Tecnologie Informatiche per il Web

AA. 2020/21

Docente: Piero Fraternali

Studente: Zheng Maria Yu

(Gruppo 88)

Catalogazione di immagini – Versione RIA

Un’applicazione permette all’utente di gestire una tassonomia di classificazione utile per etichettare immagini allo scopo di consentire la ricerca in base alla 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. Le categorie hanno nomi distinti.

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. Per ogni categoria il numero massimo di sottocategorie è 9.

Dopo la creazione di una categoria, la pagina HOME mostra l’albero aggiornato. L’utente può spostare di posizione una categoria: per fare ciò clicca sul link “sposta” associato alla categoria da spostare. A seguito di tale azione l’applicazione mostra, sempre nella HOME page, l’albero con evidenziato il sotto albero attestato sulla categoria da spostare: tutte le altre categorie hanno un link “sposta qui”. La selezione di un link “sposta qui” comporta l’inserimento della categoria da spostare come ultimo figlio della categoria destinazione.

Le modifiche effettuate da un utente e salvate nella base di dati diventano visibili agli altri utenti. Lo spostamento potrebbe creare un vuoto nella numerazione delle categorie figlie dello stesso padre.

Analisi dei dati

Un'applicazione permette all'utente di gestire una tassonomia di classificazione utile per etichettare immagini allo scopo di consentire la ricerca in base alla 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. Le categorie hanno nomi distinti.

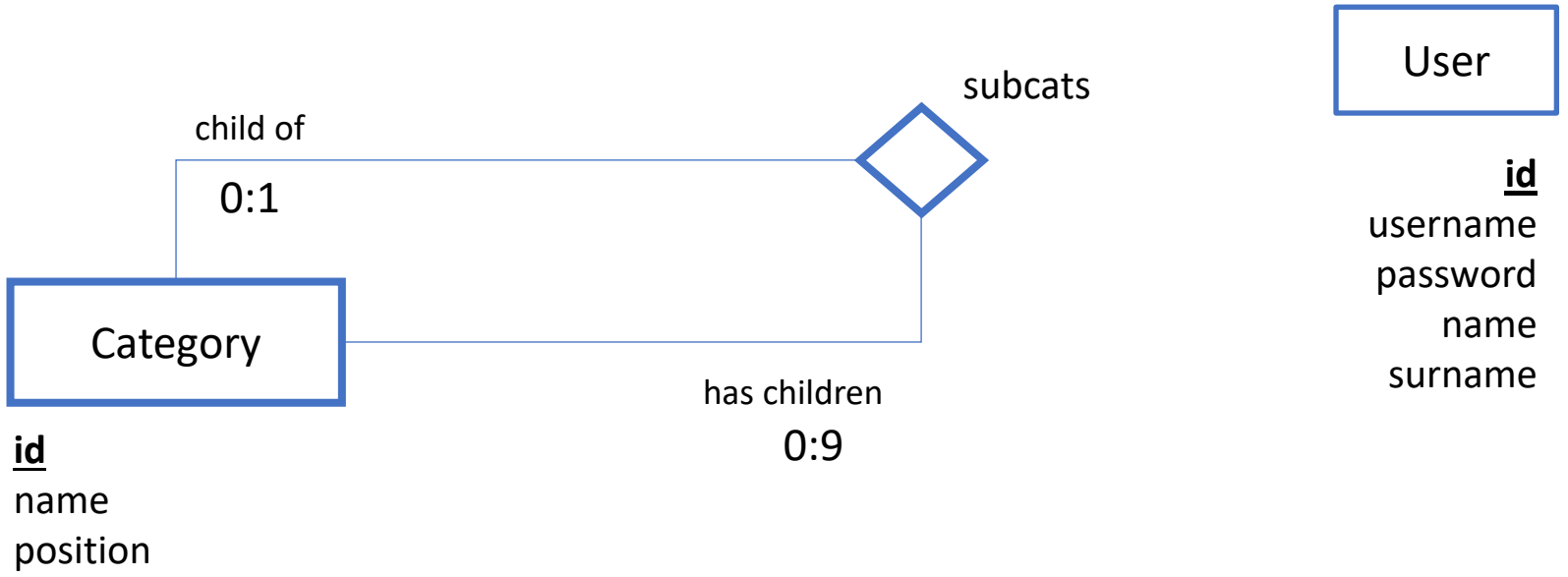
L'utente può inserire una nuova categoria nell'albero. Per fare ciò usa una form nella pagina HOME in cui specifica il nome della nuova categoria e sceglie la categoria padre. L'invio della nuova categoria comporta l'aggiornamento dell'albero: la nuova categoria è appesa alla categoria padre come ultimo sottoelemento. Alla nuova categoria viene assegnato un codice numerico che ne riflette la posizione. Per ogni categoria il numero massimo di sottocategorie è 9.

