30 OCTOBER 2023

PROGETTO DI TECNOLOGIE INFORMATICHE PER IL WEB

POLITECNICO DI MILANO

MATTEO BRISCINI
MATRICOLA: 960708
Matteo.briscini@mail.polimi.it

INDICE

1 REQUISITI

1.1 Specifica

Un'applicazione permette all'utente (ad esempio il responsabile dei servizi ambientali di una regione) di gestire una collezione di immagini satellitari e una tassonomia di classificazione utile per etichettare immagini allo scopo di consentire la ricerca per categoria. Dopo il login, l'utente accede a una pagina HOME in cui compare un albero gerarchico di categorie. Le categorie non dipendono dall'utente e sono in comune tra tutti gli utenti. Un esempio di un ramo dell'albero è il seguente:

- 9 Materiali solidi>>copia
- 91 Materiali inerti>>copia
- 911 Inerti da edilizia >>copia
- 9111 Amianto >>copia
- 91111 Amianto in lastre >>copia
- 91112 Amianto in frammenti >>copia
- 9112 Materiali cementizi >>copia
- 912 Inerti ceramici >>copia
- 9121 Piastrelle >>copia
- 9122 Sanitari >>copia

L'utente può inserire una nuova categoria nell'albero. Per fare ciò usa una form nella pagina HOME in cui specifica il nome della nuova categoria e sceglie la categoria padre. L'invio della nuova categoria comporta l'aggiornamento dell'albero: la nuova categoria è appesa alla categoria padre come ultimo sottoelemento. Alla nuova categoria viene assegnato un codice numerico che ne riflette la posizione (ad esempio, la nuova categoria "Amianto in tubi", figlia della categoria "9111 Amianto" assume il codice 91113). Dopo la creazione di una categoria, la pagina HOME mostra l'albero aggiornato. Per velocizzare la costruzione della tassonomia l'utente può copiare un intero sottoalbero in una data posizione: per fare ciò clicca sul link "copia" associato alla categoria radice del sottoalbero da copiare. A seguito di tale azione l'applicazione mostra, sempre nella HOME page, l'albero con evidenziato il sottoalbero da copiare: tutte le altre categorie hanno un link "copia qui". Ad esempio, a seguito del click sul link "copia" associato alla categoria "9111 Amianto" l'applicazione visualizza l'albero come segue.

- 9 Materiali solidi>>copia qui
- 91 Materiali inerti>>copia qui
- 911 Inerti da edilizia >>copia qui
- 9111 Amianto
- 91111 Amianto in lastre
- 91112 Amianto in frammenti
- 9112 Materiali cementizi >>copia qui
- 912 Inerti ceramici >>copia qui
- 9121 Piastrelle >>copia qui
- 9122 Sanitari >>copia qui

La selezione di un link “copia qui” comporta l’inserimento di una copia del sottoalbero come ultimo figlio della categoria destinazione. Ad esempio, la selezione del link “copia qui” della categoria “9 Materiali solidi” comporta la seguente modifica dell’albero:

- 9 Materiali solidi>>copia
- 91 Materiali inerti>>copia
- 911 Inerti da edilizia >>copia
- 9111 Amianto >>copia
- 91111 Amianto in lastre >>copia
- 91112 Amianto in frammenti >>copia
- 9112 Materiali cementizi >>copia
- 912 Inerti ceramici >>copia
- 9121 Piastrelle >>copia
- 9122 Sanitari >>copia
- 92 Materiali ferrosi >>copia
- 93 Amianto >>copia
- 931 Amianto in lastre >>copia
- 932 Amianto in frammenti >>copia

Le modifiche effettuate da un utente e salvate nella base di dati diventano visibili agli altri utenti. Per semplicità si ipotizzi che per ogni categoria il numero massimo di sottocategorie sia 9, numerate da 1 a 9. In questo caso l’operazione di copia deve controllare che lo spostamento non determini un numero di sottocategorie superiore a 9. Si preveda anche un link “copia qui” non associato a un nodo della tassonomia che permette di copiare un sotto-albero al primo livello della tassonomia (se non esistono già 9 nodi al primo livello della tassonomia).

1.2 Aggiunte alla specifica (versione HTML)

- Aggiunte le funzioni “*taglia*” ed “*elimina*”.
- Aggiunta una barra di ricerca per parametri multipli.
- Tramite il DB è possibile definire un livello per l’utente (0= guest, 1 = admin), l’interfaccia utente cambia a seconda del livello dell’utente mostrando o nascondendo pulsanti per funzioni per le quali ha l’autorizzazione o meno.

1.3 Estensione della specifica per la versione con JavaScript

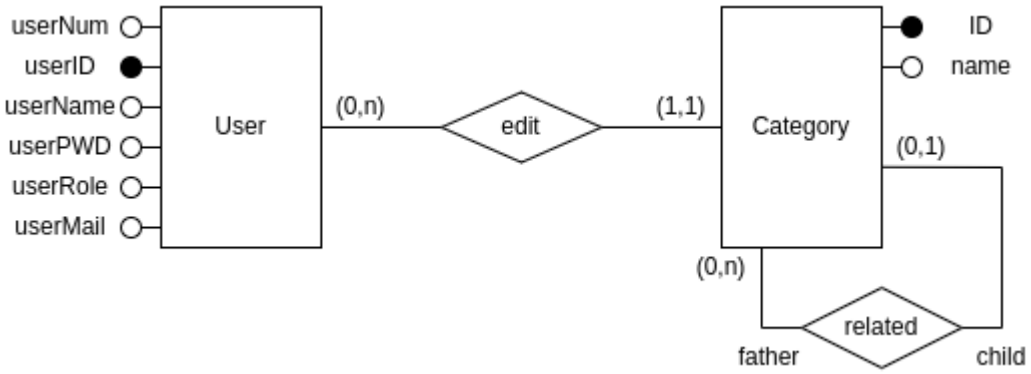
Si realizzi un’applicazione client server web che estende e/o modifica le specifiche precedenti come segue:

- 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.
- La funzione di copia di un sottoalbero è realizzata mediante drag & drop. A seguito del drop della radice del sottoalbero da copiare compare una finestra di dialogo con cui l’utente può confermare o cancellare la copia. La conferma produce l’aggiornamento solo a lato client dell’albero. La cancellazione riconduce allo stato precedente al drag & drop. A seguito della conferma compare un bottone SALVA che consente il salvataggio a lato server della tassonomia modificata.
- L’utente può cliccare sul nome di una categoria. A seguito di tale evento compare al posto del nome un campo di input contente la stringa del nome modificabile. L’evento di perdita del focus del campo di input produce il salvataggio nel database del nome modificato della categoria.

2 DESIGN DELLA BASE DI DATI

2.1 Schema ER

Dall'analisi delle specifiche si è ricavato il seguente schema relazionale:



L'utilizzo dello userID come chiave primaria nella tabella User consente di rinominare l'account dell'utente.

L'albero di tassonomia è memorizzato mediante l'auto relazione padre-figlio; un padre può avere più figli ma non è previsto il caso di ereditarietà multipla.

