Documentazione
Gruppo 134

Belfiore Mattia, Benedetti Gabriele

Analisi dei Dati

L'applicazione supporta registrazione e login mediante una pagina pubblica con opportune form. La registrazione richiede l'inserimento di **username**, **indirizzo di email** e **password** e controlla la validità sintattica dell'indirizzo di email e l'uguaglianza tra i campi "password" e "ripeti password". La registrazione controlla l'unicità dello username. Una **cartella** ha un **proprietario**, un **nome** e una **data di creazione** e **può contenere altre cartelle e/o documenti**. Un **documento** ha un **proprietario**, **nome**, una **data di creazione**, un **sommario** e un **tipo**. Quando l'**utente** accede all'applicazione appare una HOME PAGE che **contiene** un albero delle **proprie cartelle** Nell'HOME PAGE l'utente può selezionare una cartella e accedere a una pagina CONTENUTI che mostra l'elenco delle cartelle e dei documenti di una cartella. Ogni documento in elenco ha due link: accedi e sposta. Quando l'utente seleziona il link accedi, appare una pagina DOCUMENTO(nella stessa finestra e tab del browser) che mostra tutti i dati del documento selezionato. Quando l'utente seleziona il link sposta, appare la HOME PAGE con l'albero delle cartelle;

in questo caso la pagina mostra il messaggio "Stai spostando il documento X dalla cartella Y. Scegli la cartella di destinazione", la cartella a cui appartiene il documento da spostare NON è selezionabile e il suo nome è evidenziato (per esempio con un colore diverso). Quando l'utente seleziona la cartella di destinazione, il documento è spostato dalla cartella di origine a quella di destinazione e appare la pagina CONTENUTI che mostra il contenuto aggiornato della cartella di destinazione. Ogni pagina, tranne la HOME PAGE, contiene un collegamento per tornare alla pagina precedente. L'applicazione consente il logout dell'utente da qualsiasi pagina. Una pagina GESTIONE CONTENUTI raggiungibile dalla HOME PAGE permette all'utente di creare una cartella di primo livello, una cartella all'interno di una cartella esistente e un documento all'interno di una cartella. L'applicazione non richiede la gestione dell'upload dei documenti e delle sottocartelle.

Entities, **Attributes**, **Relationships**

Analisi dei Dati

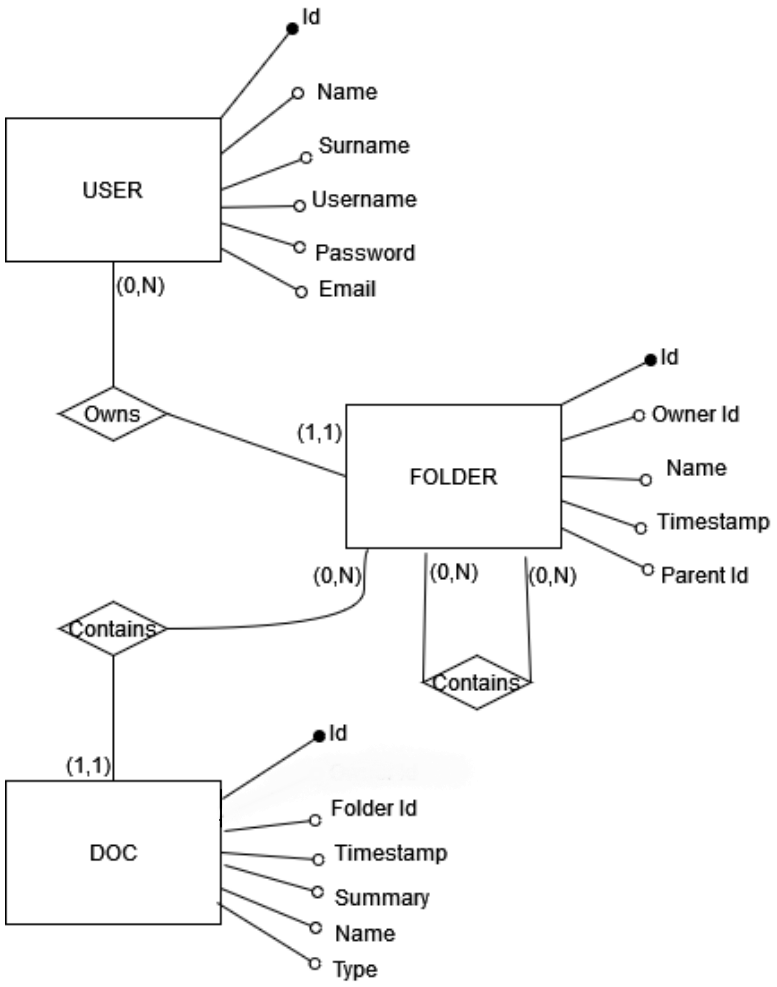
L'applicazione supporta **registrazione** e **login** mediante una pagina pubblica con opportune form. La **registrazione** richiede l'inserimento di username, indirizzo di email e password e **controlla la validità sintattica dell'indirizzo di email e l'uguaglianza tra i campi "password" e "ripeti password"**. La registrazione **controlla l'unicità dello username**. Una cartella ha un proprietario, un nome e una data di creazione e può contenere altre cartelle e/o documenti. Un documento ha un proprietario, nome, una data di creazione, un sommario e un tipo. **Quando l'utente accede all'applicazione appare una HOME PAGE** che contiene un **albero delle proprie cartelle** Nell'HOME PAGE l'utente può **selezionare una cartella** e accedere a una pagina **CONTENUTI** che **mostra l'elenco delle cartelle e dei documenti di una cartella**. Ogni documento in elenco ha due link: **accedi e sposta**. Quando l'utente seleziona il link accedi, **appare una pagina DOCUMENTO**(nella stessa finestra e tab del browser) **che mostra tutti i dati del documento selezionato**. Quando l'utente seleziona il link sposta, **appare la HOME PAGE con l'albero delle cartelle**;

in questo caso la pagina **mostra il messaggio** "Stai spostando il documento X dalla cartella Y. Scegli la cartella di destinazione", la cartella a cui appartiene il documento da spostare NON è selezionabile e **il suo nome è evidenziato** (per esempio con un colore diverso). Quando l'utente **seleziona la cartella di destinazione**, il documento è spostato dalla cartella di origine a quella di destinazione e appare la pagina CONTENUTI che **mostra il contenuto aggiornato della cartella di destinazione**. Ogni pagina, tranne la HOME PAGE, contiene un collegamento **per tornare alla pagina precedente**. L'applicazione consente il **logout dell'utente da qualsiasi pagina**. Una pagina **GESTIONE CONTENUTI** raggiungibile dalla HOME PAGE **permette all'utente di creare una cartella di primo livello, una cartella all'interno di una cartella esistente e un documento all'interno di una cartella**. L'applicazione non richiede la gestione dell'upload dei documenti e delle sottocartelle.

