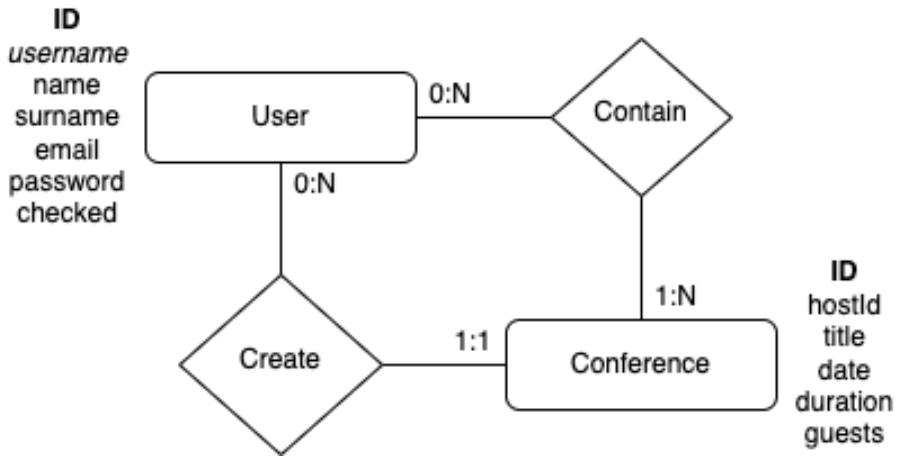
ANALISI DEI DATI

L'applicazione supporta registrazione e login 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". La registrazione controlla l'unicità dello username. Una riunione ha un titolo, una data, un'ora, una durata e un numero massimo di partecipanti. L'utente fa il login e, se autenticato, accede all'HOME page che mostra l'elenco delle riunioni indette da lui e non ancora scadute, l'elenco delle riunioni cui è stato invitato e non ancora scadute, e una form per creare una nuova riunione. Quando l'utente inoltra la form con il bottone INVIA, appare una pagina ANAGRAFICA con l'elenco degli utenti registrati. L'utente può scegliere uno o più partecipanti dall'elenco e premere il bottone INVITA per invitarli alla riunione. Se il numero d'invitati è superiore di X unità rispetto al massimo ammissibile, appare di nuovo la pagina ANAGRAFICA con un messaggio "Troppi utenti selezionati, eliminarne almeno X". La pagina evidenzia nell'elenco gli utenti scelti in precedenza come preselezionati, in modo che l'utente possa deselezionarne alcuni. Se alla pressione del bottone INVITA il numero d'invitati è inferiore al massimo ammissibile, la riunione è memorizzata nella base di dati e associata agli utenti invitati e l'utente è rimandato alla HOME PAGE. Al terzo tentativo scorretto di assegnare troppi invitati a una riunione appare una pagina CANCELLAZIONE con un messaggio "Tre tentativi di definire una riunione con troppi partecipanti, la riunione non sarà creata" e un link per tornare all'HOME PAGE. In questo caso la riunione NON è memorizzata nella base di dati. L'applicazione non deve registrare nella base di dati riunioni con numero eccessivo di partecipanti. L'applicazione consente il logout dell'utente.

Entities, attributes, relationships

DATABASE DESIGN



LOCAL DATABASE SCHEMA

```
CREATE TABLE `users` (
  `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,
  `email` varchar(45) NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `username_UNIQUE` (`username`)
)
```

```
CREATE TABLE `conferences` (
  `id` int NOT NULL AUTO_INCREMENT,
  `host` int NOT NULL,
  `title` varchar(45) NOT NULL,
  `date` timestamp NOT NULL,
  `duration` time NOT NULL,
  `guests` int NOT NULL,
  PRIMARY KEY (`id`)
)
```

```
CREATE TABLE `guests` (
  `id` int NOT NULL AUTO_INCREMENT,
  `idconference` int NOT NULL,
  `idguest` int NOT NULL,
  PRIMARY KEY (`id`)
)
```