La relazione *edit* consente di verificare, a livello di DB, che l'utente che ha creato una nuova categoria sia in possesso di un *userRole* di livello sufficiente per eseguire tale operazione (1 o superiore).

2.2 Codice per la Creazione delle tabelle SQL

2.2.1 User

Si riporta sotto il codice di creazione della tabella User:

```
User CREATE TABLE IF NOT EXIST `usersCredentials` (
            `userNum` int NOT NULL AUTO_INCREMENT,
            `userID` varchar(45) NOT NULL,
            `username` varchar(45) NOT NULL,
            `userPWD` varchar(45) NOT NULL,
            `userRole` int NOT NULL,
            `userMail` varchar(45) NOT NULL,
            PRIMARY KEY (`userID`),
            UNIQUE KEY `userID_UNIQUE` (`userID`),
            UNIQUE KEY `username_UNIQUE` (`username`),
            UNIQUE KEY `userMail_UNIQUE` (`userMail`),
            UNIQUE KEY `userNum_UNIQUE` (`userNum`)
)
```

2.2.2 Category

Si riporta di seguito il codice di creazione della tabella Categories:

```
CREATE TABLE IF NOT EXIST `categories` (
            `ID` bigint NOT NULL,
            `name` varchar(45) NOT NULL,
            `fatherID`bigint DEFAULT NULL,
            `lastModifier` varchar(45) NOT NULL,
            PRIMARY KEY (`ID`),
            UNIQUE KEY `ID_UNIQUE` (`ID`),
            KEY `lastModifier_idx` (`lastModifier`),
             CONSTRAINT `creatorKey` FOREIGN KEY (`lastModifier`) REFERENCES `usersCredentials`
(`userID`)
)
```

2.3 Trigger

In aggiunta ai vincoli sui singoli attributi e a quelli dovuti alla presenza di foreign key, sono stati pensati alcuni triggeri (di tipo AFTER INSERT, AFTER UPDATE o AFTER DELETE).

I trigger, sotto riportati, eseguono query per verificare che non vengano salvati nel DB elementi inconsistenti o che non rispettino i vincoli definiti. Nel caso siano trovati elementi non congrui i trigger lanciano un'eccezione SQL.

2.3.1 Not allowed user

Il trigger, di seguito riportato, verifica all'aggiunta di una nuova categoria che l'utente che effettua l'operazione abbia le autorizzazioni necessarie. Questo filtro è duplicato identico anche nella casistica AFTER UPDATE (non riportato per sintesi).

```
DELIMITER //
CREATE TRIGGER add_new_category_by_not_allowed_user
AFTER INSERT ON categories FOR EACH ROW BEGIN
IF EXISTS(
            SELECT * FROM categories JOIN usersCredentials ON lastModifier = userID
            WHERE userRole<1
)THEN
   SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'you do not have autorization for this
operation';
END IF;
END//
DELIMITER ;
```

2.3.2 Consistenza della relazione related (AFTER INSERT)

Questo trigger verifica, dopo l'inserimento di una nuova categoria, che tutti i codici numerici "fatherID" siano relativi a categorie realmente salvate nella base di dati.

```
DELIMITER //
CREATE TRIGGER add_new_categories_with_not_existing_father
AFTER INSERT ON categories FOR EACH ROW BEGIN
IF EXISTS(
                        SELECT * FROM categories
                        WHERE fatherID != 0 AND fatherID NOT IN (SELECT ID FROM categories)
)THEN SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'it is not possible to add a child to non
existent father';
END IF;
END//
DELIMITER;
```

2.3.3 Consistenza della relazione related (AFTER UPDATE)

Questo trigger verifica, dopo l'aggiornamento di una categoria esistente, che tutti i codici numerici "fatherID" siano relativi a categorie realmente salvate nella base di dati.

```
DELIMITER //
CREATE TRIGGER add_new_categories_with_not_existing_father_after_update
AFTER UPDATE ON categories FOR EACH ROW BEGIN
IF EXISTS(
                        SELECT * FROM categories
                        WHERE (@trigger_disable IS NULL OR @trigger_disable!=1) AND fatherID != 0
AND fatherID NOT IN (SELECT ID FROM categories)
)THEN
   SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'it is not possible to add a child to non existent
father';
END IF;
END//
DELIMITER ;
```

NOTE: *trigger_disable* è una variabile di sessione, viene posta a 1 per disabilitare i trigger in caso di update o delete multiple. Il trigger verrà quindi riabilitato prima dell'esecuzione dell'ultima query per verificare lo stato del database.

2.3.4 Consistenza della relazione related (AFTER DELETE)

Questo trigger verifica, dopo l'eliminazione di una categoria esistente, che tutti i codici numerici "fatherID" siano relativi a categorie realmente salvate nella base di dati.

```
DELIMITER //
CREATE TRIGGER add_new_categories_with_not_existing_father_after_delete
AFTER DELETE ON categories FOR EACH ROW BEGIN
IF EXISTS(
                        SELECT * FROM categories
                        WHERE (@trigger_disable IS NULL OR @trigger_disable!=1)AND fatherID != 0
AND fatherID NOT IN (SELECT ID FROM categories)
)THEN
   SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'non puoi rimuovere una categoria se prima non
rimuovi i suoi figli';
END IF;
END//
DELIMITER ;
```