Pages (Views), **View components**, **Events**, **Actions**

Database Design

Schema Concettuale



Schema Logico

User(**Id**, Name, Surname, Username, Email, Password)

Folder(**Id**, Owner Id, Name, Timestamp, Parent Id)

Doc(**Id**, Folder Id, Name, Timestamp, Summary, Type)

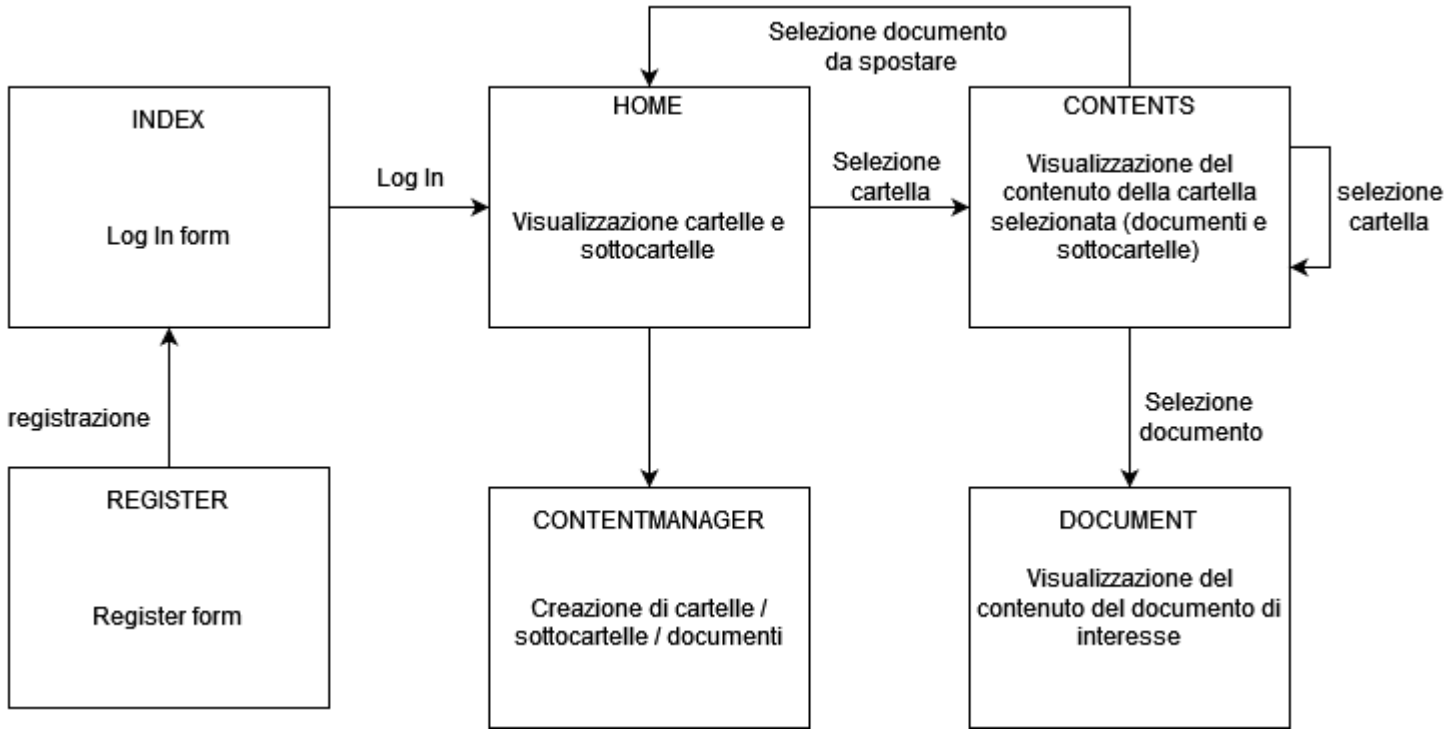
Database schema

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

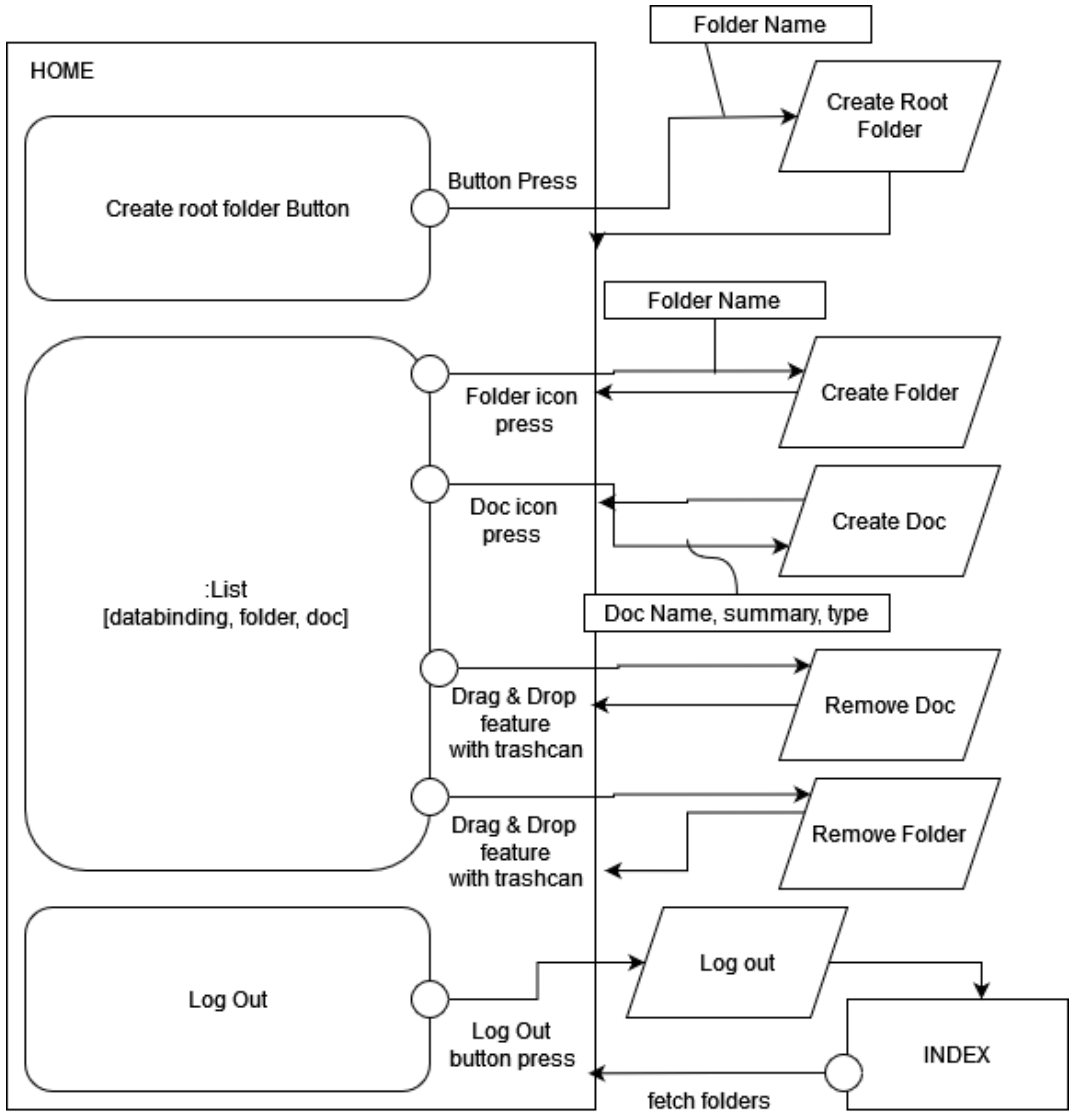
```
CREATE TABLE `folder` (
  `folder_id` bigint unsigned NOT NULL AUTO_INCREMENT,
  `owner_id` int DEFAULT NULL,
  `folder_name` varchar(100) NOT NULL,
  `created_at` timestamp NULL DEFAULT CURRENT_TIMESTAMP,
  `parent_folder_id` bigint unsigned DEFAULT NULL,
  PRIMARY KEY (`folder_id`),
  UNIQUE KEY `folder_id` (`folder_id`),
  FOREIGN KEY (`parent_folder_id`) REFERENCES `folder` (`folder_id`) ON UPDATE CASCADE ON DELETE CASCADE
)
```

