Esercizio 3: gestione documenti
(versione RIA)

Matteo Velati – 920929

Presentazione 14 Luglio 2020

Gestione documenti

Un'applicazione web consente la gestione di cartelle, sottocartelle e documenti online. Una cartella ha un nome e una data di creazione e può contenere (solo) sottocartelle. Una sottocartella può contenere (solo) dei documenti. Un documento ha un 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 cartelle e delle sottocartelle.

Nell'HOME page l'utente può selezionare una sottocartella e accedere a una pagina DOCUMENTI che mostra l'elenco dei documenti di una sottocartella. Ogni documento in elenco ha due link: accedi e sposta. Quando l'utente seleziona il link accedi, appare una pagina DOCUMENTO che mostra tutti i dati del documento selezionato. Quando l'utente seleziona il link sposta, appare la HOME PAGE con l'albero delle cartelle e delle sottocartelle; in questo caso la pagina mostra il messaggio "*Stai spostando il documento X dalla sottocartella Y. Scegli la sottocartella di destinazione*", la sottocartella 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 sottocartella di destinazione, il documento è spostato dalla sottocartella di origine a quella di destinazione e appare la pagina DOCUMENTI che mostra il contenuto aggiornato della sottocartella di destinazione. (continua)

Gestione documenti

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

- L'applicazione supporta registrazione e login di utenti mediante una pagina pubblica con opportune form. La registrazione controlla la validità sintattica dell'indirizzo di email e l'uguaglianza tra i campi "password" e "ripeti password", anche a lato client. La registrazione controlla l'unicità dello username.
- Dopo il login, 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 un documento è realizzata mediante drag and drop.
- Si aggiunge una cartella denominata "cestino". Il drag & drop di un documento o di una cartella nel cestino comporta la cancellazione. Prima di inviare il comando di cancellazione al server l'utente vede una finestra modale di conferma e può decidere se annullare l'operazione o procedere.

Analisi dei dati

Un'applicazione web consente la gestione di cartelle, sottocartelle e documenti online. Una cartella ha un nome e una data di creazione e può contenere (solo) sottocartelle. Una sottocartella può contenere (solo) dei documenti. Un documento ha un 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 cartelle e delle sottocartelle.

Nell'HOME page l'utente può selezionare una sottocartella e accedere a una pagina DOCUMENTI che mostra l'elenco dei documenti di una sottocartella. Ogni documento in elenco ha due link: accedi e sposta. Quando l'utente seleziona il link accedi, appare una pagina DOCUMENTO che mostra tutti i dati del documento selezionato. Quando l'utente seleziona il link sposta, appare la HOME PAGE con l'albero delle cartelle e delle sottocartelle; in questo caso la pagina mostra il messaggio "*Stai spostando il documento X dalla sottocartella Y. Scegli la sottocartella di destinazione*", la sottocartella 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 sottocartella di destinazione, il documento è spostato dalla sottocartella di origine a quella di destinazione e appare la pagina DOCUMENTI che mostra il contenuto aggiornato della sottocartella di destinazione.