2.4 Query significative

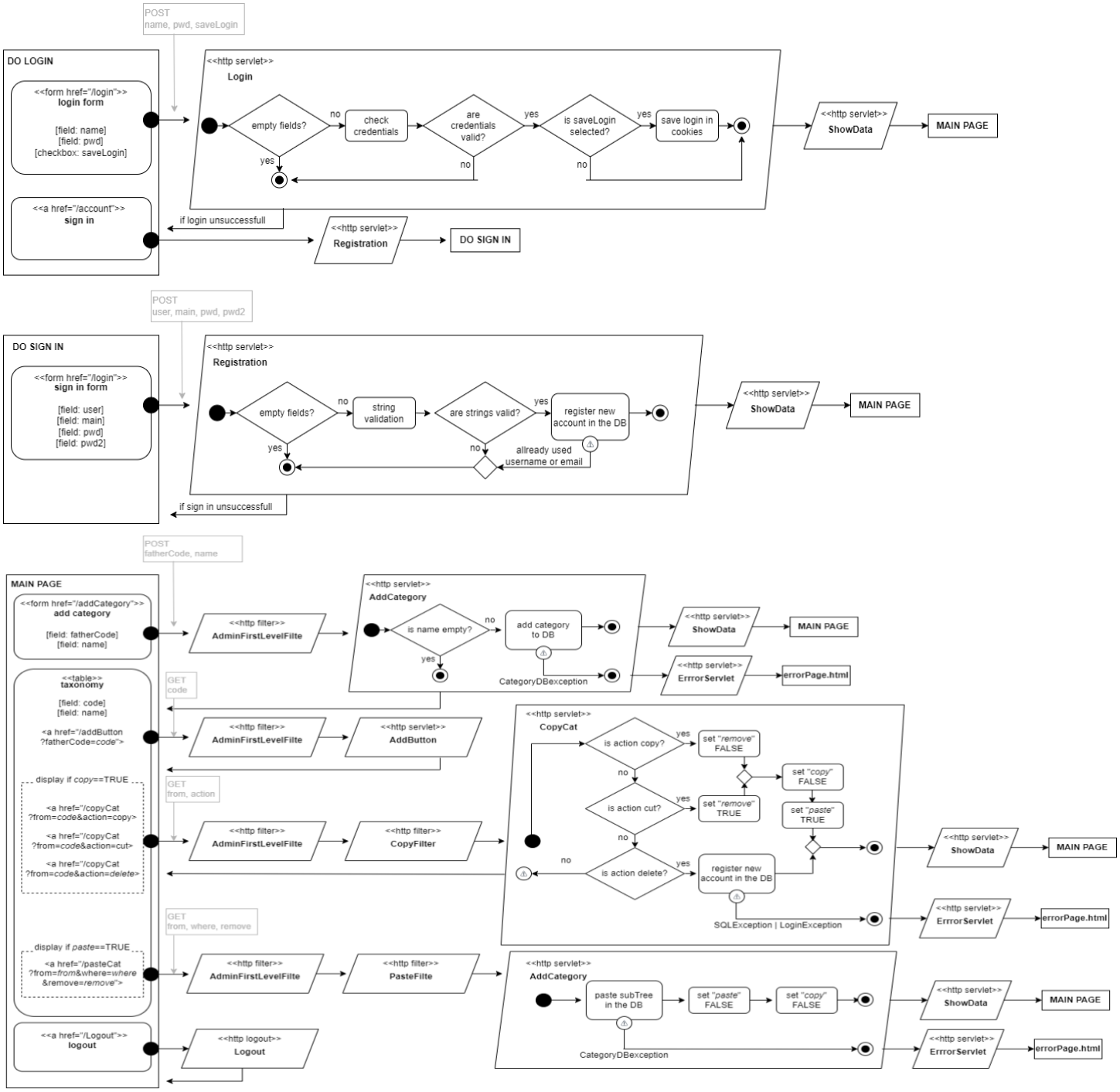
Sotto è riportata la query per selezionare tutte le categorie dominate da un'altra categoria N.

Questa query è utilizzata in più metodi, utili ad implementare funzioni quali "*Paste*", "*Search*" e "*Select*" (anche se parzialmente modificata per far fronte alle diverse necessità implementative).

```
SELECT  * FROM categories AS t, lastSon AS l
WHERE t.ID IN (
        WITH RECURSIVE subTree (fatherID, ID) AS (
                SELECT fatherID, ID FROM categories
                WHERE fatherID = N
                UNION ALL
                SELECT c.fatherID, c.ID FROM (subTree AS f) JOIN (categories AS c) ON f.ID =
c.fatherID
        )
        SELECT c.ID FROM subTree AS f JOIN categories AS c ON f.ID = c.ID
        UNION ALL
        SELECT ID from categories WHERE ID= N
)
```

3 DESIGN APPLICATIVO DELLA VERSIONE HTML PURO

3.1 Struttura delle servlet



3.2 Controllo dell'accesso

A tutte le servlet (escluse quelle di Login, Logout e Registration) è anteposta una servlet di tipo filter, denominato "Login filter". Questo verifica la presenza di un attributo User nella sessione, a confermare il successo dell'azione precedente di login.

L'attributo di tipo User viene inoltre utilizzato per conservare i dati dell'utente, utilizzati quando si effettuano Query che agiscono sulla tabella "*Categories*", permettendo così il funzionamento del filtro SQL "*add_new_category_by_not_allowed_user*".

4 COMPONENTI PER LA VERSIONE SOLO HTML

4.1 Interazione con il DB

Per gestire le connessioni con la base di dati è stata implementata una classe “ConnectionsHandler”, che permette di salvare connessioni attive evitando di aprire una nuova connessione per ogni accesso al DB.

Nella classe sono implementati due metodi:

- takeConnection: che permette alle DAO di utilizzare una delle connessioni disponibili o crearne una nuova qualora non ve ne siano;
- releaseConnection: una volta che la DAO ha terminato di utilizzare la connessione con il DB la restituisce, rendendola disponibile per usi futuri.

Sul data base è stata ridotta la durata della vita di una connessione ad un’ora.

Quando una DAO richiede di accedere a una connessione, se vi sono connessioni scadute, queste vengono rimosse dal ConnectionsHandler.

4.2 Data beams

- Category
- UpdateCategory
- User

4.3 Funzionalità DAO

4.3.1 CategoriesDAO

- getCategories(from [optional], context)
- addCategory(fatherCode, name, creator, context)
- removeSubTree(fatherCode, context)
- pasteSubTree(from, where, creator, remove, context)
- addNewHead(name, creator, context)
- searchCategory(search, context)

4.3.2 UserDAO

- checkCredentials(pwd, user, context)
- checkUserID(userID, user, context)
- getUserRole(userID, context)
- signInMandatoryData(user, email, pwd, userRole, context)

4.4 Servlets

4.4.1 Controlli

- AddButton
- AddCategory
- CopySubTree
- Login
- Logout

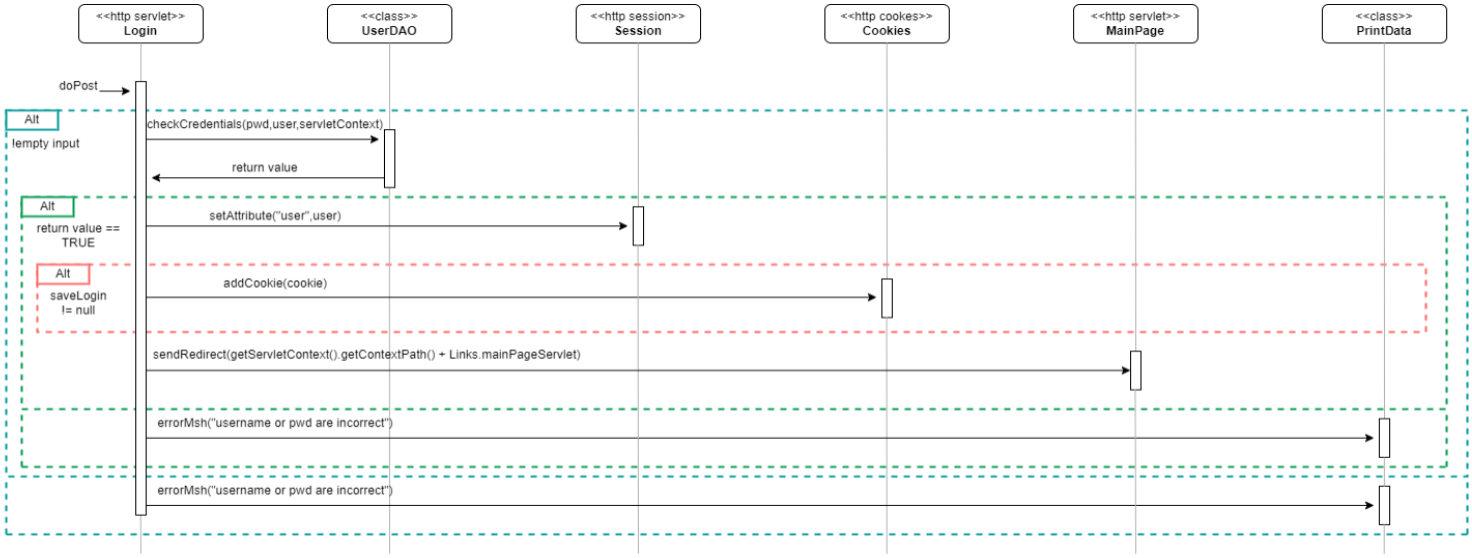
- PasteSubTree
- Registration
- SearchCategory
- ShowData
- ErrorServlet