APPLICATION REQUIREMENTS ANALYSIS

Un'applicazione web consente la gestione di riunioni online. L'applicazione supporta registrazione e login mediante una pagina pubblica con opportune form(x2). La registrazione 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 riunione ha un titolo, una data, un'ora, una durata e un numero massimo di partecipanti. L'utente fa il login e, se autenticato, accede all'HOME page che mostra l'elenco delle riunioni indette da lui e non ancora scadute, l'elenco delle riunioni cui è stato invitato e non ancora scadute, e una form per creare una nuova riunione. Quando l'utente inoltra la form con il bottone INVIA, appare una pagina ANAGRAFICA con l'elenco degli utenti registrati. L'utente può scegliere uno o più partecipanti dall'elenco e premere il bottone INVITA per invitarli alla riunione. Se il numero d'invitati è superiore di X unità rispetto al massimo ammissibile, appare di nuovo la pagina ANAGRAFICA con un messaggio "Troppi utenti selezionati, eliminarne almeno X". La pagina evidenzia nell'elenco gli utenti scelti in precedenza come preselezionati, in modo che l'utente possa deselezionarne alcuni. Se alla pressione del bottone INVITA il numero d'invitati è inferiore al massimo ammissibile, la riunione è memorizzata nella base di dati e associata agli utenti invitati e l'utente è rimandato alla HOME PAGE. Al terzo tentativo scorretto di assegnare troppi invitati a una riunione appare una pagina CANCELLAZIONE con un messaggio "Tre tentativi di definire una riunione con troppi partecipanti, la riunione non sarà creata" e un link per tornare all'HOME PAGE. In questo caso la riunione NON è memorizzata nella base di dati. L'applicazione non deve registrare nella base di dati riunioni con numero eccessivo di partecipanti. L'applicazione consente il logout dell'utente.

Versione con JavaScript

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

- L'applicazione supporta registrazione e login 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 scelta dall'anagrafica deve essere realizzata con una pagina modale con i bottoni invia e cancella. NB: è una finestrella che si apre per dare una qualche scelta o un qualche messaggio all'utente: https://it.wikipedia.org/wiki/Finestra_modale
- I controlli di correttezza del numero di invitati e del massimo numero di tentativi, con i relativi messaggi di avvertimento, devono essere realizzati anche a lato client.
- Lo stato dell'interazione (numero di tentativi) deve essere memorizzato a lato client.

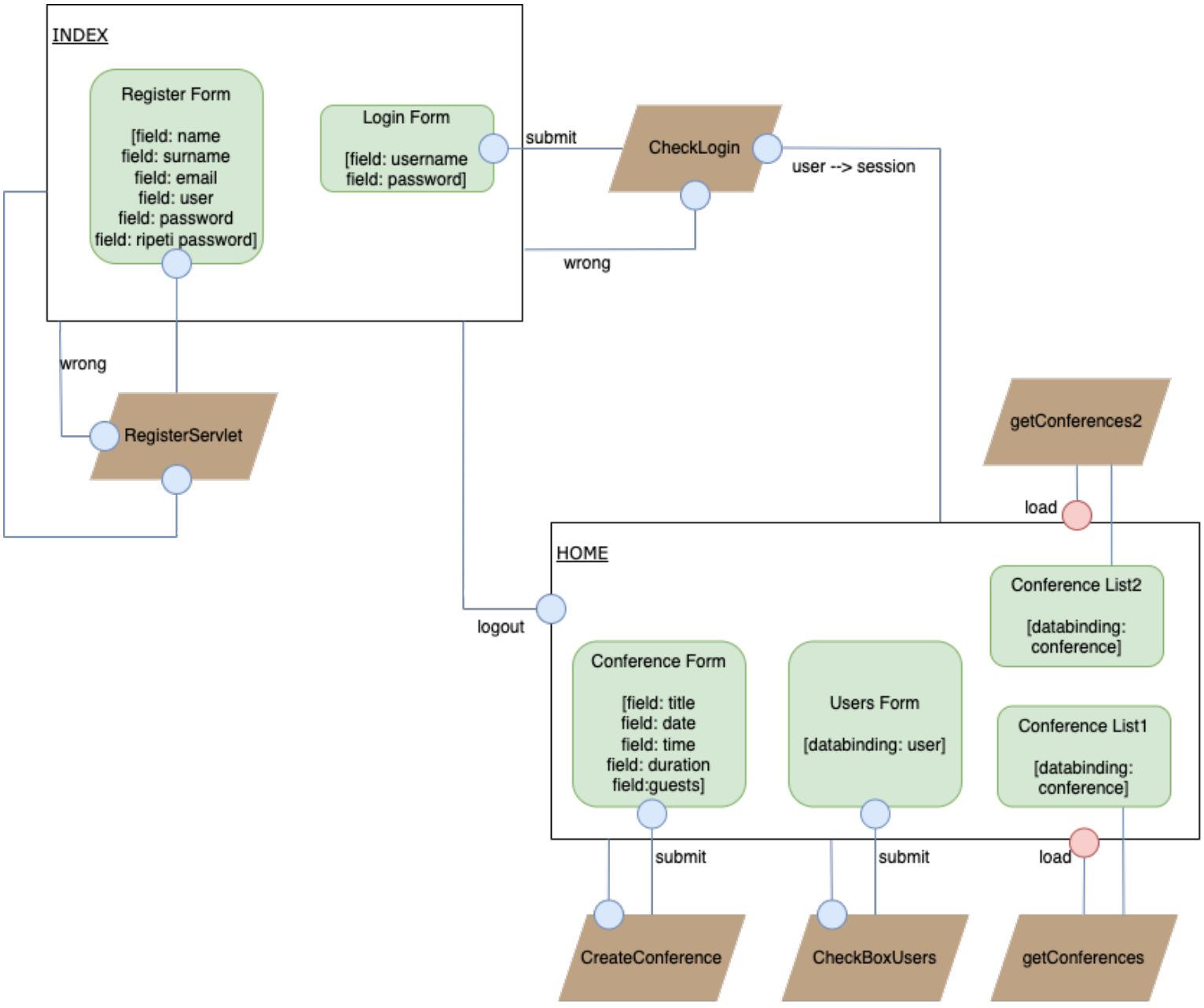
Pages (views), view components, events, actions