Entities, attributes, relationships

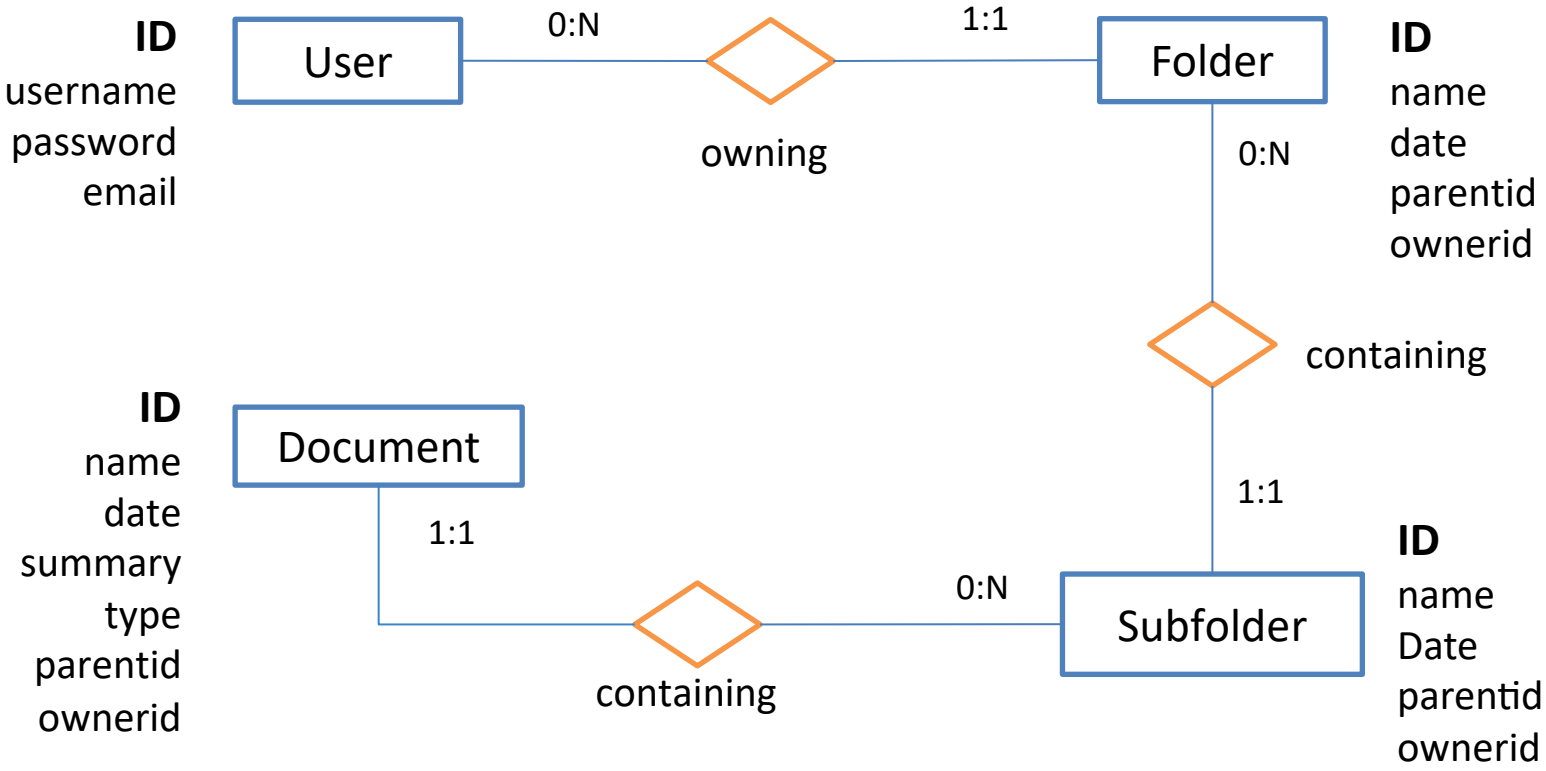
Analisi dei dati

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

- L'applicazione supporta registrazione e login di utenti mediante una pagina pubblica con opportune form. La registrazione controlla la validità sintattica dell'indirizzo di email e l'uguaglianza tra i campi "password" e "ripeti password", anche a lato client. La registrazione controlla l'unicità dello username.
- Dopo il login, 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 un documento è realizzata mediante drag and drop.
- Si aggiunge una cartella denominata "cestino". Il drag & drop di un documento o di una cartella nel cestino comporta la cancellazione. Prima di inviare il comando di cancellazione al server l'utente vede una finestra modale di conferma e può decidere se annullare l'operazione o procedere.

Entities, attributes, relationships

Database design



Local database schema

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

```
CREATE TABLE `folder` (
  `id` int NOT NULL AUTO_INCREMENT,
  `name` varchar(45) NOT NULL,
  `date` date NOT NULL,
  `ownerid` int NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `name_UNIQUE` (`name`),
  KEY `id_owner_idx` (`ownerid`),
  CONSTRAINT `id_owner_fold` FOREIGN
        KEY (`ownerid`) REFERENCES
        `user` (`id`) ON UPDATE CASCADE
)
```

Local database schema

```
CREATE TABLE `subfolder` (
  `id` int NOT NULL AUTO_INCREMENT,
  `name` varchar(45) NOT NULL,
  `date` date NOT NULL,
  `parentid` int NOT NULL,
  `ownerid` int NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `name_UNIQUE` (`name`),
  KEY `id_subfolder` (`parentid`),
  KEY `id_owner_idx` (`ownerid`),
  CONSTRAINT `id_owner_sub` FOREIGN KEY (`ownerid`) REFERENCES
        `user` (`id`) ON UPDATE CASCADE,
  CONSTRAINT `id_subfolder` FOREIGN KEY (`parentid`) REFERENCES
        `folder` (`id`) ON DELETE CASCADE ON UPDATE CASCADE
)
```

Local database schema

```
CREATE TABLE `document` (
  `id` int NOT NULL AUTO_INCREMENT,
  `name` varchar(45) NOT NULL,
  `date` date NOT NULL,
  `summary` varchar(128) NOT NULL,
  `type` varchar(45) NOT NULL,
  `parentid` int NOT NULL,
  `ownerid` int NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `name_UNIQUE` (`name`),
  KEY `id_document` (`parentid`),
  KEY `id_owner_doc_idx` (`ownerid`),
  CONSTRAINT `id_document` FOREIGN KEY (`parentid`) REFERENCES
         `subfolder` (`id`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `id_owner_doc` FOREIGN KEY (`ownerid`) REFERENCES `user`
         (`id`) ON UPDATE CASCADE
)
```

Application requirements analysis

Un'applicazione web consente la gestione di cartelle, sottocartelle e documenti online. Una cartella ha un nome e una data di creazione e può contenere (solo) sottocartelle. Una sottocartella può contenere (solo) dei documenti. Un documento ha un 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 cartelle e delle sottocartelle.

Nell' HOME PAGE l'utente può selezionare una sottocartella e accedere a una pagina DOCUMENTI che mostra l'elenco dei documenti di una sottocartella. Ogni documento in elenco ha due link: accedi e sposta. Quando l'utente seleziona il link accedi, appare una pagina DOCUMENTO che mostra tutti i dati del documento selezionato. Quando l'utente seleziona il link sposta, appare la HOME PAGE con l'albero delle cartelle e delle sottocartelle; in questo caso la pagina mostra il messaggio "*Stai spostando il documento X dalla sottocartella Y. Scegli la sottocartella di destinazione*", la sottocartella 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 sottocartella di destinazione, il documento è spostato dalla sottocartella di origine a quella di destinazione e appare la pagina DOCUMENTI che mostra il contenuto aggiornato della sottocartella di destinazione.