4.4.2 Filtri

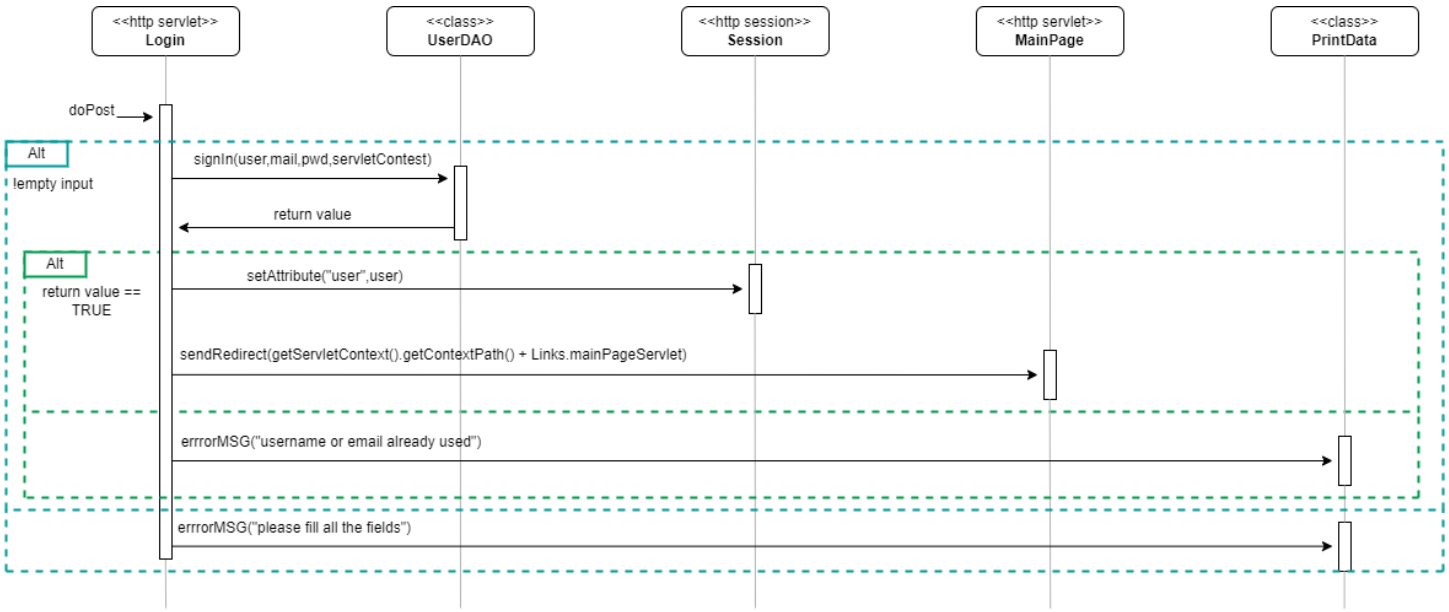
- AdminFirstLevelFilter
- CopyFilter
- LoginFilter
- PasteFilter