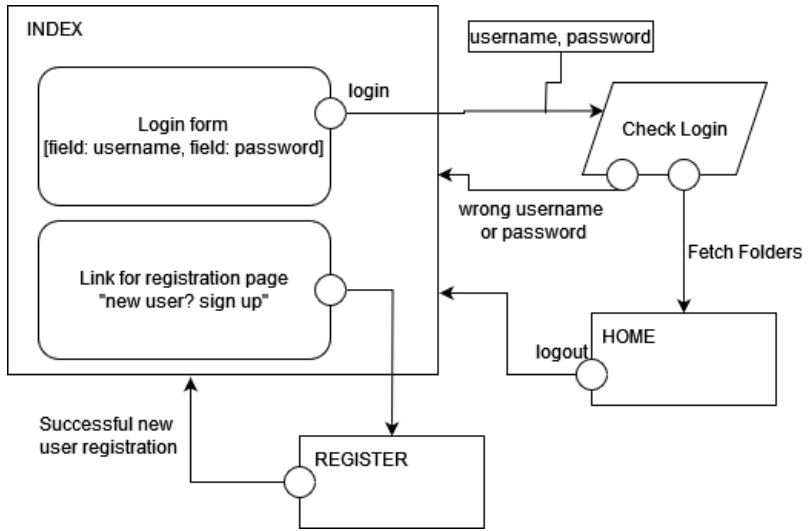
```
CREATE TABLE `doc` (
  `document_id` bigint unsigned NOT NULL AUTO_INCREMENT,
  `folder_id` bigint unsigned DEFAULT NULL,
  `doc_name` varchar(100) NOT NULL,
  `summary` text,
  `created_at` timestamp NULL DEFAULT CURRENT_TIMESTAMP,
  `type` varchar(50) DEFAULT NULL,
  PRIMARY KEY (`document_id`),
  UNIQUE KEY `document_id` (`document_id`),
  FOREIGN KEY (`folder_id`) REFERENCES `folder` (`folder_id`) ON UPDATE CASCADE ON DELETE CASCADE
)
```

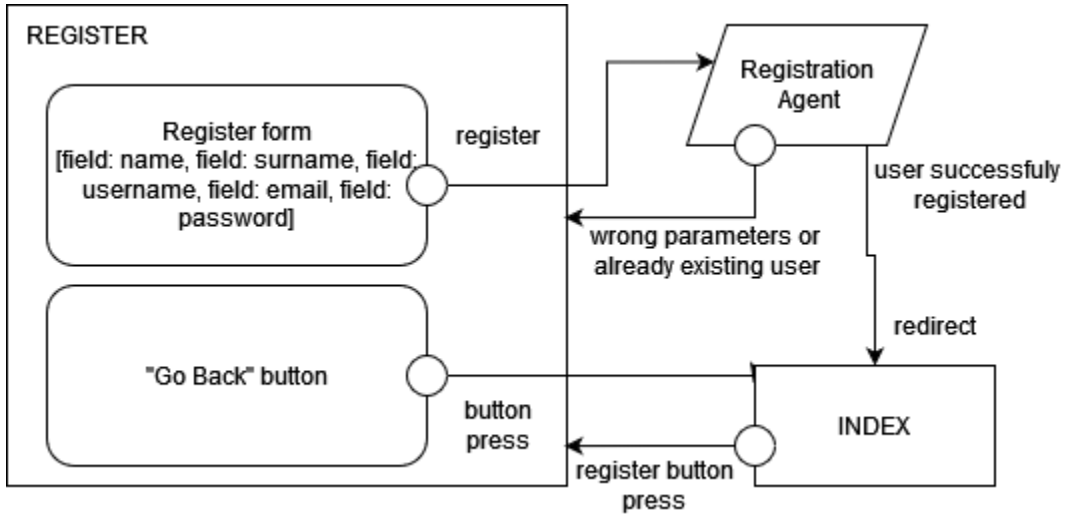
Application design – HTML overview



Application design – RIA overview







Componenti ⟶ Server Side

- Controllers
  - CheckLogin
  - CreateDocument
  - CreateFolder
  - CreateRootFolder
  - GoToContentManagement
  - GoToContents
  - GoToDocumentDetails
  - GoToHome
  - GoToMoveDocument
  - GoToRegistrationPage
  - Logout
  - MoveDocument
  - RegistrationAgent
  - GetDocs
  - GetFolders
  - RemoveDoc
  - RemoveFolder