Dopo la creazione di una categoria, la pagina HOME mostra l'albero aggiornato. L'utente può spostare di posizione una categoria: per fare ciò clicca sul link "sposta" associato alla categoria da spostare. A seguito di tale azione l'applicazione mostra, sempre nella HOME page, l'albero con evidenziato il sotto albero attestato sulla categoria da spostare: tutte le altre categorie hanno un link "sposta qui". La selezione di un link "sposta qui" comporta l'inserimento della categoria da spostare come ultimo figlio della categoria destinazione.

Le modifiche effettuate da un utente e salvate nella base di dati diventano visibili agli altri utenti. Lo spostamento potrebbe creare un vuoto nella numerazione delle categorie figlie dello stesso padre.

(**Entità**, **attributi**, **relazioni**)

Progettazione del database



Ogni categoria può avere al massimo 9 categorie figlie,
ed è figlio di una categoria (tranne root che non ha categoria padre).

Schema database locale

```
CREATE TABLE `user` (
        `id` int NOT NULL AUTO_INCREMENT,
        `username` varchar(45) NOT NULL,
        `password` varchar(45) NOT NULL,
        `name` varchar(45) NOT NULL,
        `surname` varchar(45) NOT NULL,
        PRIMARY KEY (`id`),
        UNIQUE KEY `username_UNIQUE` (`username`)
)
```

Schema database locale

```
CREATE TABLE `category` (
        `id` int NOT NULL AUTO_INCREMENT,
        `name` varchar(45) NOT NULL,
        `position` varchar(45) NOT NULL,
        PRIMARY KEY (`id`),
        UNIQUE KEY `name_UNIQUE` (`name`)
)
```

Schema database locale

```
CREATE TABLE `subcats` (
        `father` int NOT NULL,
        `child` int NOT NULL,
        PRIMARY KEY (`father`,`child`),
        KEY `childtocategory_idx` (`child`),
        CONSTRAINT `childtocategory` FOREIGN KEY (`child`) REFERENCES
        `category` (`id`),
        CONSTRAINT `fathertocategory` FOREIGN KEY (`father`) REFERENCES
        `category` (`id`)
)
```

Analisi dei requisiti dell'applicazione

Un'applicazione permette all'utente di gestire una tassonomia di classificazione utile per etichettare immagini allo scopo di consentire la ricerca in base alla 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. Le categorie hanno nomi distinti.

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. Per ogni categoria il numero massimo di sottocategorie è 9.

Dopo la creazione di una categoria, la pagina HOME mostra l'albero aggiornato. L'utente può spostare di posizione una categoria: per fare ciò clicca sul link "sposta" associato alla categoria da spostare. A seguito di tale azione l'applicazione mostra, sempre nella HOME page, l'albero con evidenziato il sotto albero attestato sulla categoria da spostare: tutte le altre categorie hanno un link "sposta qui". La selezione di un link "sposta qui" comporta l'inserimento della categoria da spostare come ultimo figlio della categoria destinazione.

Le modifiche effettuate da un utente e salvate nella base di dati diventano visibili agli altri utenti. Lo spostamento potrebbe creare un vuoto nella numerazione delle categorie figlie dello stesso padre.

(**Pagine**, **componenti**, **eventi**, **azioni**)