5 SEQUENCE DIAGRAMS PER LA VERSIONE HTML

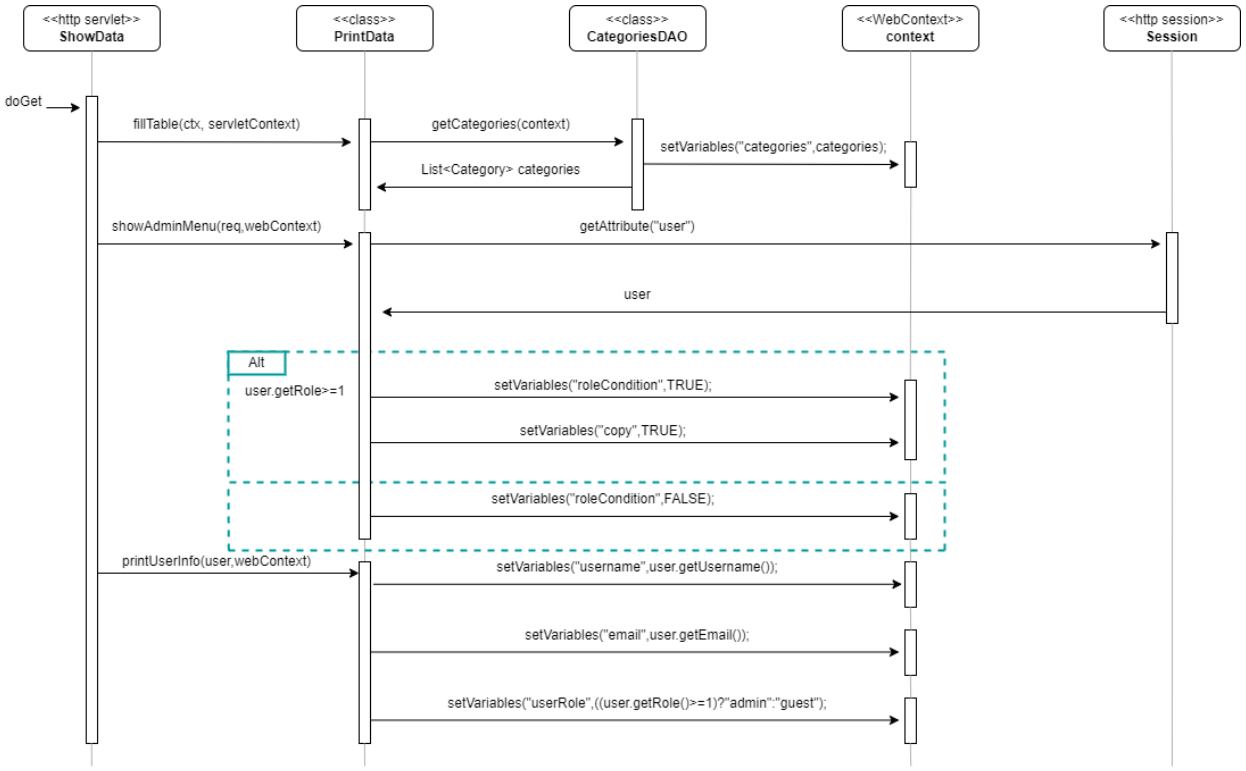
5.0.1 Un utente effettua il login



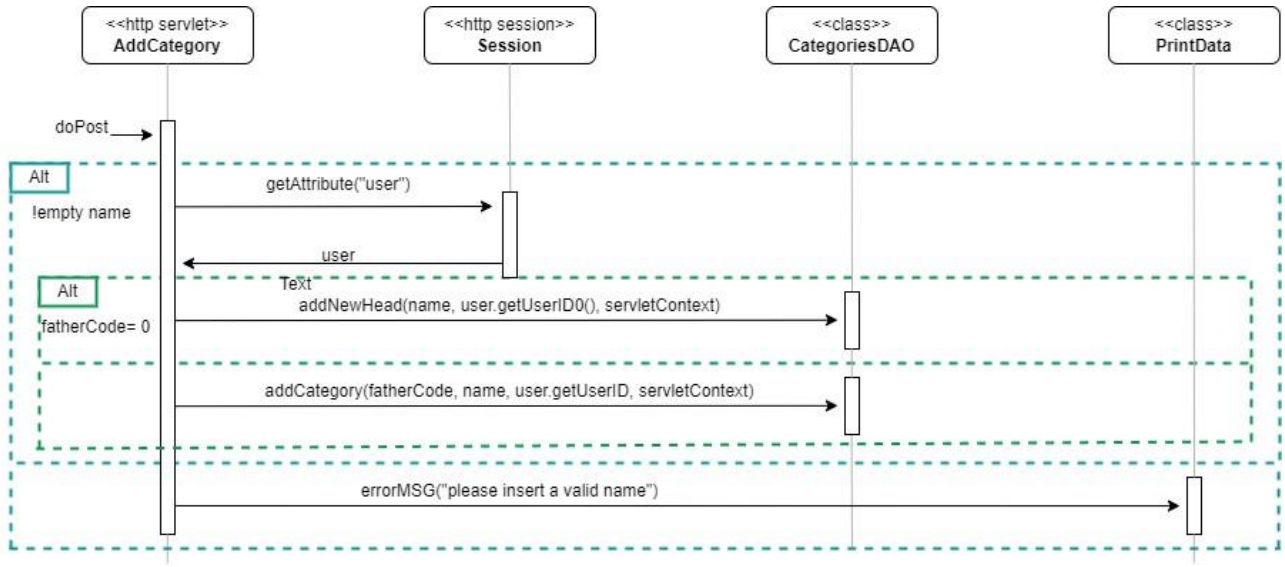
5.0.2 Un nuovo utente si registra



5.0.3 Un utente richiede la pagina principale

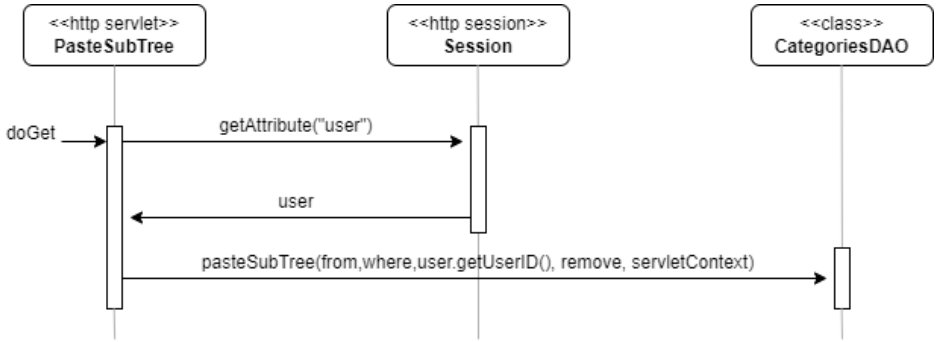


5.0.4 Un utente aggiunge una nuova categoria

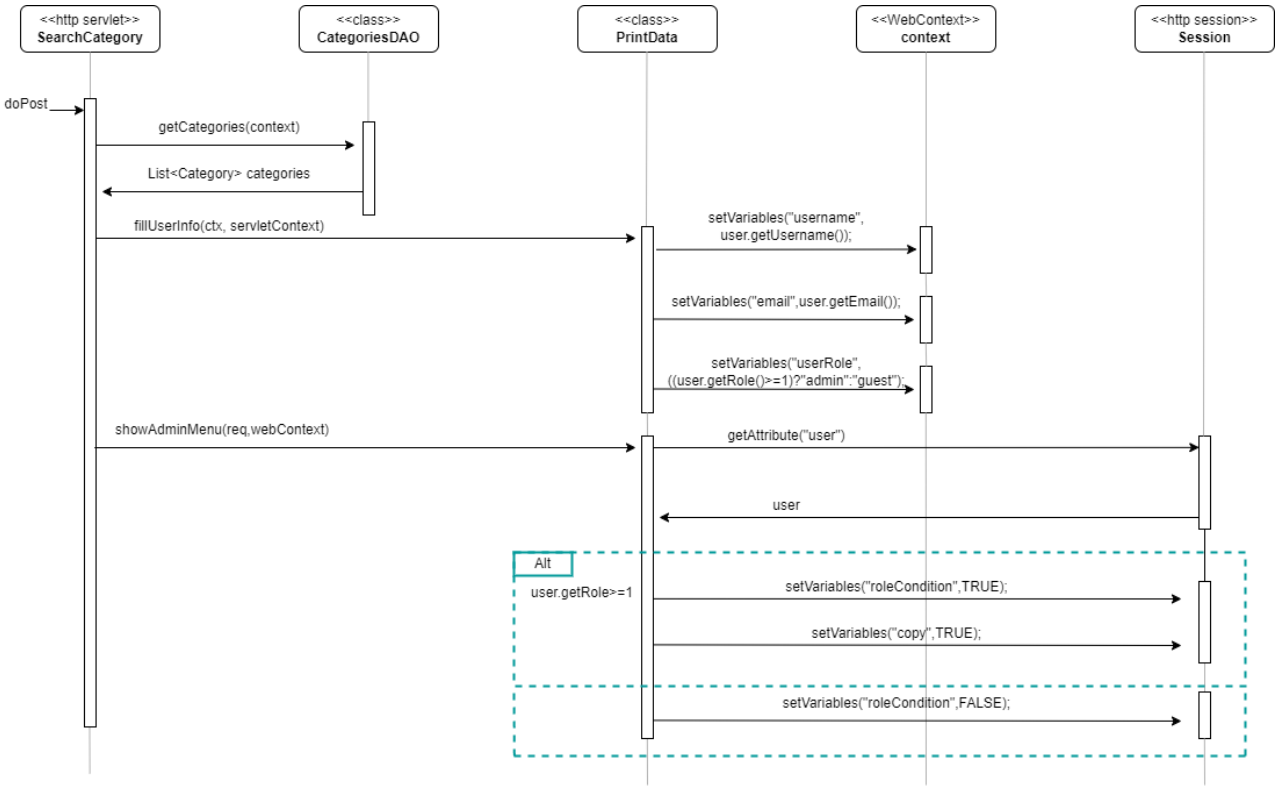


5.0.5 Un utente incolla una modifica alla tassonomia

Tramite quest'azione un utente salva le sue modifiche (copia-incolla o taglia-incolla) nel data base.

