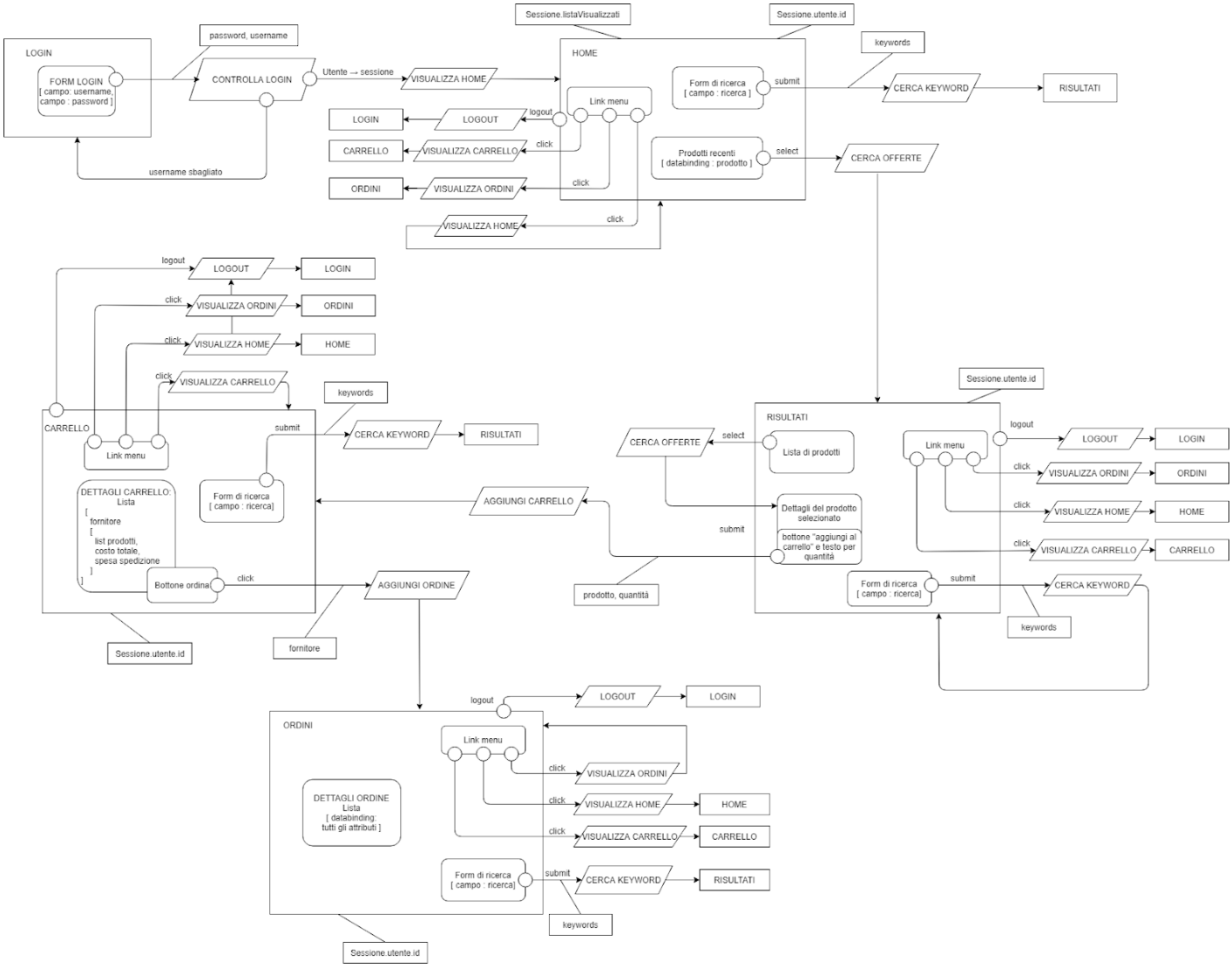
● Completamento specifiche

- La pagina iniziale è il form di login
- Tutti i dati di spedizione per un ordine devono essere inseriti
- La quantità di un prodotto nel carrello/ordine non può essere negativa
- La quantità di prodotti nel carrello/ordine non può superare 999 unità