Versione con Javascript

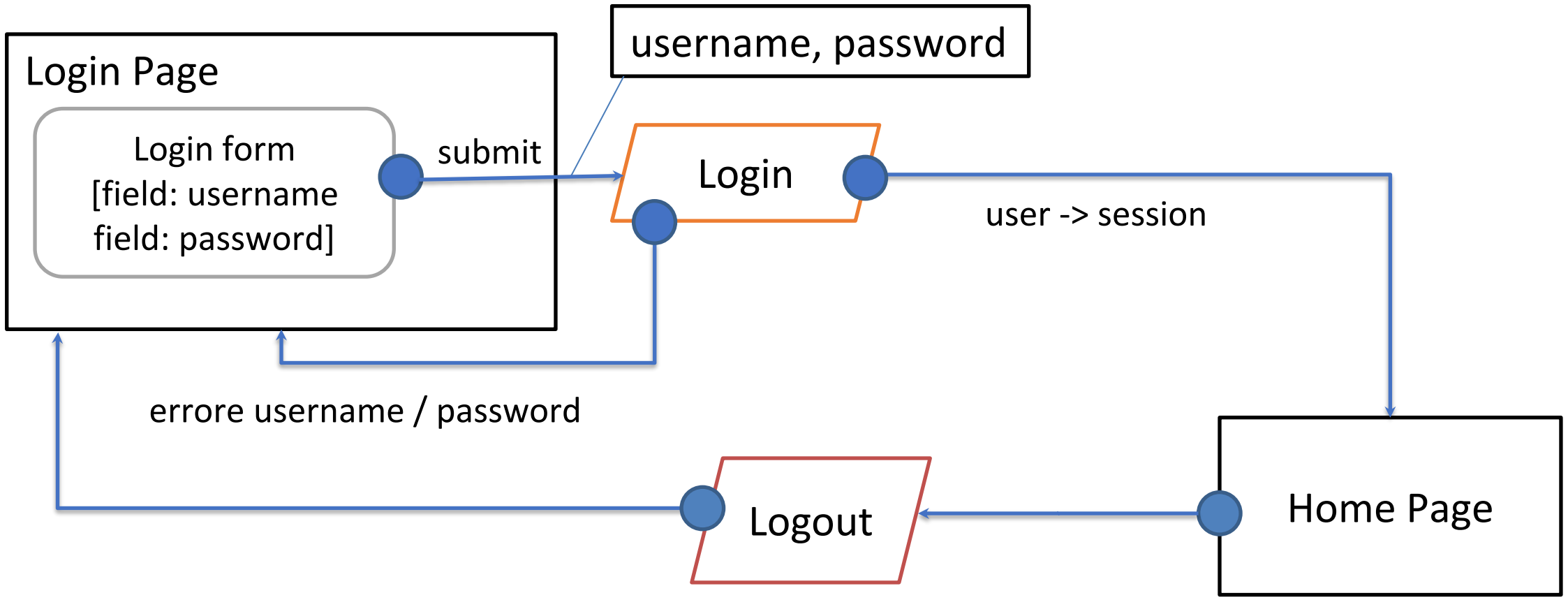
Si realizzi un'applicazione client server web che estende e/o modifica le specifiche precedenti come segue:

- 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 spostamento di una categoria è realizzata mediante drag & drop.
- A seguito del drop della categoria da spostare compare una finestra di dialogo con cui l'utente può confermare o cancellare lo spostamento. La conferma produce l'aggiornamento a lato client dell'albero.
- L'utente realizza spostamenti anche multipli a lato client. A seguito del primo spostamento compare un bottone SALVA la cui pressione provoca l'invio al server dell'elenco degli spostamenti realizzati (NON dell'intero albero). L'invio degli spostamenti produce l'aggiornamento dell'albero nella base dei dati e la comparsa di un messaggio di conferma dell'avvenuto salvataggio.