- Model Objects(Beans)
  - User
  - Doc
  - Folder

- Filters
  - LoginChecker

- Data Access Objects (DAO)
  - UserDAO
    - checkCredentials(String username, String password)
    - uniqueUsername(String username)
    - createUser(String username. String name, String surname, String email, String password)
  - FolderDAO
    - getRootFoldersByOwner(int ownerId)
    - getFoldersByOwner(int ownerId)
    - getFolder(int ownerId, int folderId)
    - createRootFolder(int ownerId, String foldername)
    - createFolder(int ownerId, String foldername, int parentId)
    - uniqueRootFolderName(int ownerId, String foldername)
    - uniqueFolderName(int ownerId, int parentId, String foldername)
    - removeFolder(int folderId, int userId)
  - DocDAO
    - getDocById(int docId)
    - getDocsByFolder(int folderId)
    - getDocsByOwner(int ownerId)
    - createDoc(int ownerId, int folderId, String docName, String summary, String type)
    - uniqueDocName(int ownerId, int folderId, String name)
    - moveDoc(int ownerId, int docId, int to)
    - removeDoc(int docId)

Legend
- Solo per la versione HTML
- Solo per la versione RIA

Funzioni js

- Login
  - makePost(url, formElement)
  - showResults()

- Register
  - makePost(url, formElement)
  - showRegistrationResults()

- Home
  - fetchFolders()
  - fetchDocs()
  - createFolderList(folders)
  - showAddSubFolderInput(parentElement, parentId)
  - showAddDocumentInput(parentElement, parentId)
  - addSubfolder(parentId, subfolderName, parentElement)
  - addDocument(parentId, docName, docSummary, docType, parentElement)
  - enableDragAndDrop()
  - handleDragStartDoc(event)
  - handleDragStartFolder(event, folderId)
  - handleDragOver(event)
  - handleDrop(event)
  - initTrashCan()

Eventi e Azioni → HTML

Client side		Server side	
Evento	**Azione**	**Evento**	**Azione**
Index -> login	Controllo dati	Post username e password	Controllo credenziali
Register -> sign up	Controllo validità parametri	Post informazioni user	Inserimento nel db
Home -> logout	Reindirizzamento ad Index.html	Get pagina iniziale	Invalidazione sessione
Home -> Contents	Apre la cartella selezionata	Get folder Id	Estrazione cartelle e documenti dal db
Contents -> Document	Mostra le informazioni sul documento	Get doc Id	Estrazione informazioni dal db
ContentManager -> create	Crea una nuova cartella o documento	Post informazioni	Inserimento nel db
Contents -> home -> move	Spostamento documento e reindirizzamento home (con selezione destinazione)	Get informazioni per lo spostamento	Update del db

Eventi e Azioni → RIA

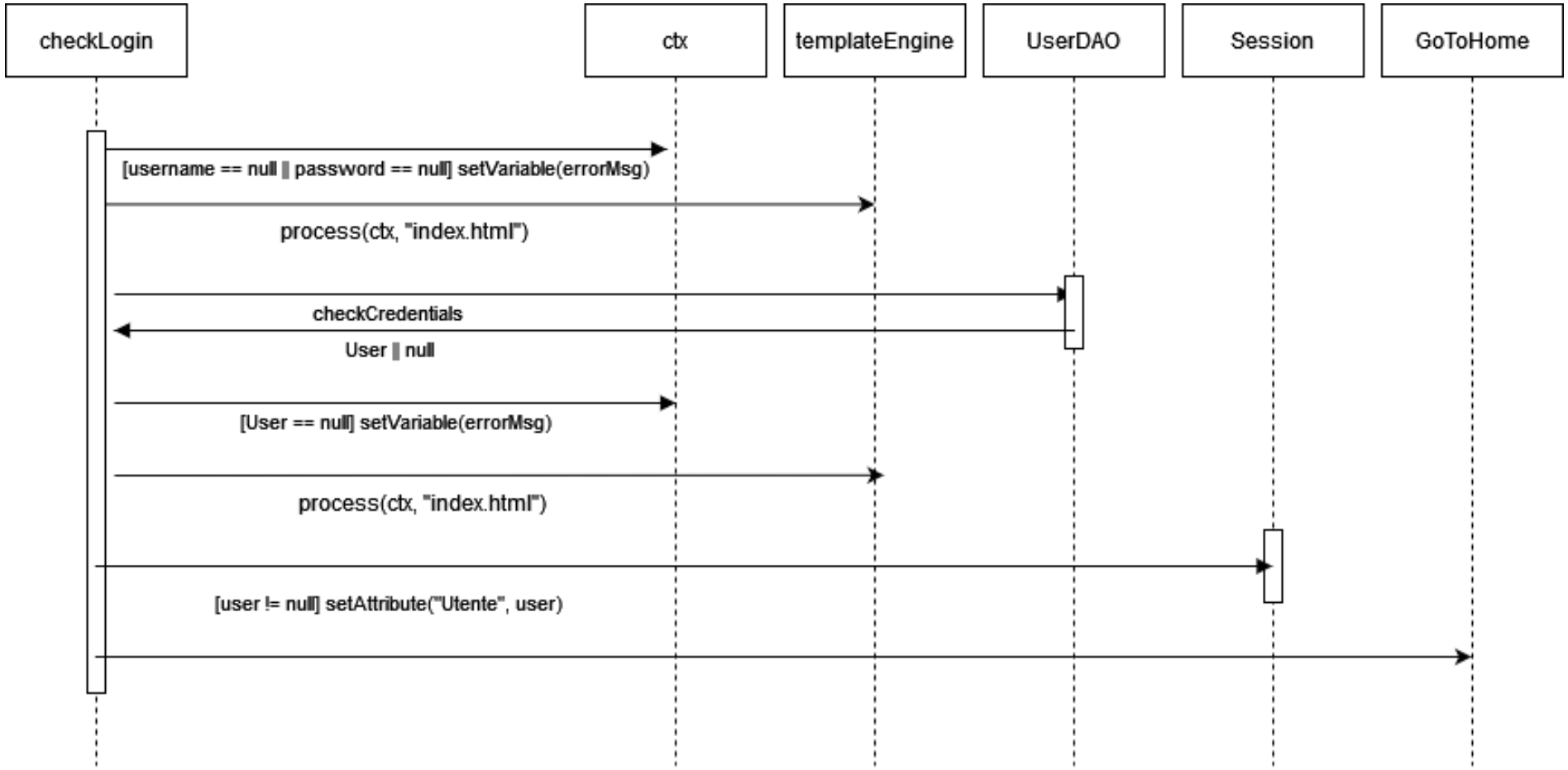
Client side		Server Side	
Evento	**Azione**	**Evento**	**Azione**
Index -> login	Controllo dati	Post Username e password	Controllo credenziali
Register -> sign up	Validità parametri	Post informazioni user	Inserimento nel db
Home -> load	Caricamento cartelle e documenti dello user	Get user id	Estrazione dal db delle cartelle e dei documenti dello user
Home -> document	Mostra le informazioni del documento	Get doc Id	Estrazione dal db dei dati del documento
Home -> drag Doc	Muove il documento selezionato		
Home -> drag Folder	Muove la cartella selezionate		
Home -> drop on trash	Cancella l'elemento selezionato	Get Id (cartella o documento)	Eliminazione dal db
Home -> drop document on folder	Sposta il documento	Get doc Id e folder Id	Update del db