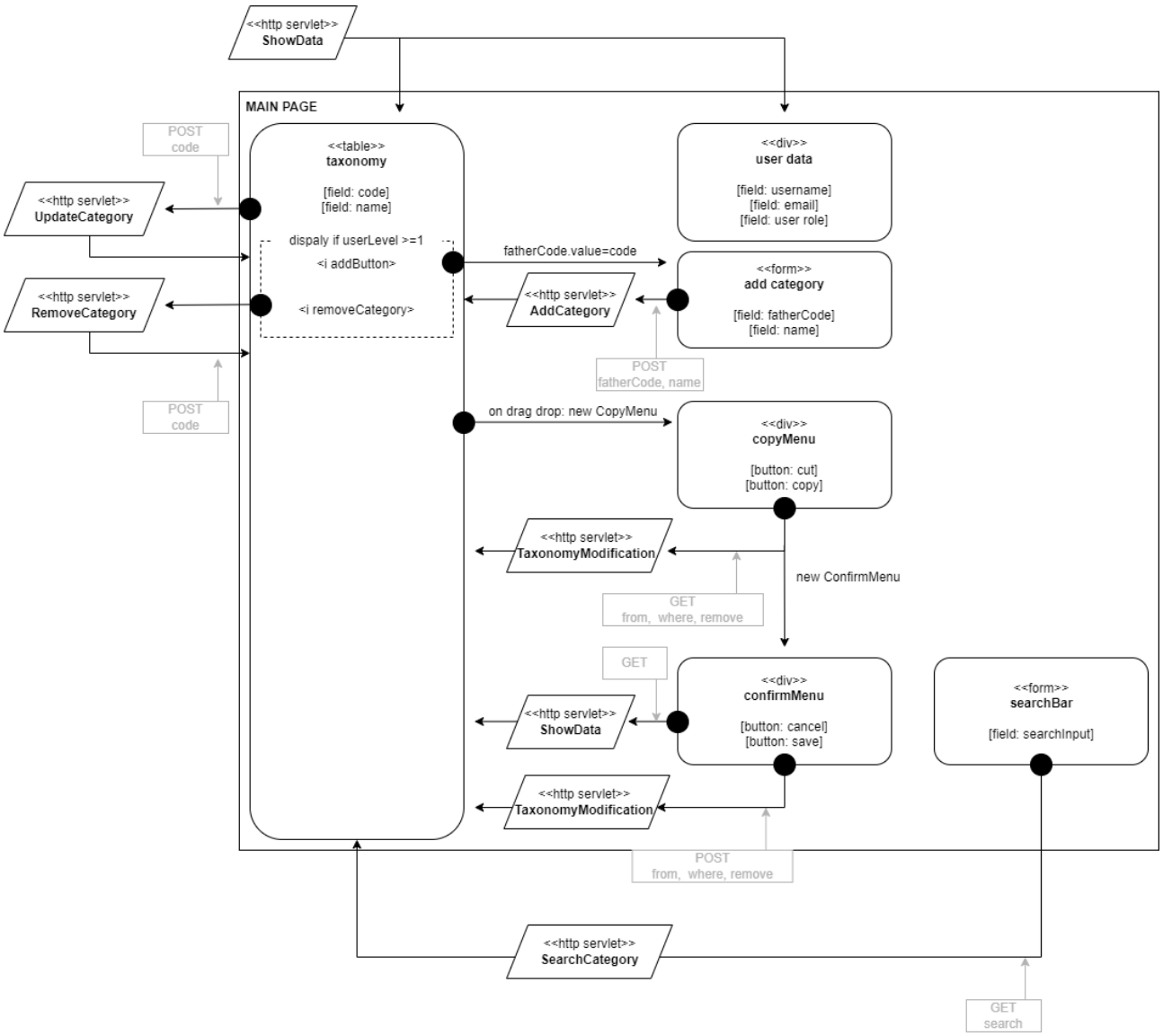


5.0.6 Un utente effettua una ricerca mediante la barra di ricerca



6 DESIGN DELL'APPLICAZIONE PER LA VERSIONE CON JAVASCRIPT

Le specifiche richiedono, per la versione javaScript, che il sito si strutturi in un'unica pagina dopo l'azione di login (o registrazione). Inoltre, tutti gli aggiornamenti all'unica pagina devono avvenire dinamicamente senza ricaricare completamente la pagina web.



NOTE: le specifiche prevedono inoltre che dopo l'azione "*copia*" (per drag and drop) venga fornito all'utente una tassonomia locale aggiornata da poter successivamente salvare sul server, rendendola quindi disponibile a tutti gli utenti.

Si è tuttavia deciso, considerando le possibili dimensioni della tassonomia, di far elaborare ed aggiornare la lista al server e non al client (tramite l'azione di una servlet dedicata).

7 COMPONENTI PER LA VERSIONE JAVASCRIP

7.1 Interazione con il DB

Per questa sezione si faccia riferimento a quanto detto per la versione pure-HTML poiché non sono state apportate modifiche a queste componenti.

7.2 Servlets

7.2.1 Controlli

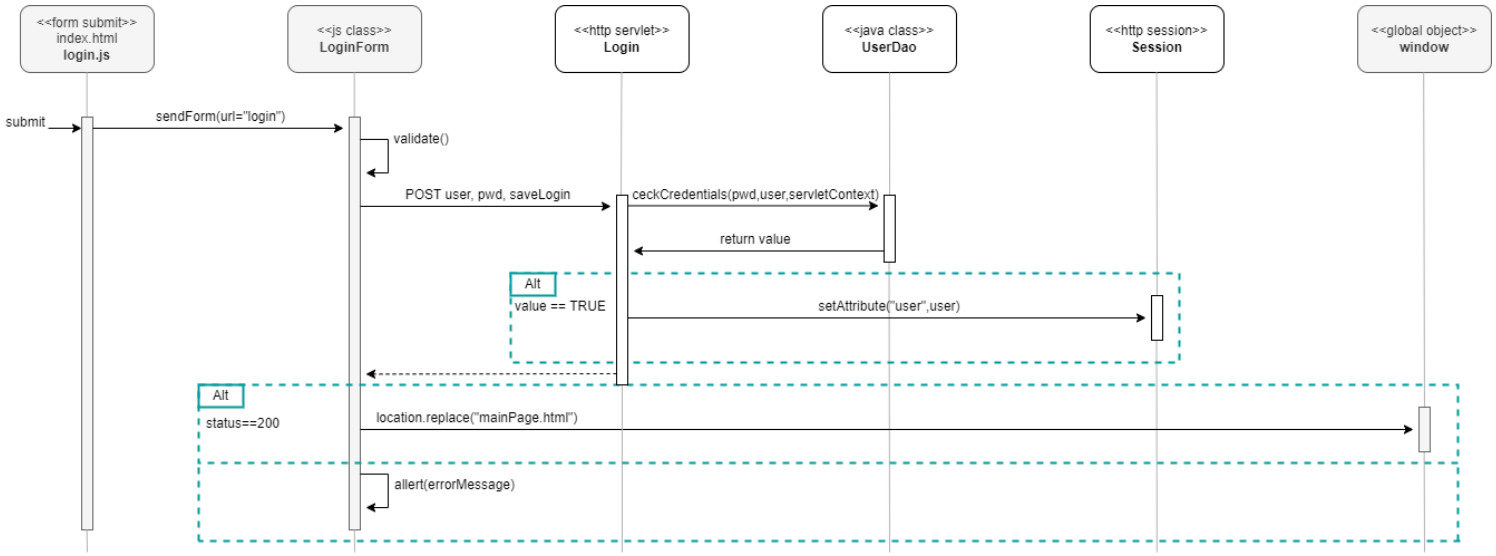
- AddCategory
- Login
- LoginCookies
- Logout
- RemoveCategory
- SearchCategory
- ShowData
- SignIn
- TaxonomyModification
- UpdateCategory