● Aggiunta specifiche

o Aggiunta del messaggio personalizzato di benvenuto
o Aggiunta del bottone di logout

● Application design



Components

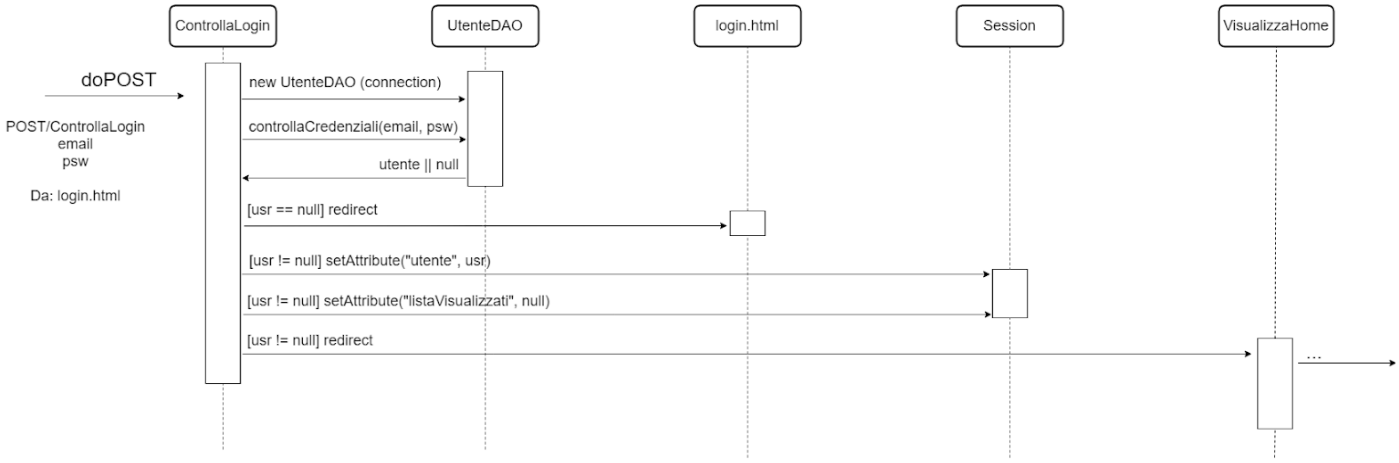
- **Model Objects (Beans)**
  - Prodotto
  - Fornitore
  - Utente
  - Ordine
  - Carrello
  - Indirizzo
  - Range

- **Data Access Object (Classes)**
  - ProdottoDAO
    - prendiProdotti(Queue<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)