Pages (views), view components, events, actions

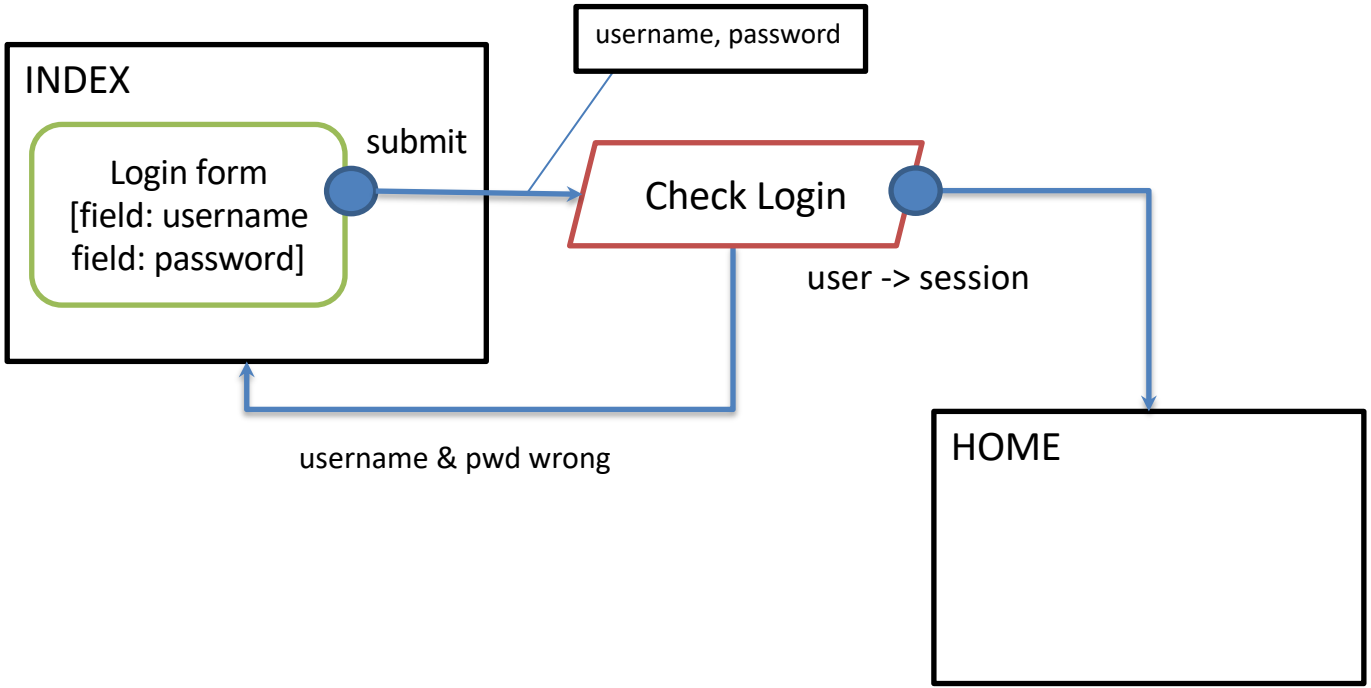
Application requirements analysis

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

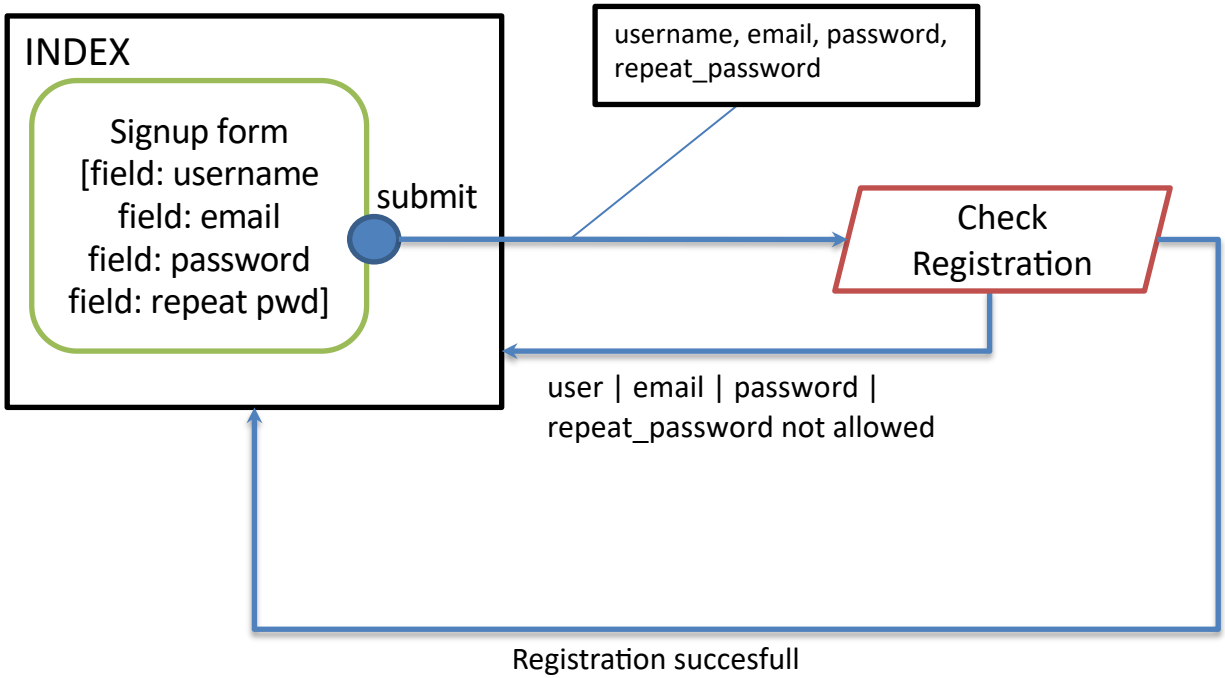
- L'applicazione supporta registrazione e login di utenti mediante una pagina pubblica con opportune form. La registrazione controlla la validità sintattica dell'indirizzo di email e l'uguaglianza tra i campi "password" e "ripeti password", anche a lato client. La registrazione controlla l'unicità dello username.
- Dopo il login, 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 un documento è realizzata mediante drag and drop.
- Si aggiunge una cartella denominata "cestino". Il drag & drop di un documento o di una cartella nel cestino comporta la cancellazione. Prima di inviare il comando di cancellazione al server l'utente vede una finestra modale di conferma e può decidere se annullare l'operazione o procedere.