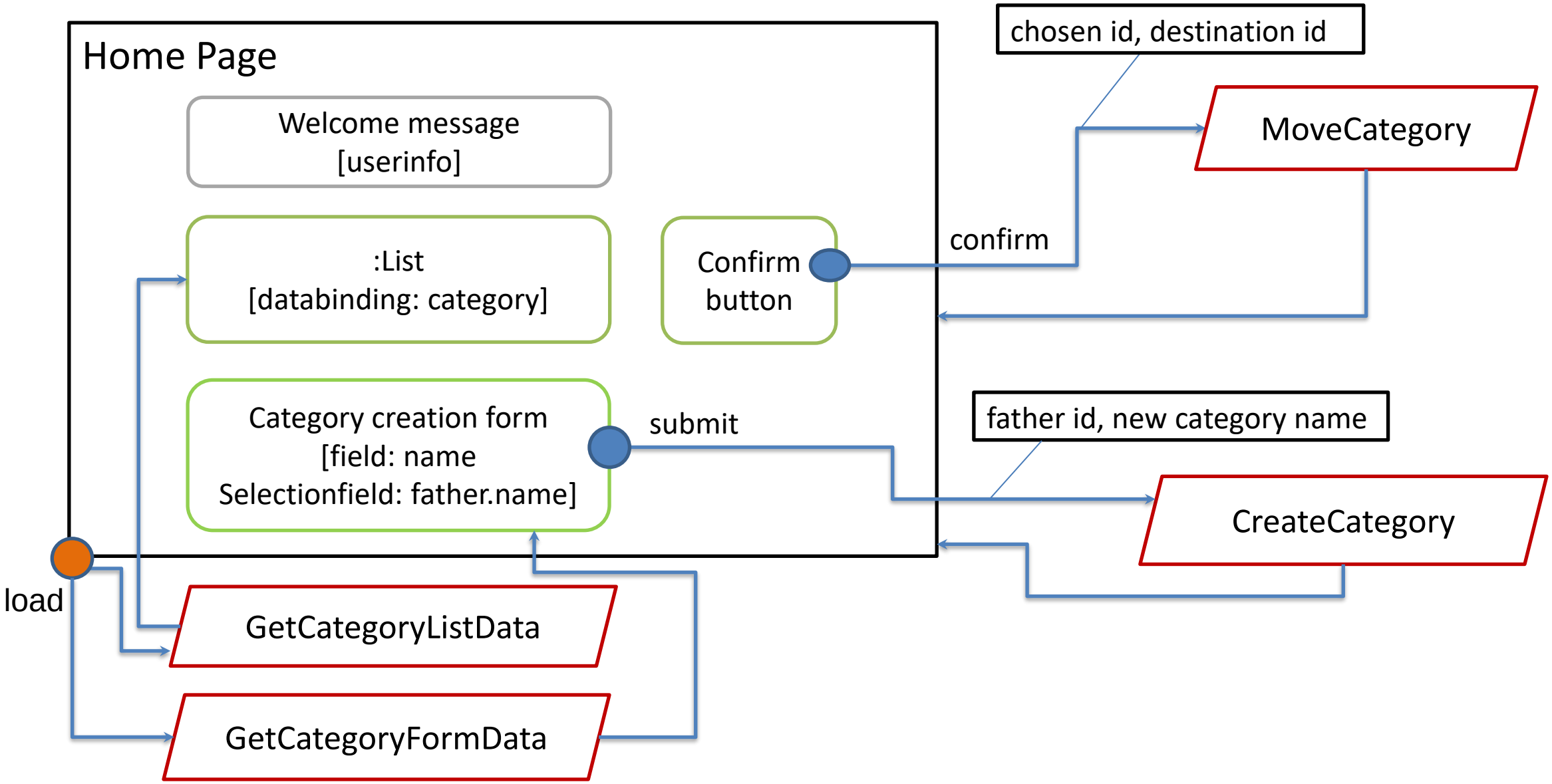
Completamento delle specifiche

- Si effettua il login nella pagina di default tramite una form.
- Le categorie non possono avere nomi nulli.
- I dati dell’utente non possono essere nulli.
- Non è possibile spostare una categoria in una delle sue sottocategorie.
- Dalla Home è possibile effettuare logout cliccando su "Logout" , tornando quindi alla pagina di default.

Progettazione dell'applicazione



Progettazione dell'applicazione



Eventi & Azioni

Client side		Server side	
Evento	**Azione**	**Evento**	**Azione**
index.html → login form → submit	Controllo dati	POST (username, password)	Controllo credenziali
HomePage→ load	Aggiornamento view con dati dell'albero di categorie e del form	GET	Estrazione dati albero categorie Estrazione dati form
HomePage → albero categorie → drag and drop	Aggiornamento view dovuto all'operazione Richiesta conferma Salvataggio modifiche in locale	-	-
HomePage → confirm	Estrazione elenco modifiche	POST (elenco modifiche)	Controllo validità operazioni Inserimento modifiche
HomePage → form creazione → submit	Controllo dati	POST (dati categoria nuova)	Controllo validità dati Inserimento categoria
Logout	-	GET	Terminazione della sessione

Controller & Event Handler

Client side		Server side	
Evento	**Controllore**	**Evento**	**Controllore**
index.html → login form → submit	Function makeCall	POST (username, password)	Login (servlet)
HomePage→ load	Function PageOrchestrator	GET	GetCategoryListData (servlet) GetCategoryFormData (servlet)
HomePage → albero categorie → drag and drop	Function dragStart, dragOver, dragLeave, drop	-	-
HomePage → confirm	Function makeCallJson	POST (elenco modifiche)	MoveCategory (servlet)
HomePage → form creazione → submit	Function makeCall	POST (dati categoria nuova)	CreateCategory (servlet)
Logout	-	GET	Logout (servlet)

Componenti lato server

- Model Objects (Beans)
  - User
  - Category
- Controllers (Servlets)
  - Login
  - GetCategoryListData
  - GetCategoryFormData
  - CreateCategory
  - MoveCategory
  - Logout
- Data Access Objects (Classes)
  - UserDAO
    - checkCredentials(username,password)
  - CategoryDAO
    - findAllCategories()
    - findTopsAndSubtrees()
    - findSubclasses(category)
    - createCategory(name,fid)
    - handleMove(array)
    - moveCategory(cid,destid)
    - findNumChild(father)
    - updatePosition(position,id)
    - insertNewCategory(name,position)
    - addLink(fid,cid)