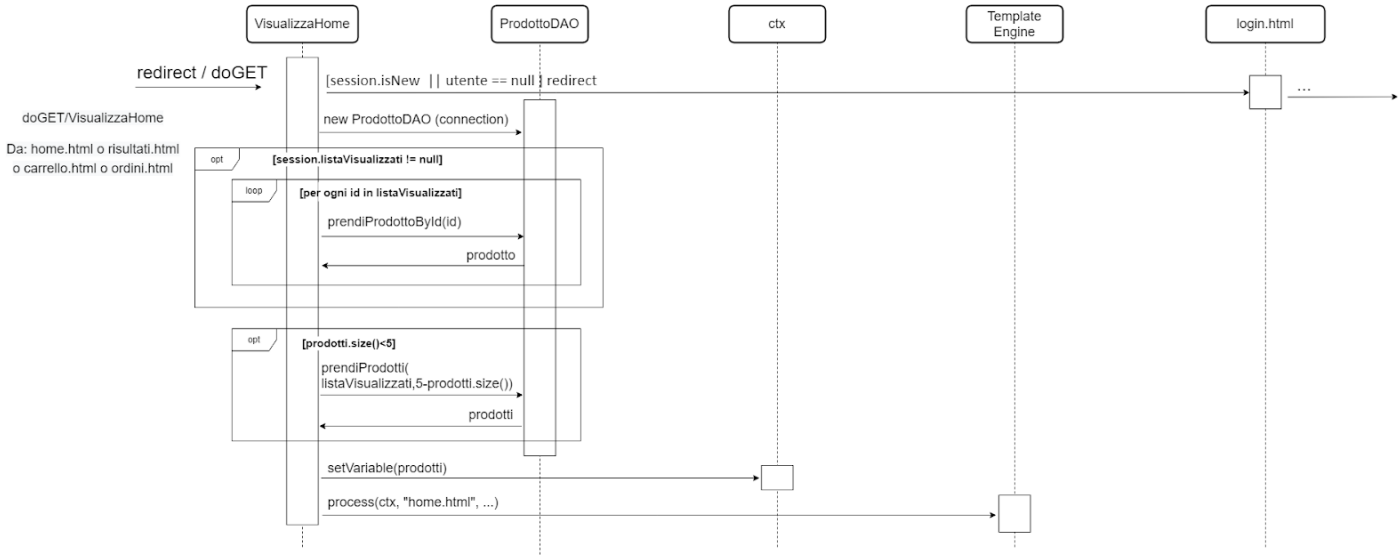
- **Controllers (Servlets)**
  - ControllaLogin
  - VisualizzaHome
  - VisualizzaOrdine
  - VisualizzaCarrello
  - AggiungiCarrello
  - AggiungiOrdine
  - CercaKeyword
  - CercaProdotto
  - CercaOfferte
  - Logout

- **Pages (Templates)**
  - **o** Login
  - **o** Home
  - **o** Carrello
  - **o** Fragment
  - **o** Risultati
  - **o** Ordini