Pages (views), view components, events, actions

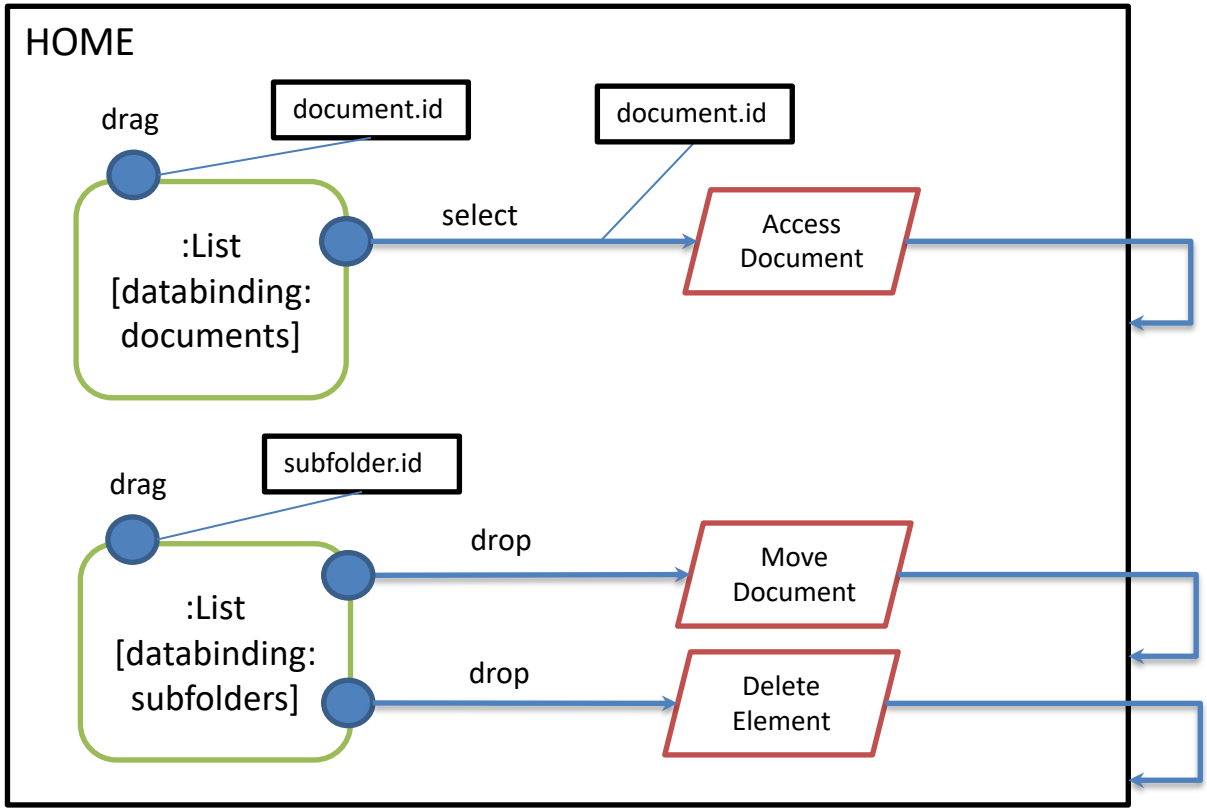
Application design



Application design



Application design



Eventi & azioni

Client side		Server side	
Evento	**Azione**	**Evento**	**Azione**
index → login form → submit	Controllo dati	POST (username password)	Controllo credenziali
index → signup form → submit	Controllo dati	POST (dati form)	Aggiunta nuovo user
Home → load	Aggiorna view albero	GET ()	Estrazione di cartelle, sottocartelle, documenti
Home → drag & drop documento → sottocartella	Cambio view albero	POST (documentid)	Spostamento documento
Home → drag & drop documento → cestino	Comparsa finestra modale		
Home → drag & drop sottocartella → cestino	Comparsa finestra modale		
Documento → select	Aggiorna view dati documento	GET (documentid)	Estrazione dati documento
Modal → confirm	Aggiorna view albero	POST (elementid)	Eliminazione elemento (sottocartella / documento)

Controller / event handler

Client side		Server side	
Evento	**Controller**	**Evento**	**Controller**
index → login form → submit	Function makeCall	POST (username password)	CheckLogin (servlet)
index → signup form → submit	Function makeCall	POST (dati form)	CheckRegistration (servlet)
Home → load	Function pageManager	GET ()	GetFolders (servlet) GetSubfolders (servlet) GetDocuments (servlet)
Home → drag & drop documento → sottocartella	Function tree	POST (documentid)	MoveDocument (servlet)
Home → drag & drop documento → cestino	Function tree		
Home → drag & drop sottocartella → cestino	Function tree		
Documento → select	Function tree	GET (documentid)	AccessDocument (servlet)
Modal → confirm	Function makeCall	POST (elementid)	DeleteElement (servlet)