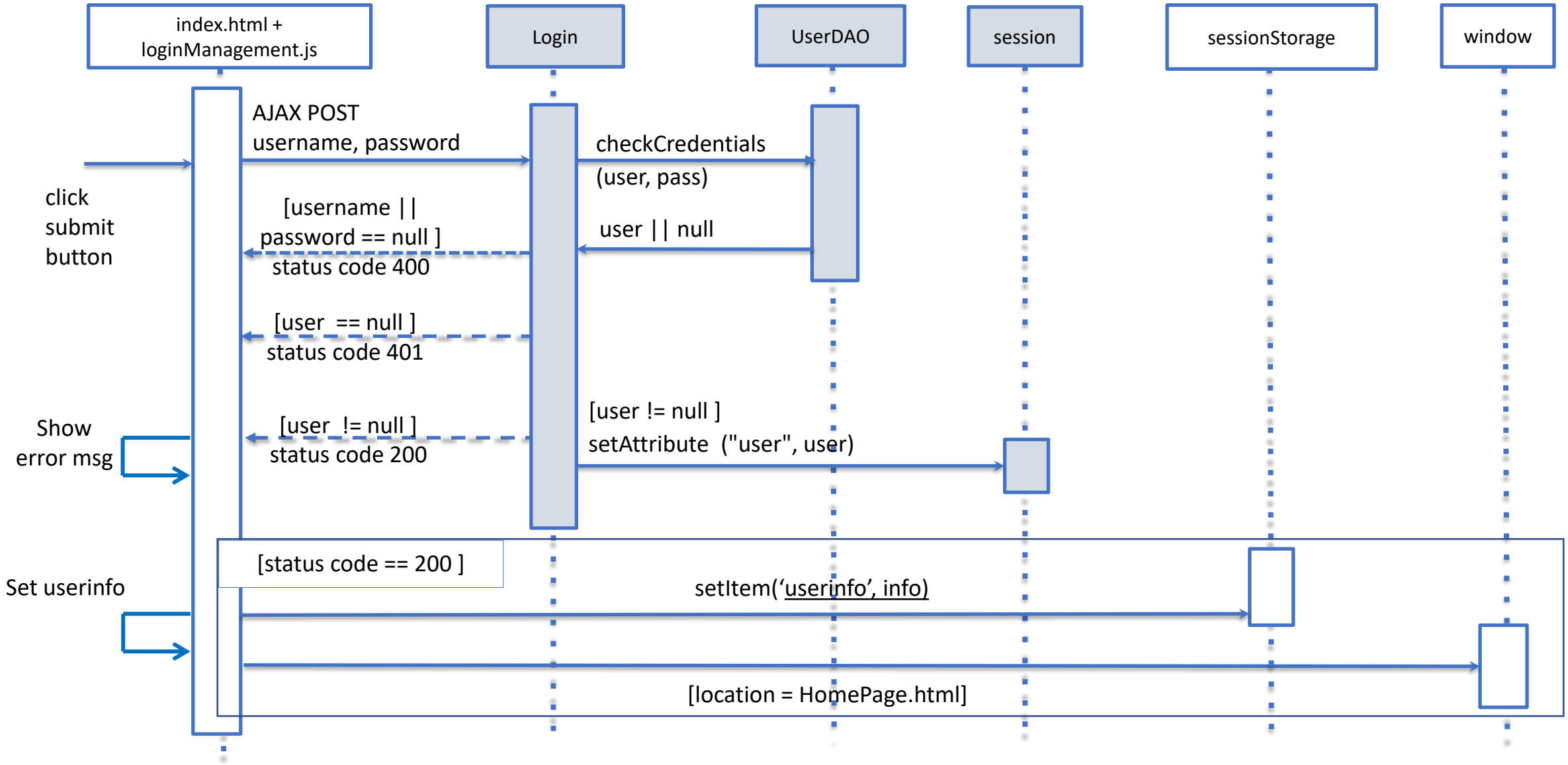
Componenti lato client

- index.html
  - Login form: gestione submit ed errori
- HomePage
  - CategoriesList
    - show(): richiede al server i dati dell'albero delle categorie
    - update(): riceve i dati e aggiorna la vista
  - CreationForm
    - show(): richiede al server i dati delle opzioni del form
    - update(): aggiorna le opzioni del form
  - ConfirmButton
    - reset(): nasconde il pulsante di conferma
    - show(): mostra il pulsante di conferma
    - registerEvents(): associa al componente le funzioni per la gestione dei spostamenti