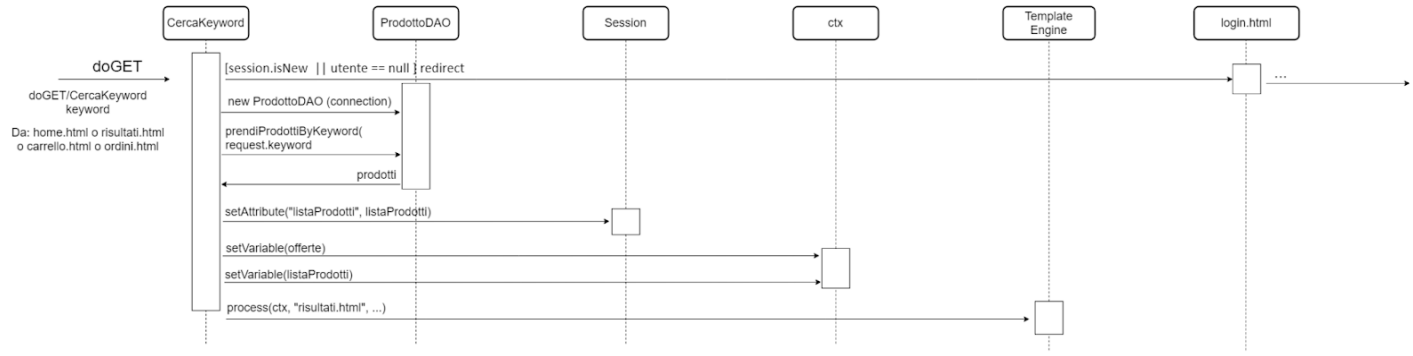
- **Sequence Diagram**
  - Login



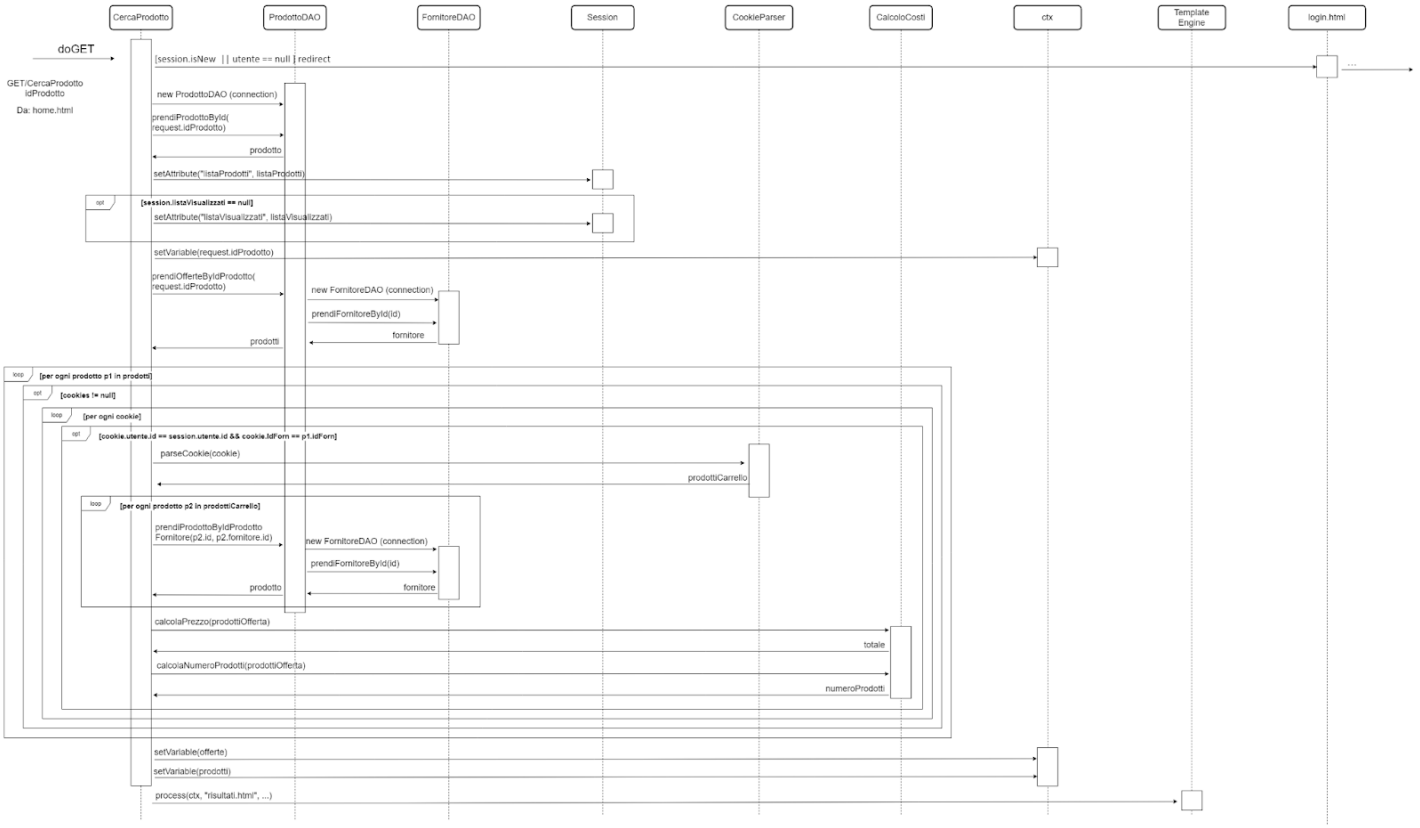