Server side: model objects, controllers, DAO

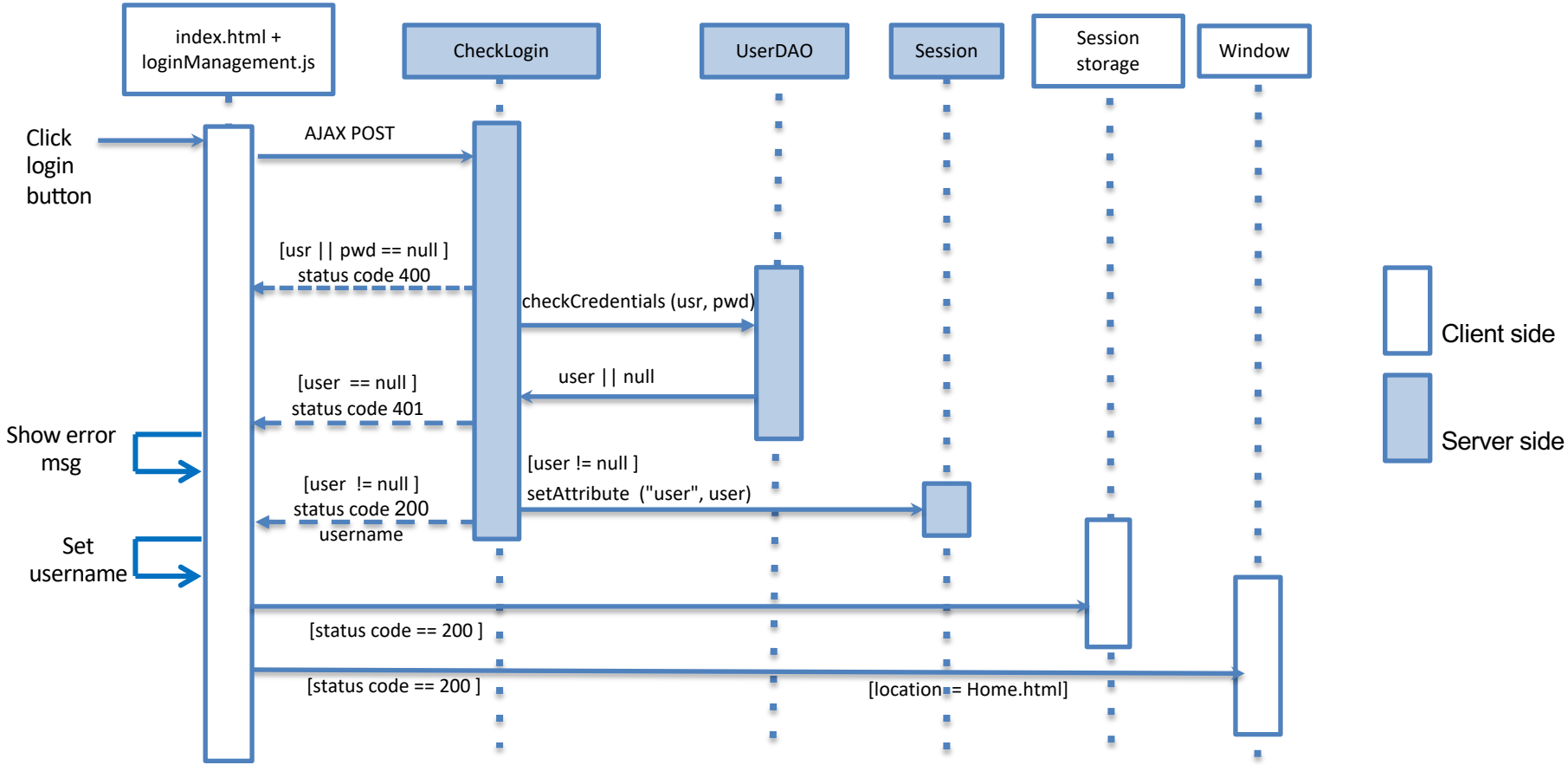
- Model objects (Beans)
  - User
  - Folder
  - Subfolder
  - Document
- Controllers (servlets)
  - CheckLogin
  - CheckRegistration
  - AccessDocument
  - DeleteDocument
  - MoveDocument
  - GetFolders
  - GetSubfolders
  - GetDocuments
  - Logout

- Data Access Objects (Classes)
  - UserDAO
    - addUser(usrn, pwd, email)
    - checkUsernameIsOk(usrn)
    - checkCredentails(usrn, pwd)
  - FolderDAO
    - findAllFolders(userid)
  - SubfolderDAO
    - findAllSubfoldersByParentID(folID)
    - findSubfolderByID(subID)
    - deleteSubfolder(subID)
  - DocumentDAO
    - findAllDocumentsByParentID(parentID)
    - findDocumentByID(docID)
    - moveDocument (docID, subID)
    - deleteDocumentByID(docID)