7.2.2 Filtri

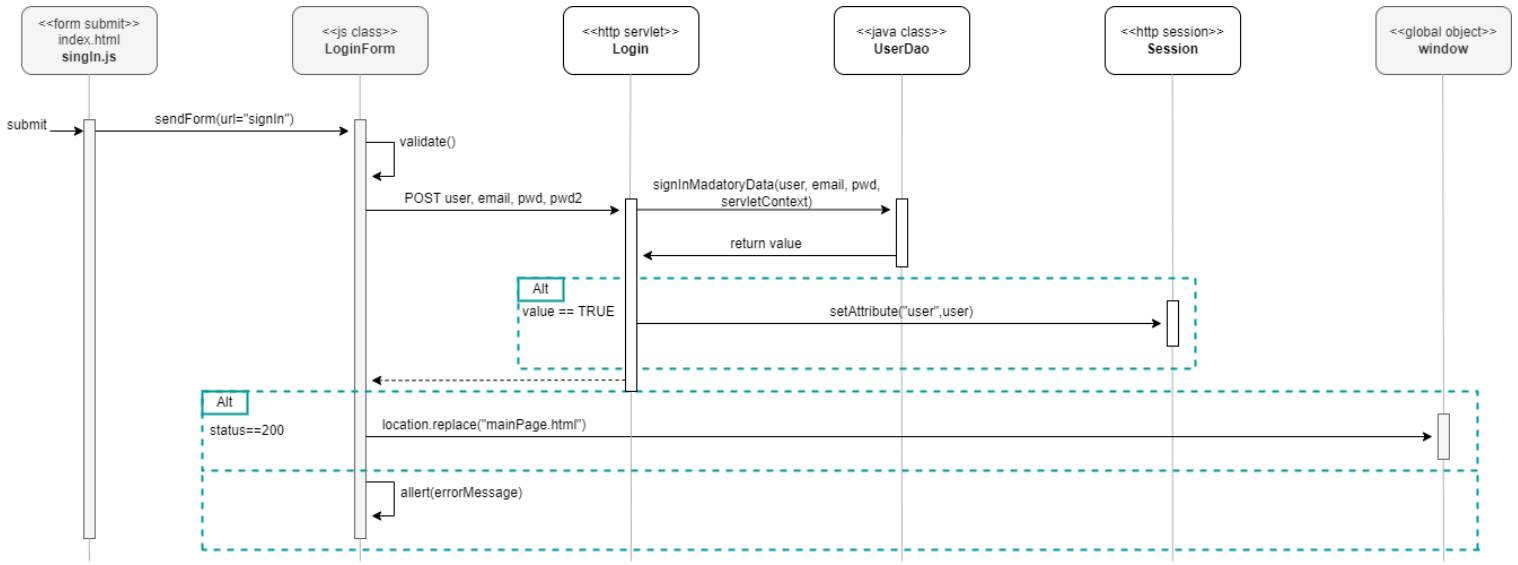
- AdminFistLevlFilter
- LoginFilter
- PasteFilter