Eventi e Azioni → RIA

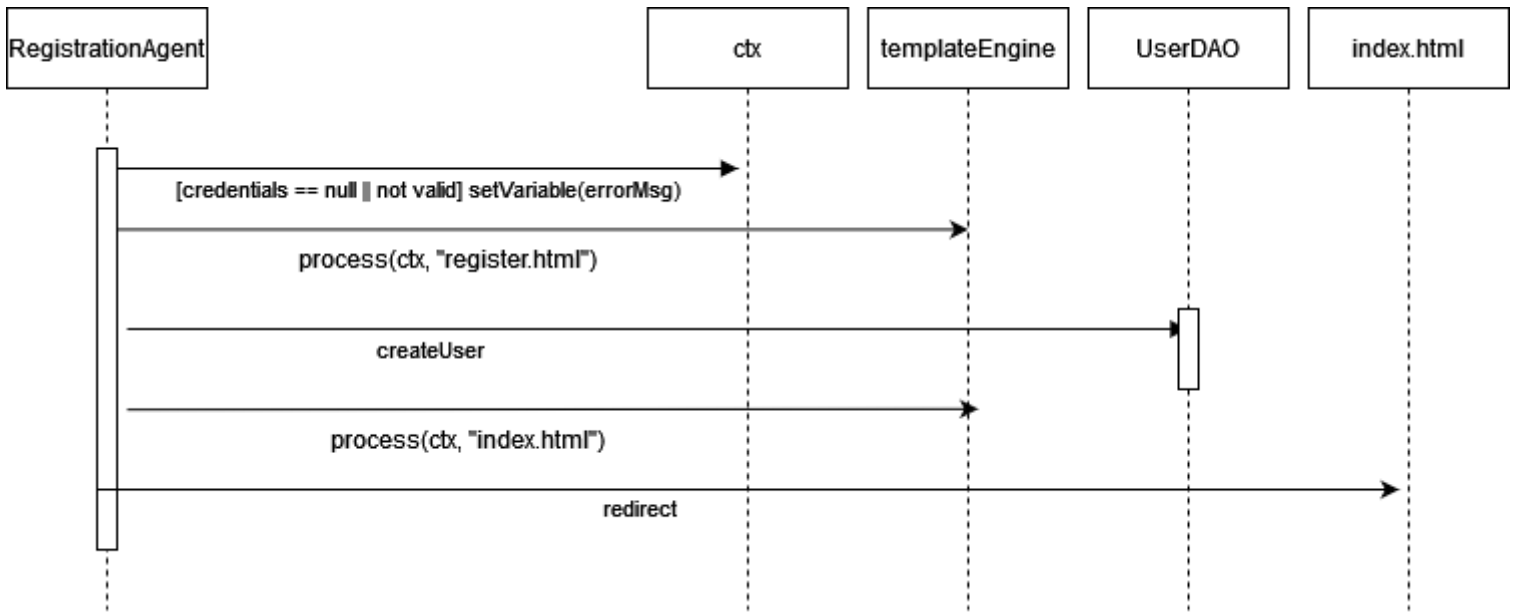
Client side		Server Side	
Evento	**Azione**	**Evento**	**Azione**
Home -> create root folder	Crea una cartella root	Post nome della cartella	Inserimento nel db
Home -> create folder	Crea una cartella nella posizione desiderata	Post user id, nome cartella e id cartella padre	Inserimento del db
Home -> create doc	Crea un documento nella cartella desiderata	Post informazioni documento e id cartella di appartenenza	Inserimento nel db
Home -> log out	Log out	Get	Logout