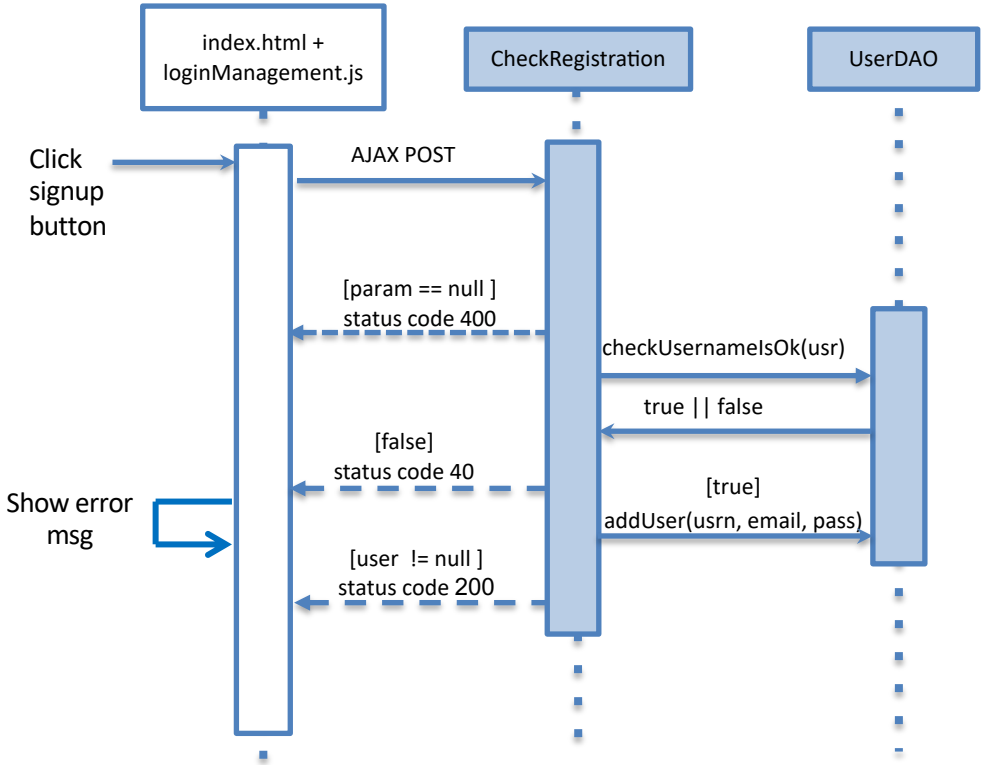
Client side: view & view components

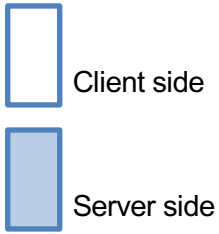
- Index
  - Login form
    - Gestione del submit ed errori
  - Sign up form
    - Gestione del submit ed errori
- Home
  - PageManager
    - start(): crea e inizializza i componenti dell'interfaccia
    - refresh(): aggiorna la visualizzazione dei componenti
- Home
  - Tree
    - showlevel1(): richiede al server la lista delle cartelle
    - showlevel2(): richiede al server la lista delle sottocartelle e popola la vista con le cartelle
    - showlevel3(): richiede al server la lista dei documenti e popola la vista con sottocartelle e documenti
    - registerevents(): associa i listener al componente per gestire la funzione di drag & drop
    - movenode(): invia dati al server e aggiorna la vista
    - removenode(): invia dati server e aggiorna la vista
    - update(): ripulisce e aggiorna l'interfaccia
  - DocumentDetails
    - show(): richiede i dati di un documento al server
    - update(): riceve i dati dal server e aggiorna l'interfaccia
    - reset(): ripulisce l'interfaccia