  - Visualizza Home



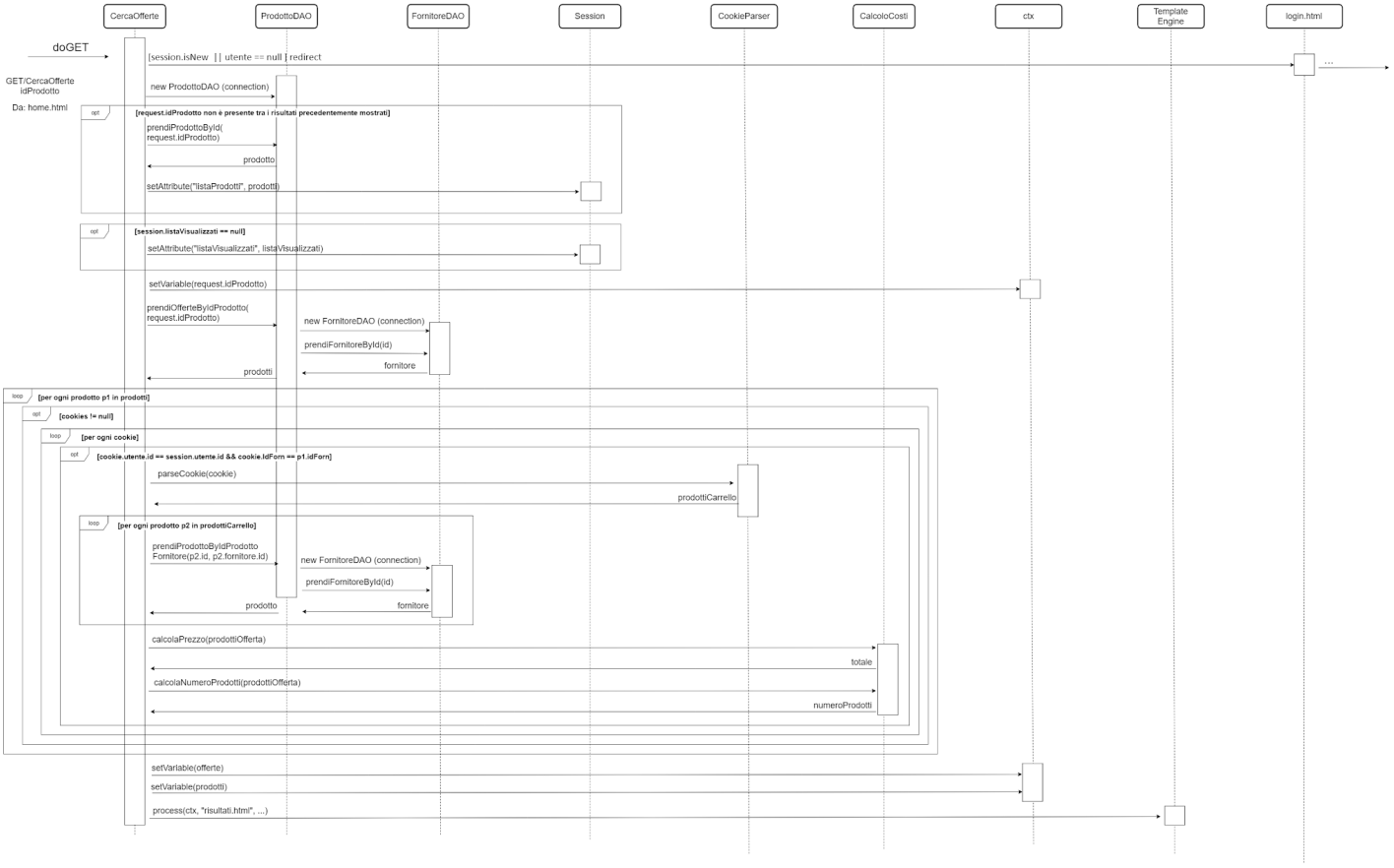
- Cerca con keyword