Sequence diagram – HTML

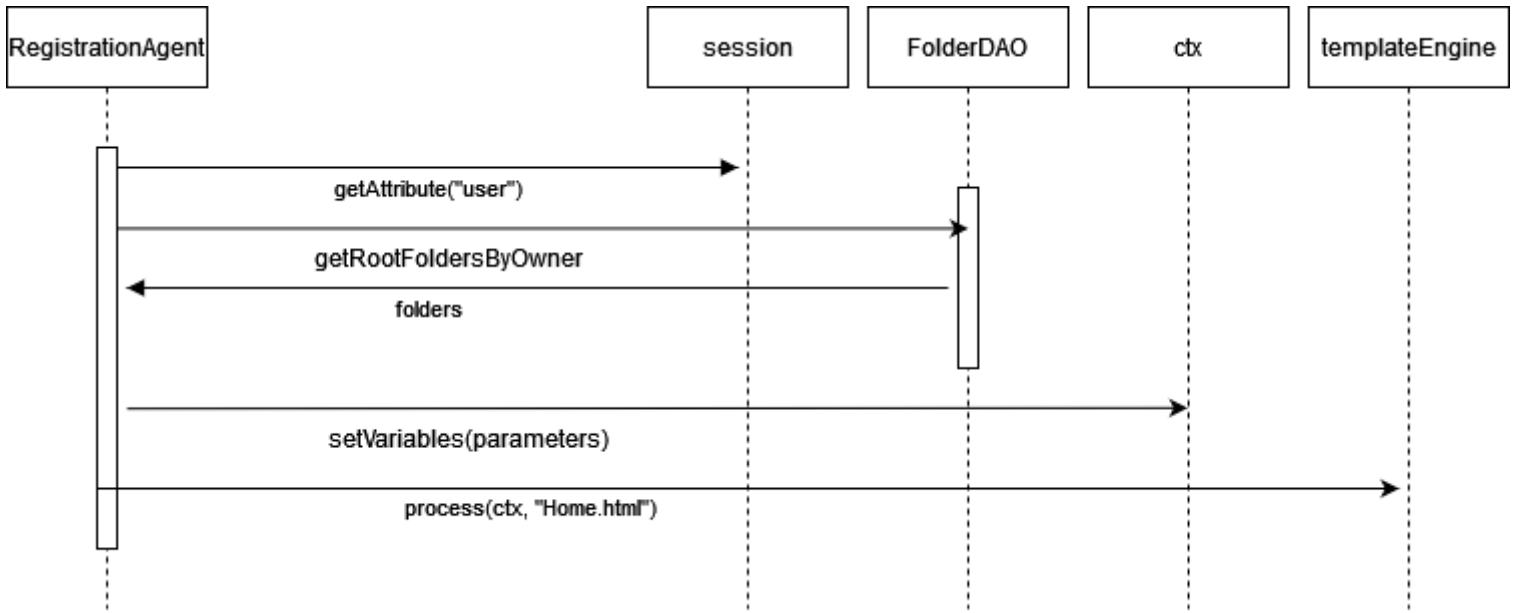
Event: login



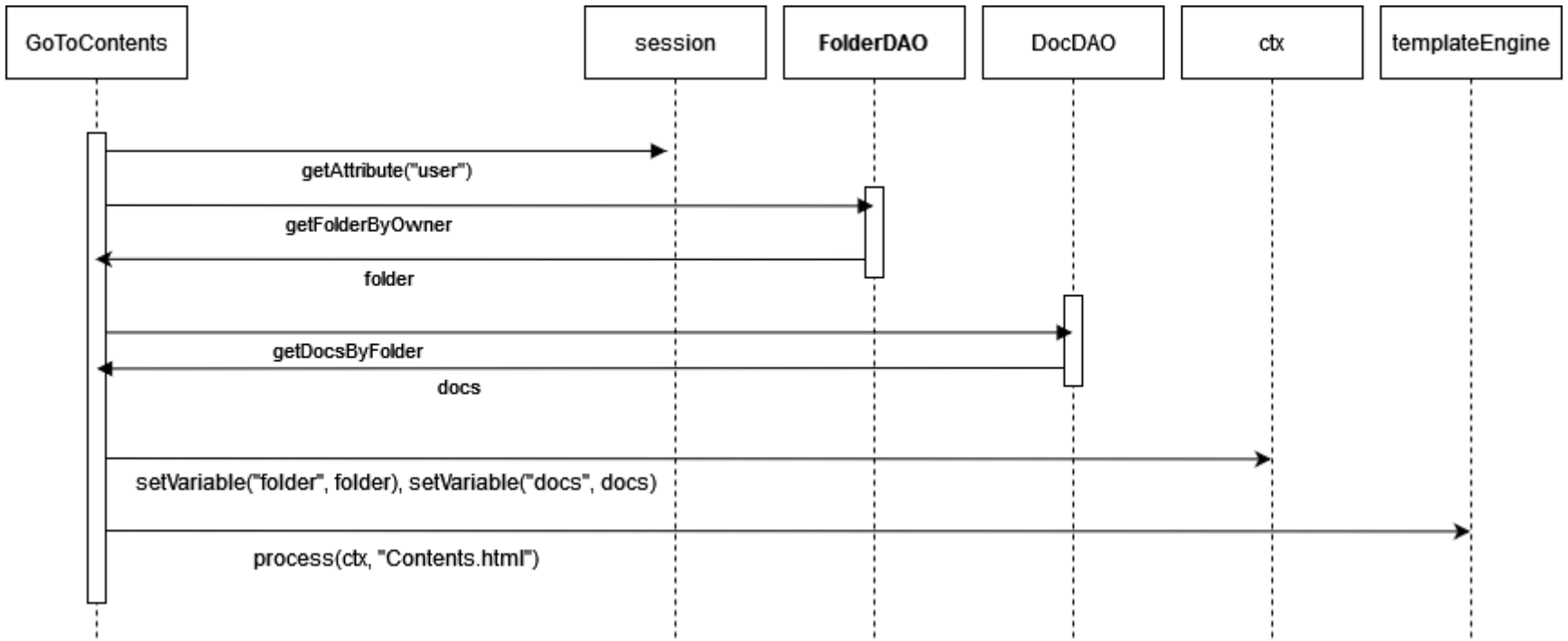
Event: registration



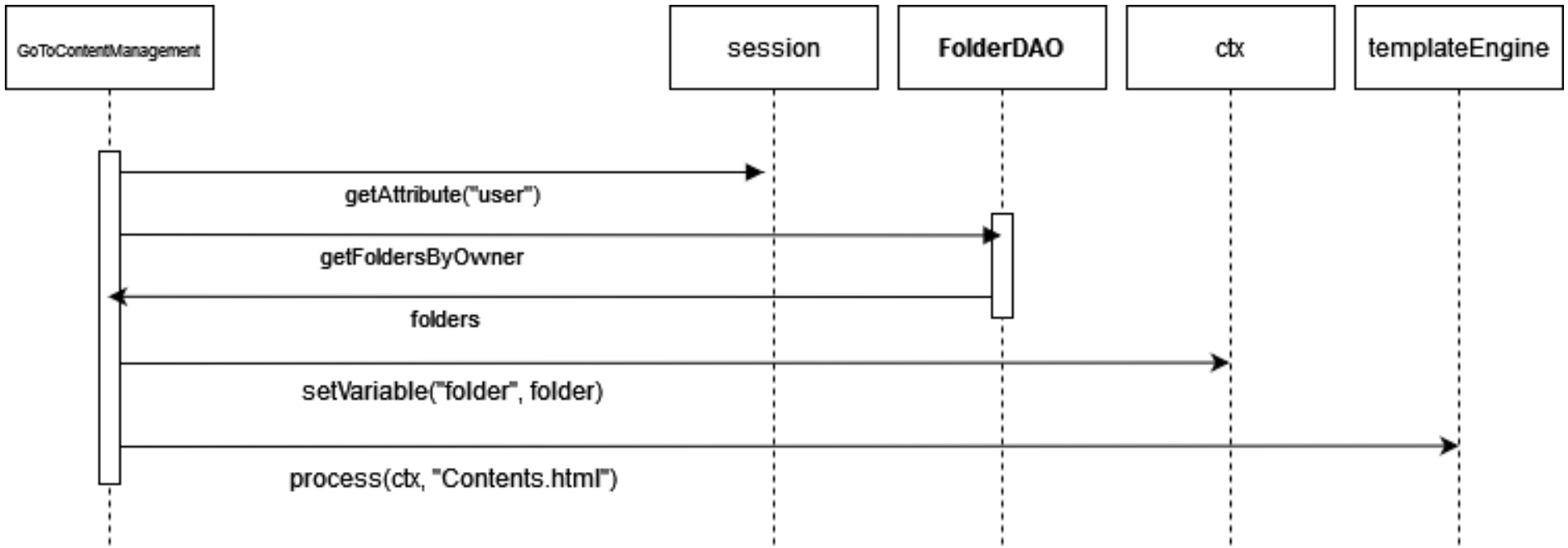
Event: GoToHome



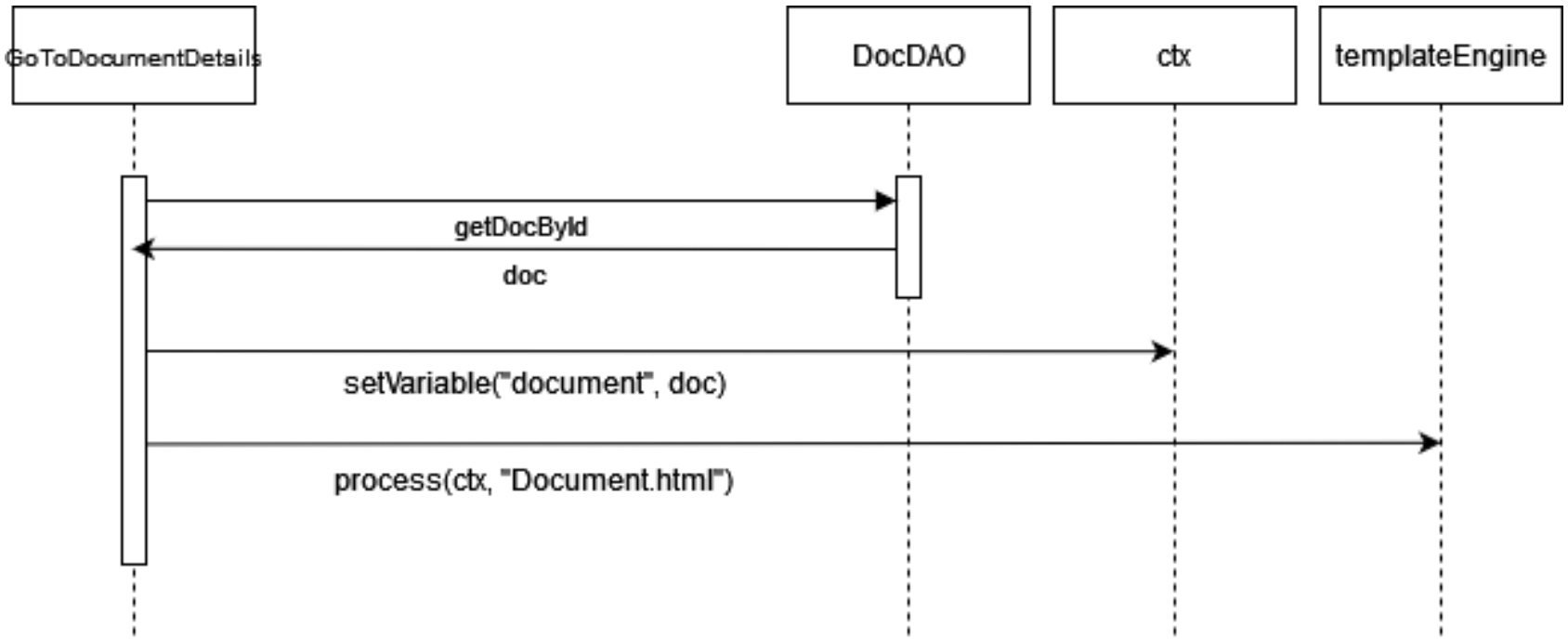
Event: GoToContents



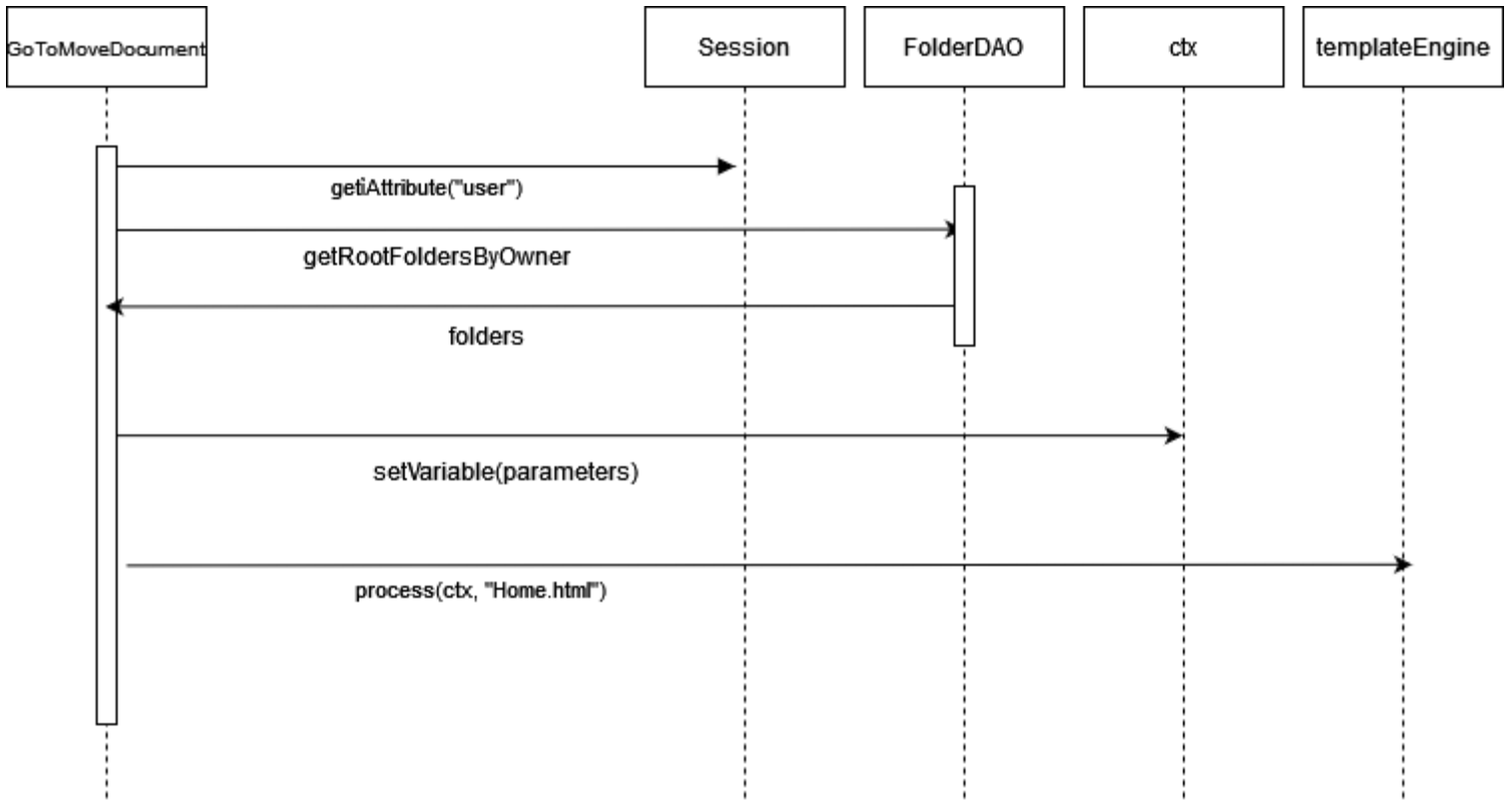
Event : GoToContentManagement



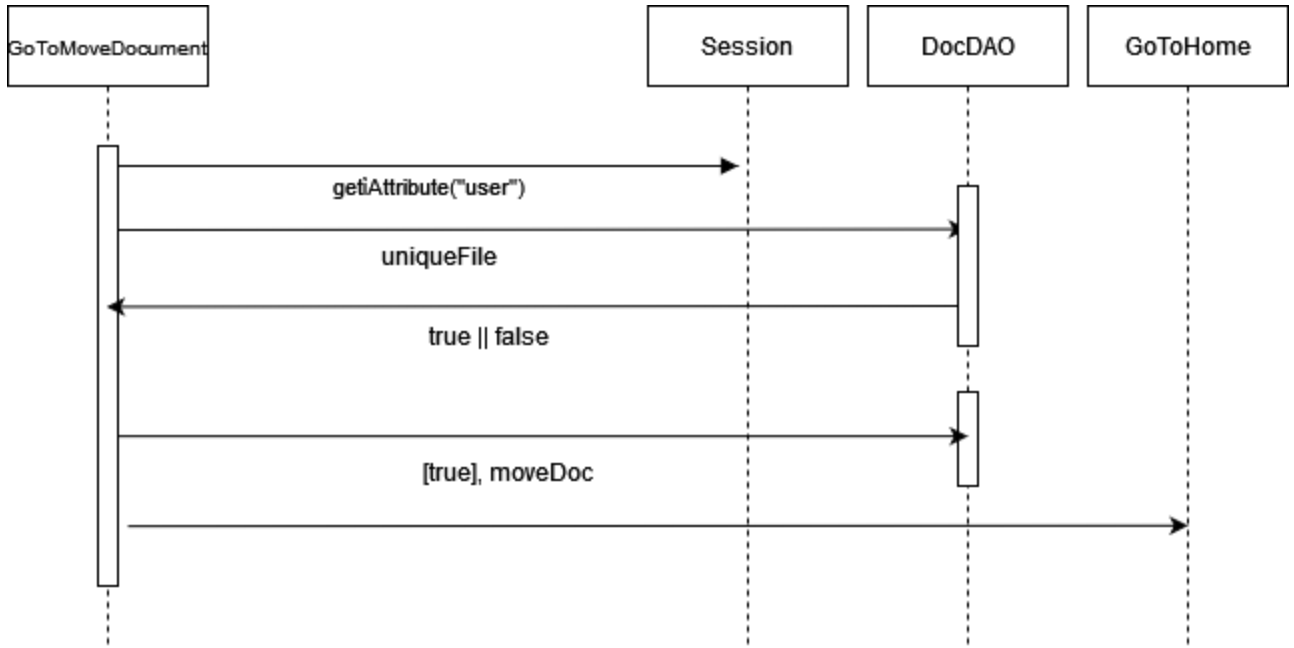
Event : GoToDocumentDetails

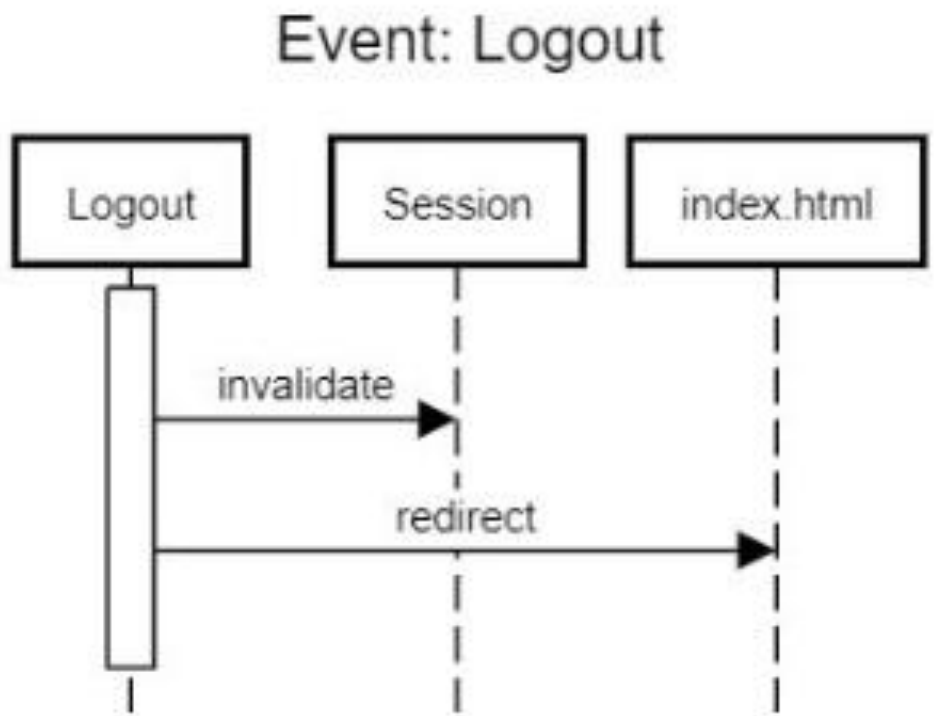


Event : GoToMoveDocument



Event : MoveDocument




Event: Logout
Logout
Session
index.html
invalidate
redirect

Eventi e gestione delle risposte

PURE HTML

- Gli eventi sono richieste HTTP
- Provenienti dal Client
- A rispondere è una servlet apposita
- L'associazione di un evento con la rispettiva servlet è specificata al di fuori dell'applicazione (web.xml)
- La risposta è sincrona, ovvero il client (momentaneamente sospeso) attende una HTTP response
- Questa response può aggiornare il contenuto della pagina corrente o forzare l'emissione di una nuova request portando dunque ad una redirect
- Tutto ciò che è relativo alla response (stato e contenuto) è gestito dal browser

RIA

- Gli eventi sono di tipo DOM o HTML API
- Provenienti dal browser oppure dal server (come HTTP response)
- A rispondere è una funzione js riportata nel corrispettivo file omonimo della pagina
- Ad associare l'evento con la rispettiva funzione è il programmatore sfruttando `«addEventListener»`
- Distinguo due tipi di eventi:
  - Eventi Interattivi:
    - La funzione che risponde opera solamente lato client oppure crea una richiesta al server tipicamente asincrona
  - Eventi di Callback:
    - La HTTP response (stato e contenuto) sono gestiti dalla funzione che si occupa dell'evento