Evento: Login

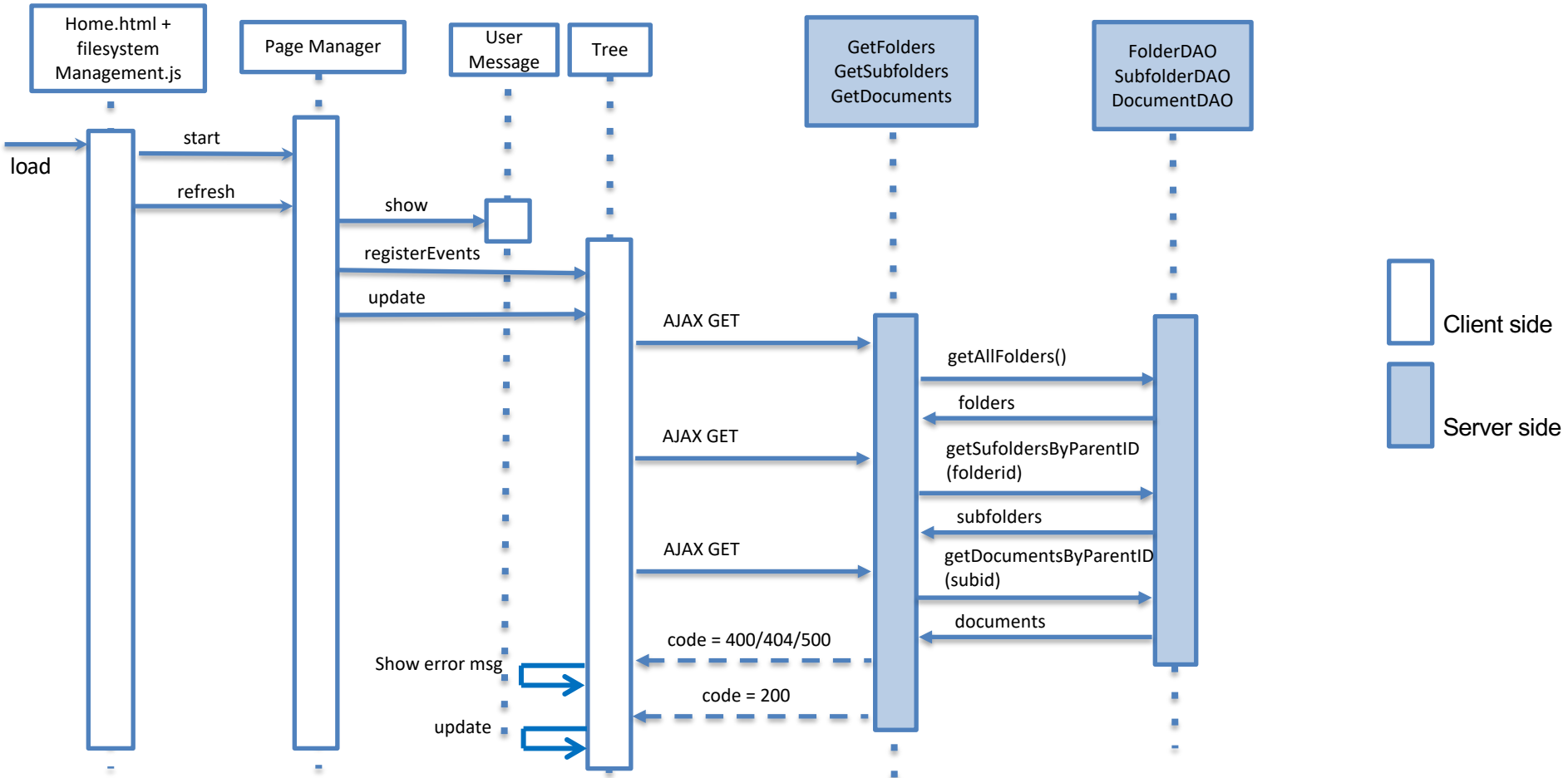


Evento: Singup

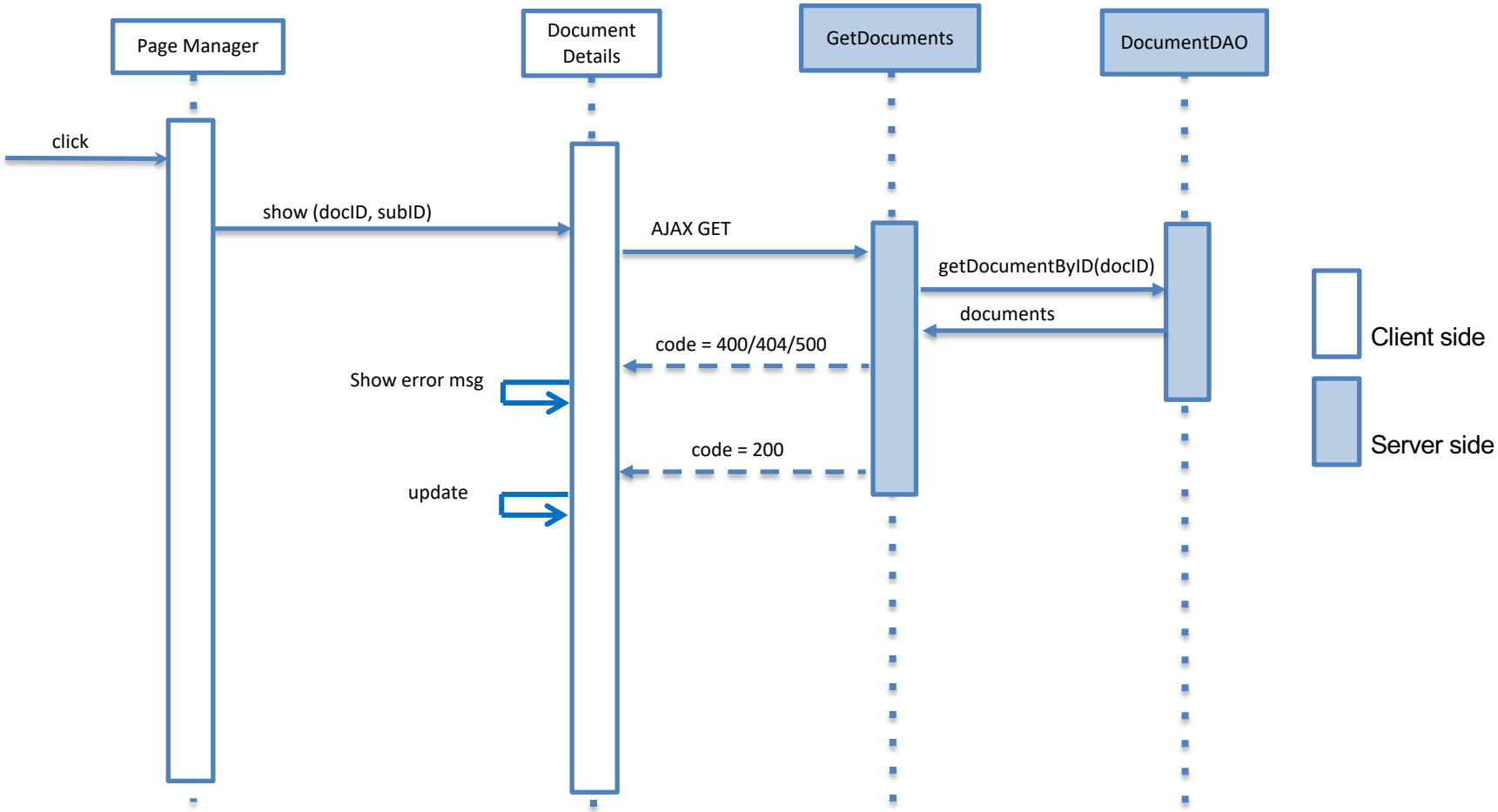




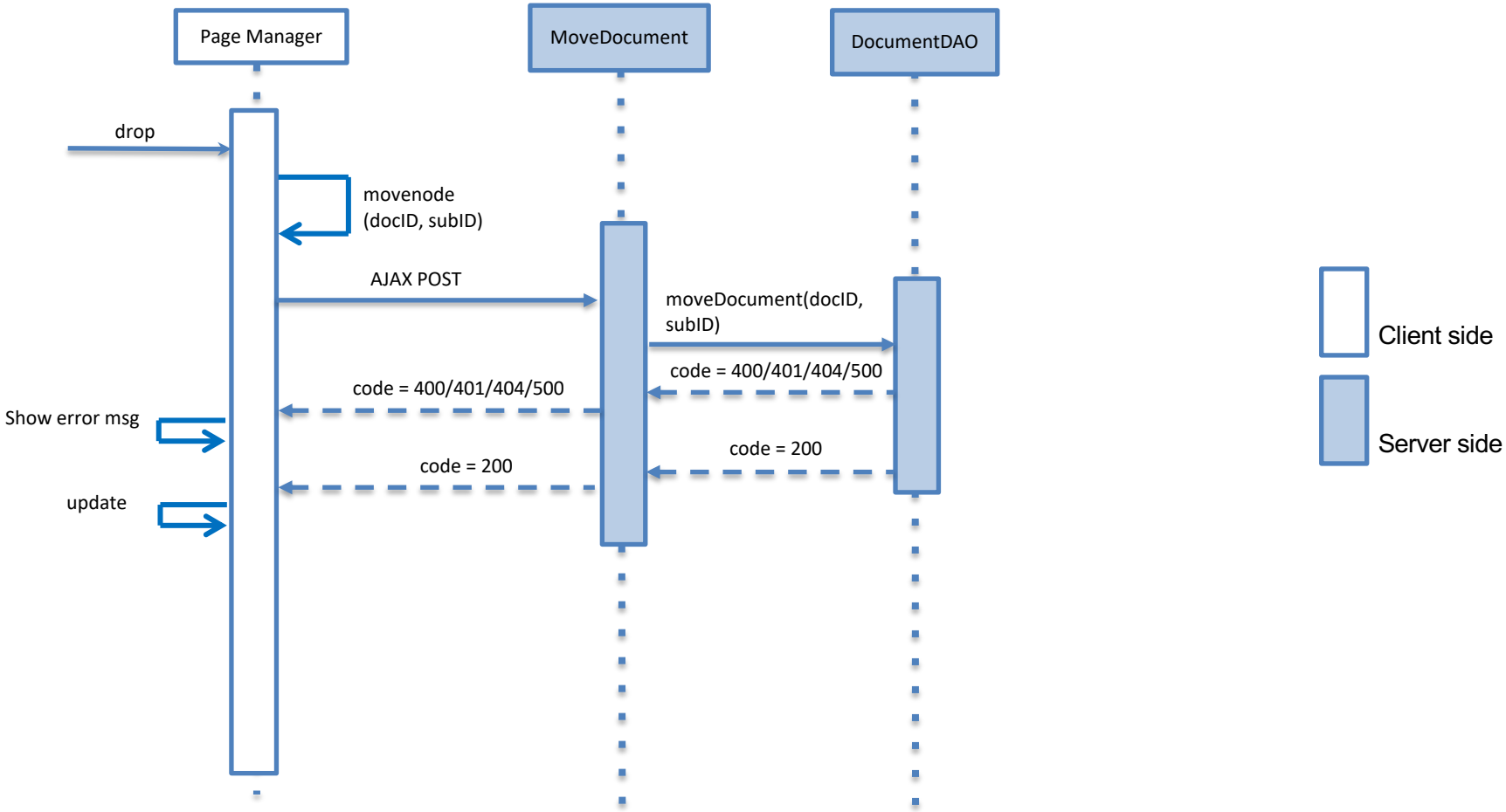
Evento: Load


Home.html + filesystem Management.js
Page Manager
User Message
Tree
GetFolders GetSubfolders GetDocuments
FolderDAO SubfolderDAO DocumentDAO
load
start
refresh
show
registerEvents