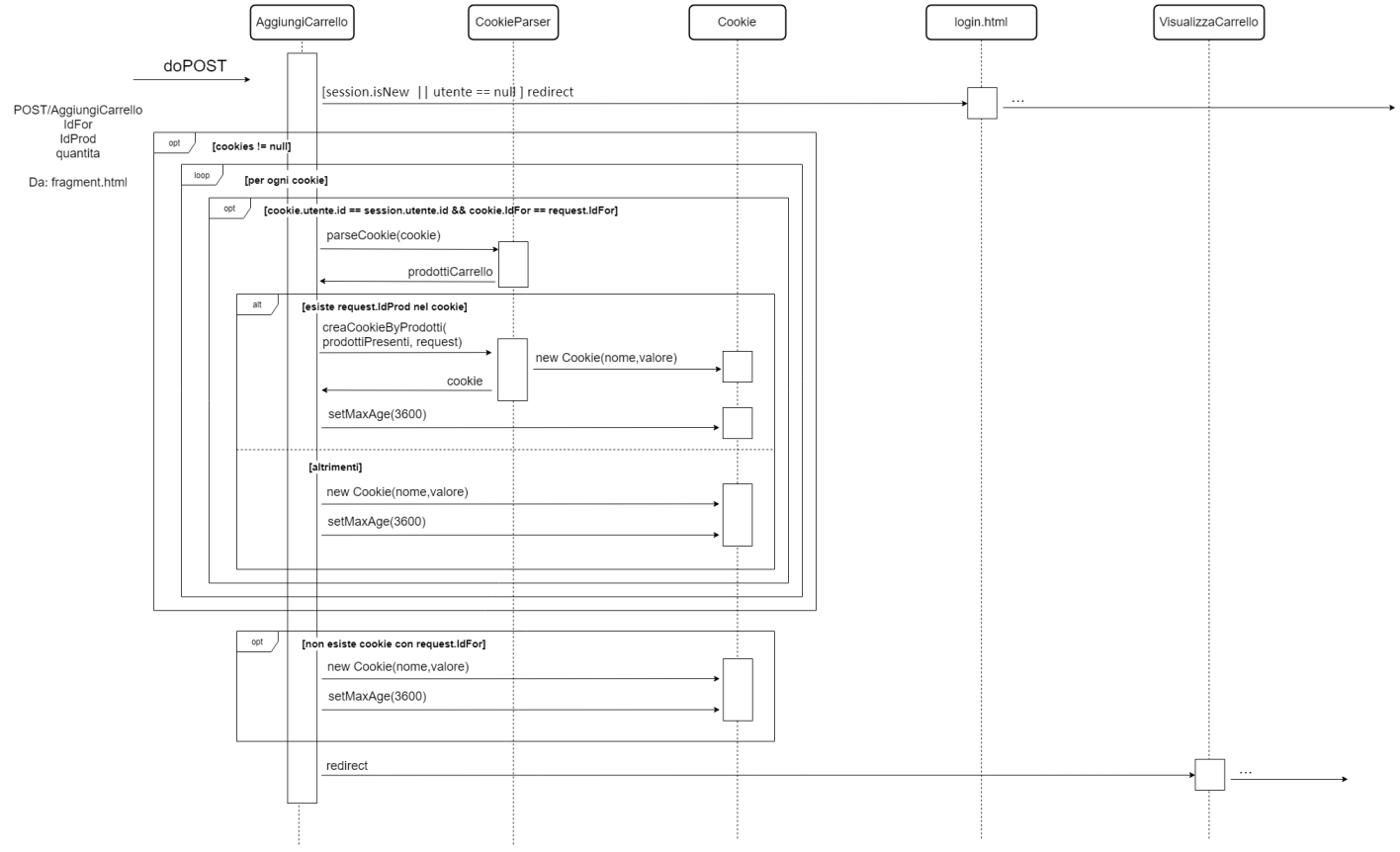
- Cerca prodotto



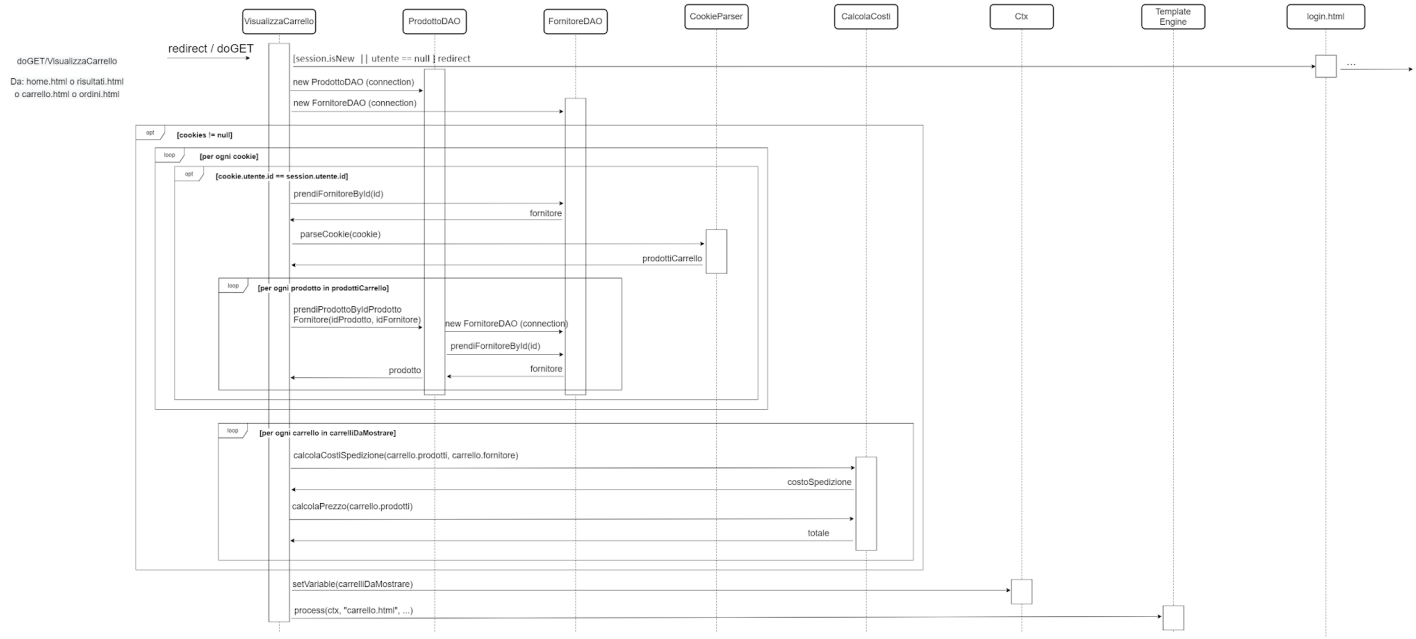