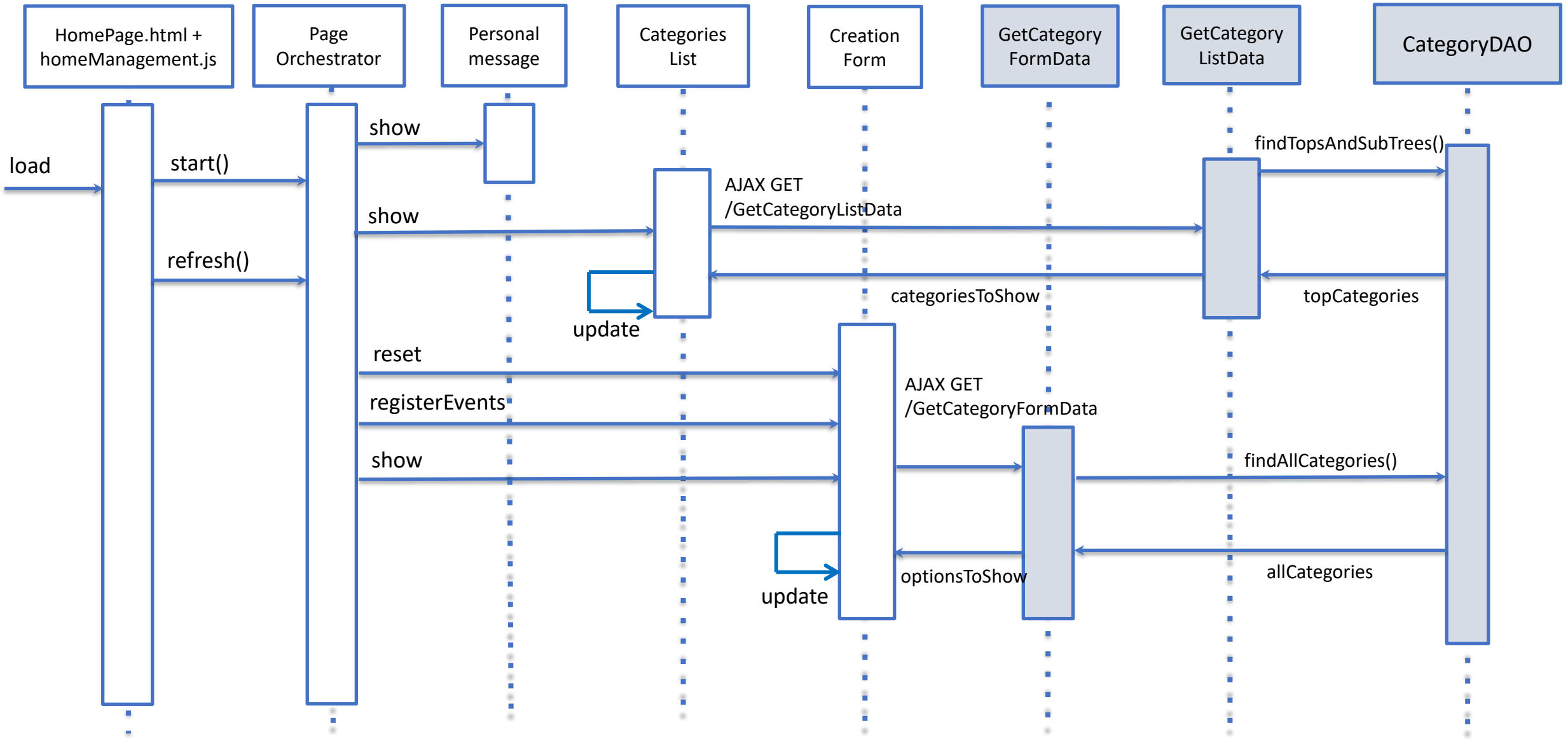
Client side

Server side

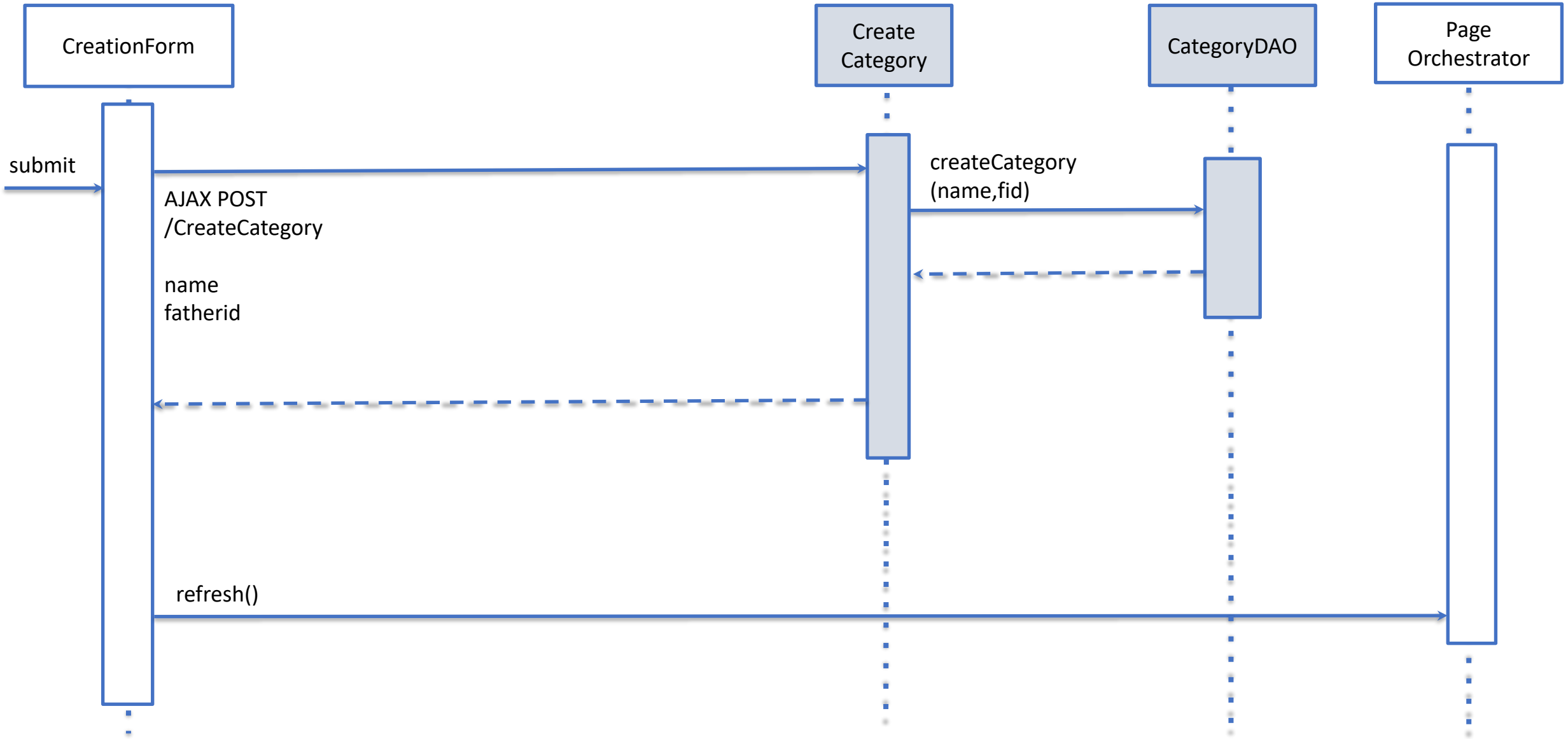