8 EVENTI E SEQUENCE DIAGRAMS PER LA VERSIONE JAVASCRIPT

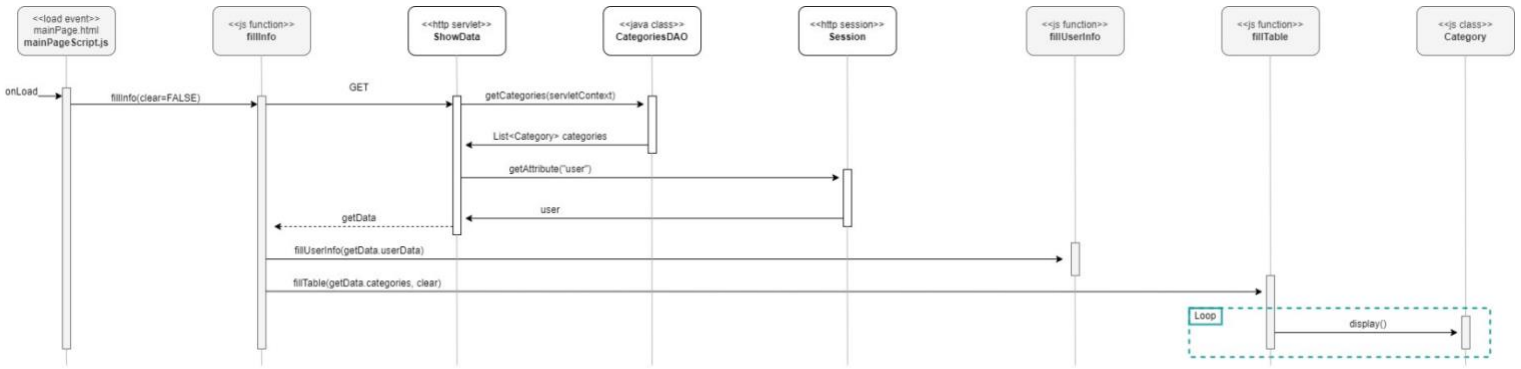
8.0.1 Un utente effettua il login



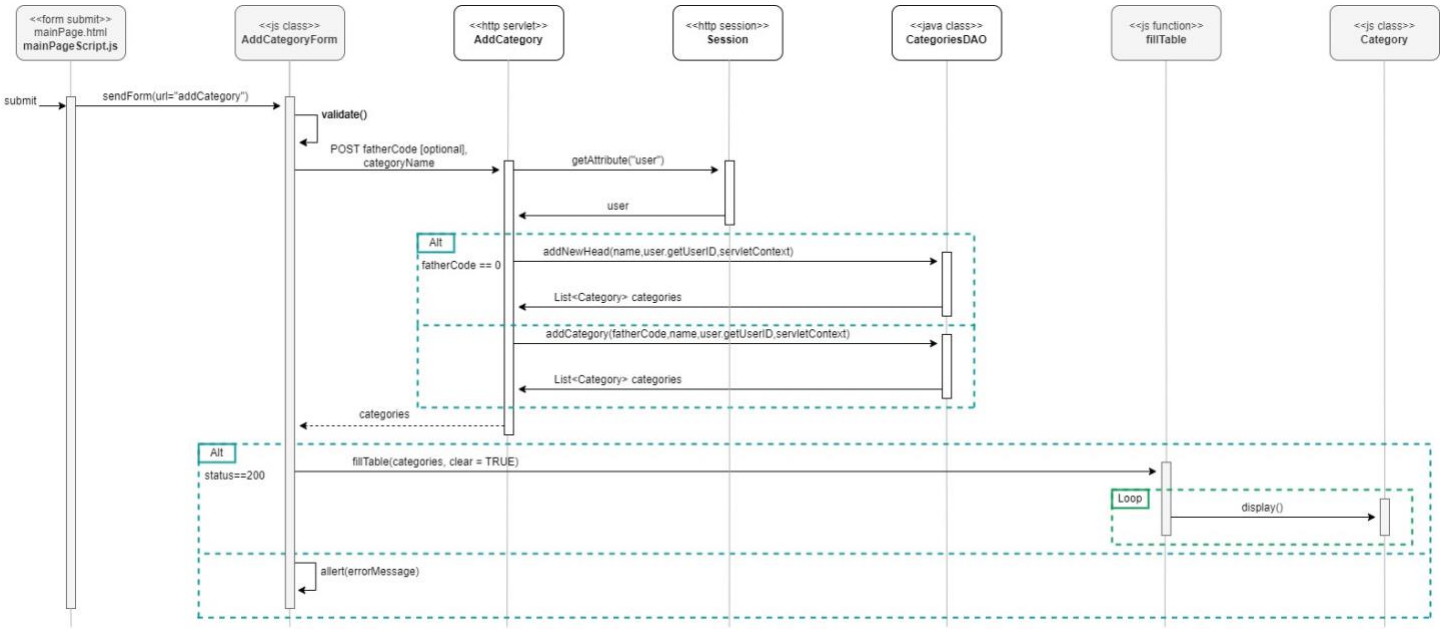
8.0.2 Un nuovo utente si registra



8.0.3 Un utente richiede la pagina principale



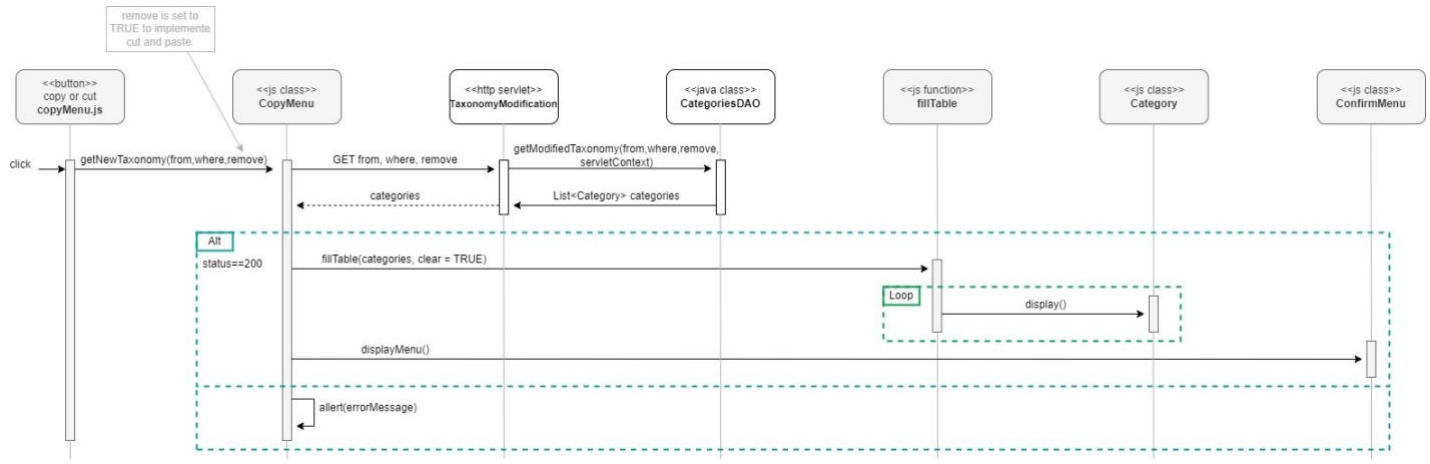
8.0.4 Un utente aggiunge una nuova categoria



8.0.5 Un utente richiede una versione locale della tassonomia modificata

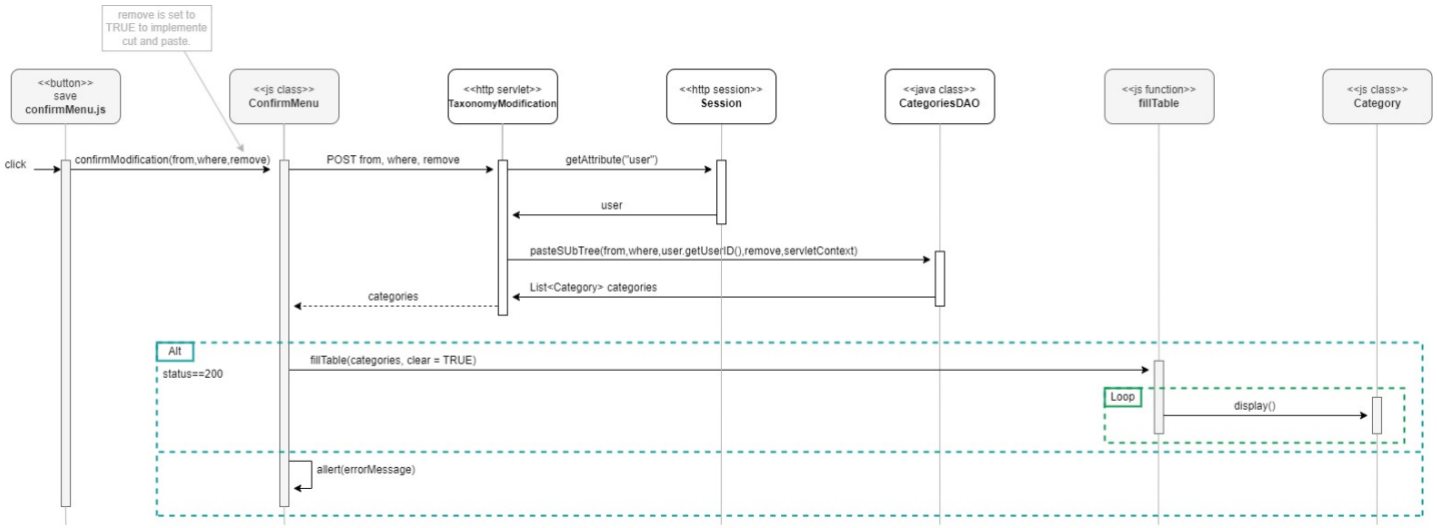
A seguito di un'operazione copia-incolla (o taglia-incolla), l'utente visualizza il risultato delle sue modifiche in una versione locale della tassonomia. Quest'ultima è comunque elaborata lato server (vedere nota CAP.6 per le ragioni di questa scelta).

> **NOTE:** con versione locale della tassonomia si fa riferimento ad una tassonomia visibile unicamente all'utente corrente.



8.0.6 Un utente salva una modifica alla tassonomia

Dopo aver ricevuto la versione locale della tassonomia modificata, l'utente può decidere di salvarla nel DB, rendendo così visibile la modifica al resto degli utenti.



8.0.7 Un utente aggiorna il nome di una categoria