- Cerca offerte



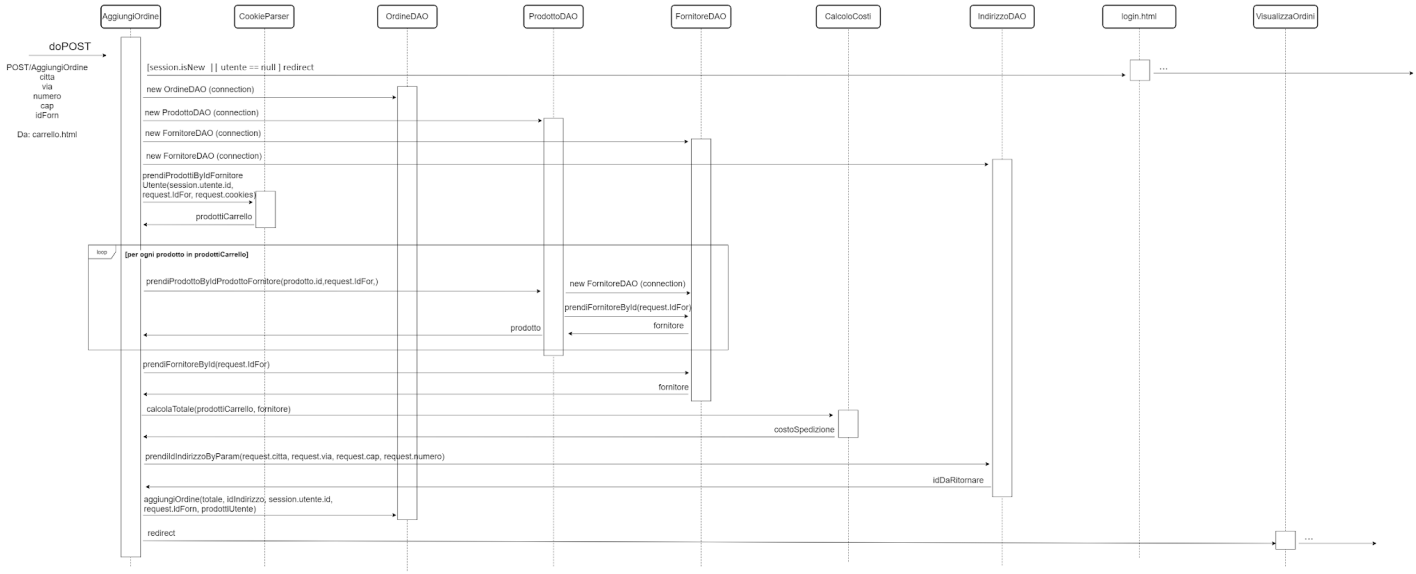
- Aggiungi al carrello