Evento: Login



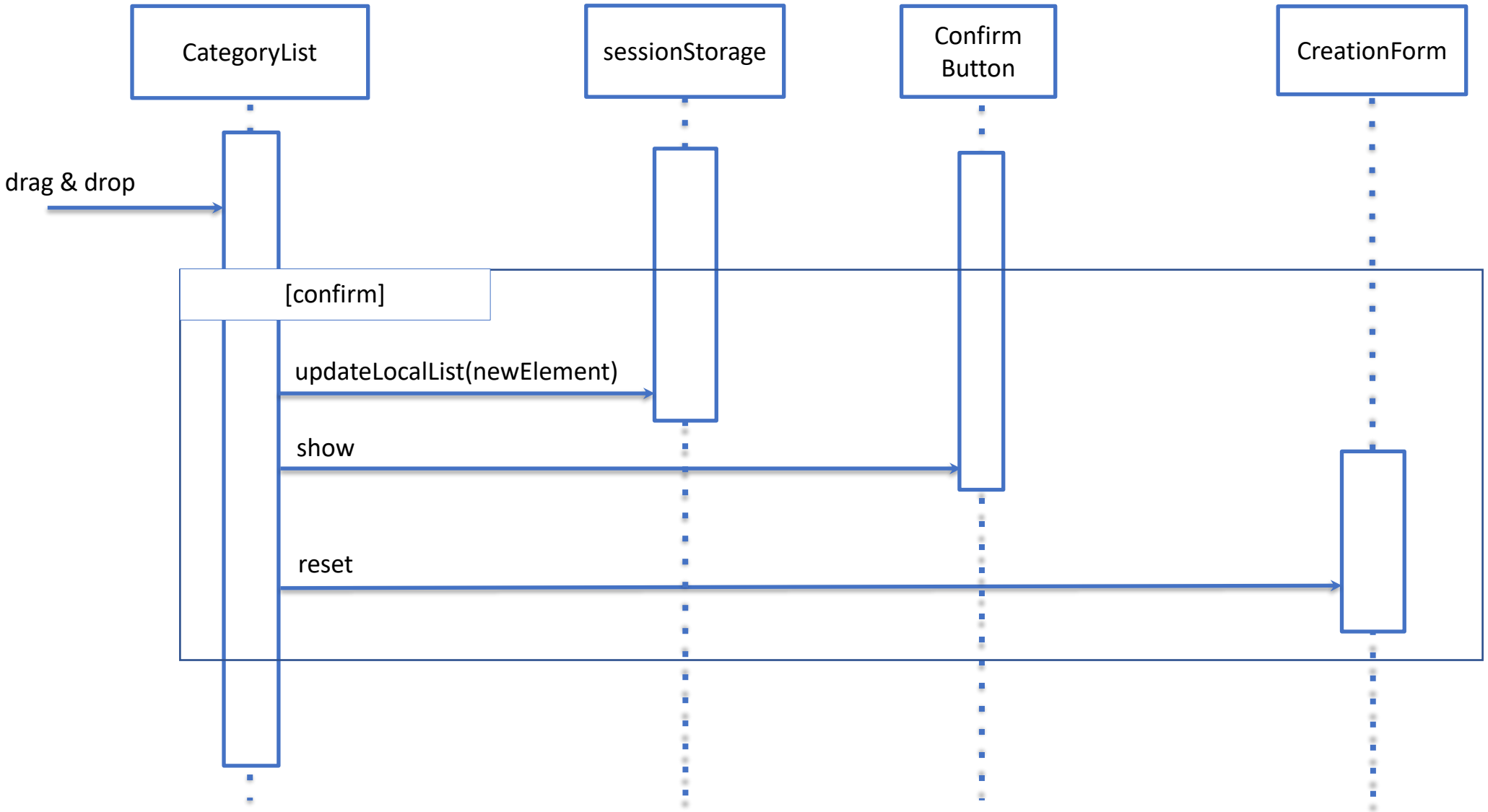
Evento: Caricamento Home page



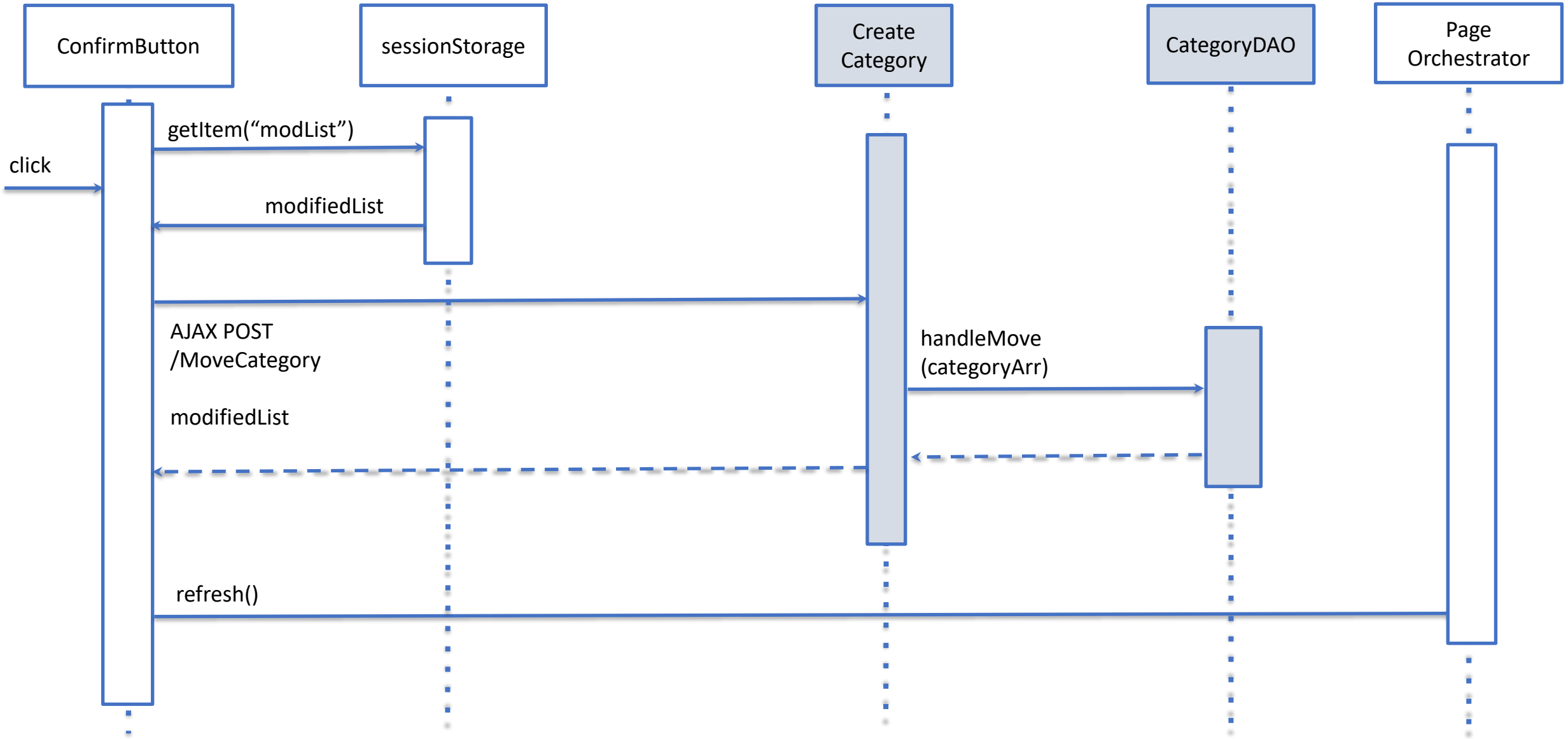
Evento: Creazione categoria



Evento: Drag & Drop



Evento: Spostamento categoria



Evento: Logout