update
AJAX GET
getAllFolders()
folders
AJAX GET
getSufoldersByParentID (folderid)
subfolders
AJAX GET
getDocumentsByParentID (subid)
documents
code = 400/404/500
Show error msg
code = 200
update
Client side
Server side

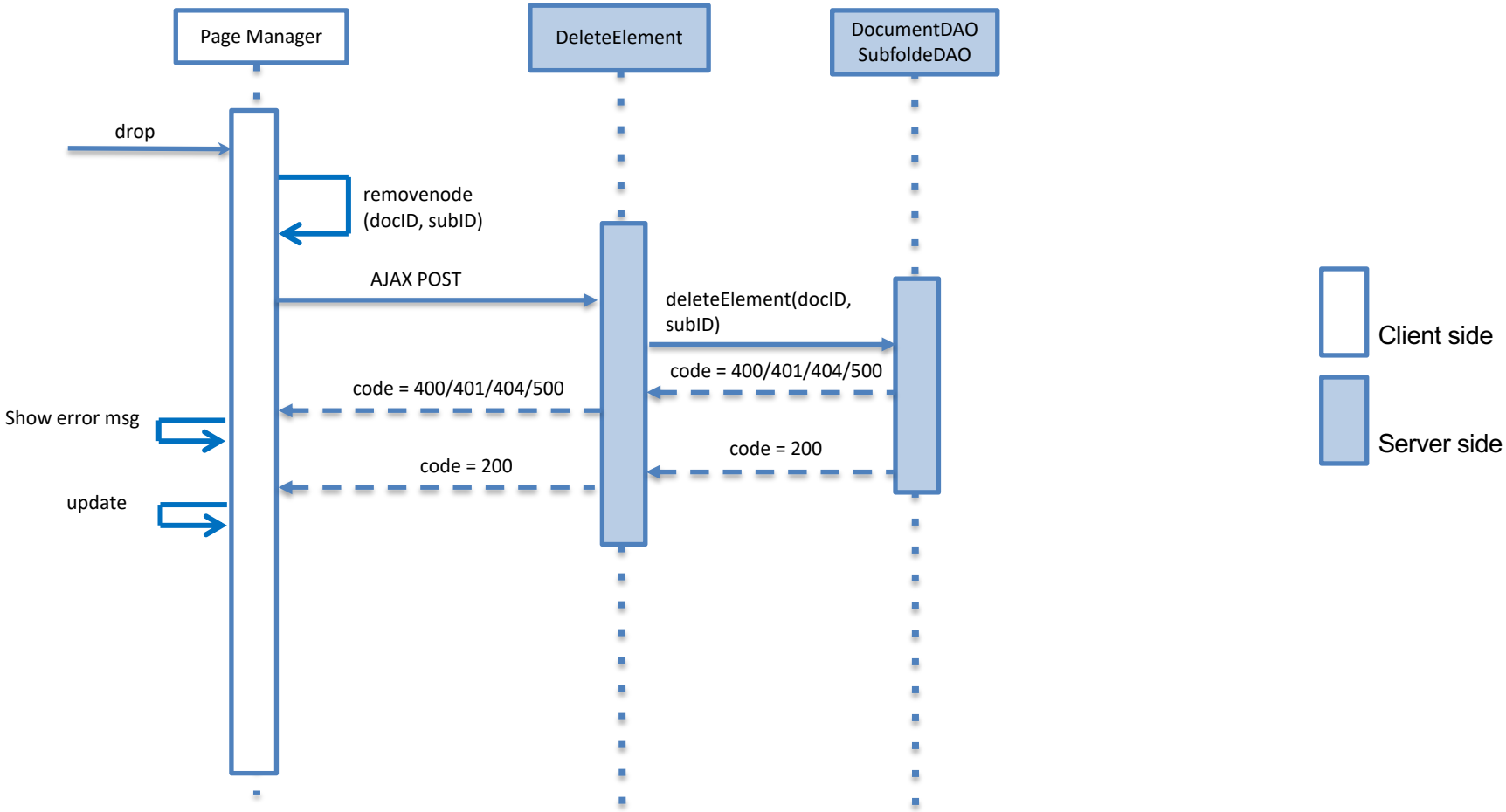
Evento: Access Document



Evento: Move Document



Evento: Delete Element



Evento: logout