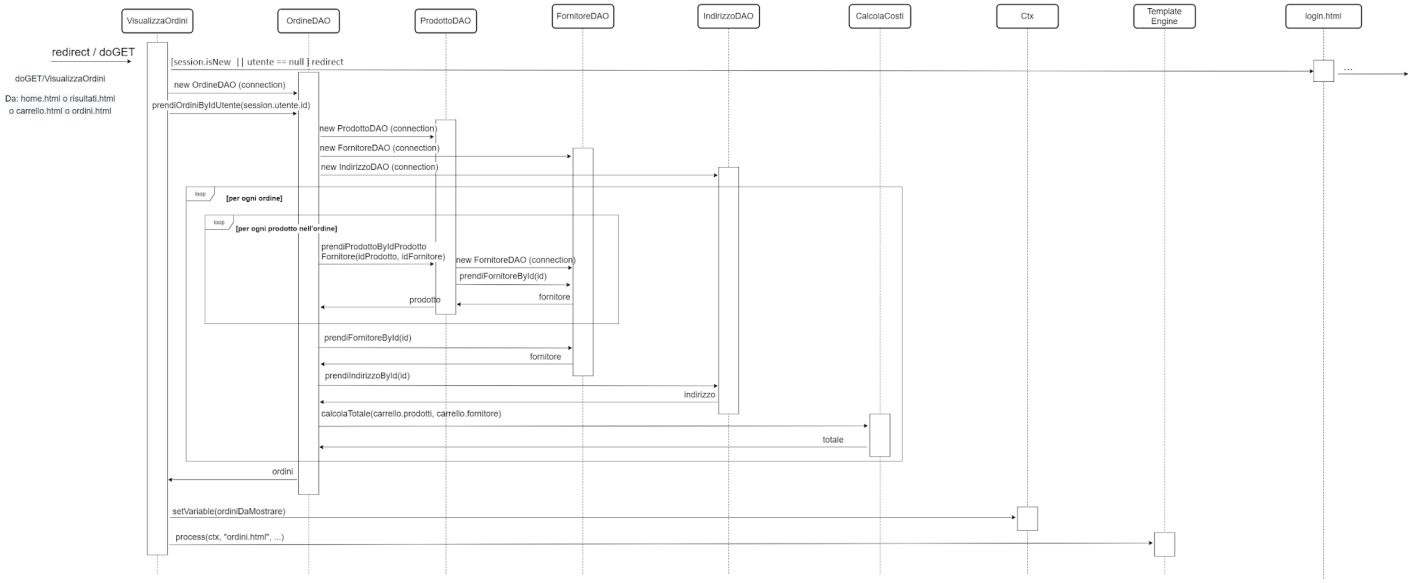
- Visualizza Carrello



- Aggiungi Ordine



- Visualizza ordini



- Logout