COMPLETAMENTO DELLE SPECIFICHE

- Il bottone di Logout reindirizza alla pagina pubblica
- Tutte le form non accettano campi vuoti
- Le riunioni non possono essere create per date del passato

SOMMARIO

APPLICATION DESIGN



EVENTI E AZIONI

Client side		Server side	
Evento	**Azione**	**Evento**	**Azione**
index → login form → submit	Controllo dati	POST username, password	- Controllo credenziali
index → register form → submit	Controllo dati	POST name, surname, email, username, oassword, passord2	- Controllo dati - Inserimento utente nel database
Home page→ load (ps: due chiamate diverse)	Aggiorna le due liste con i dati delle conferenze	GET (nessun parametro)	- Estrazione conferenze dell'utente - Estrazione conferenze a cui è stato invitato
Wizard → Create conference form	Controllo dati	POST title, date, time, duration, guests	- Controllo dati - Salvataggio conference nella sessione - Invio lista guests
Wizard → select guests	- Controllo numero guests - Controllo numero tentativi	POST (lista userId selezionati)	- Controllo dati - Inserimento conference nel database - Inserimento guests nel database
Logout		GET (nessun parametro)	Terminazione della sessione

Client side		Server side	
Evento	**Controllore**	**Evento**	**Controllore**
index → login form → submit	Function makeCall	POST username, password	CheckLogin (servlet)
Index → register form → submit	Function makeCall	POST name, surname, email, username, password, password2	RegisterServlet (servlet)
Home page → load (ps: due chiamate diverse)	Functions: - PageOrchestrator → - ConferencesList - ConferencesList2	GET (nessun parametro)	- getConferences (servlet) - getConferences2 (servlet)
Wizard → create conference form → submit	Function makeCall	- POST title, date, time, duration, guests - Ritorna lista users come file json	CreateConference (servlet)
-	Functions: - PageOrchestrator → - userList	-	-
Wizard → select guests form → submit	Function makeCall	POST userscheckbox (array di userId)	CheckBoxUsers (servlet)
Logout		GET	Logout (servlet)

SERVER SIDE

- Model Objects (Beans)

  - UserBean
  - Conference

- Data Access Objects (DAO)

  - UserDAO

    - checkCredentials(username, password)
    - checkUsername(username)
    - registerUser(userBean)
    - getUsers(userId)

  - ConferenceDAO

    - findConferenceByUser(userId)
    - findConference2ByUser(userId)
    - findLastConferenceByUser(userId)
    - createConference(conferenceBean)

  - GuestsDAO

    - registerGuets(guets, conferenceId)

- Controllers (Servlets)

  - CheckLogin
  - RegisterServlet
  - getConferences
  - getConferences2
  - CreateConference
  - CheckBoxUsers
  - Logout

- Views (Templates)

  - Index (login + register)
  - Home

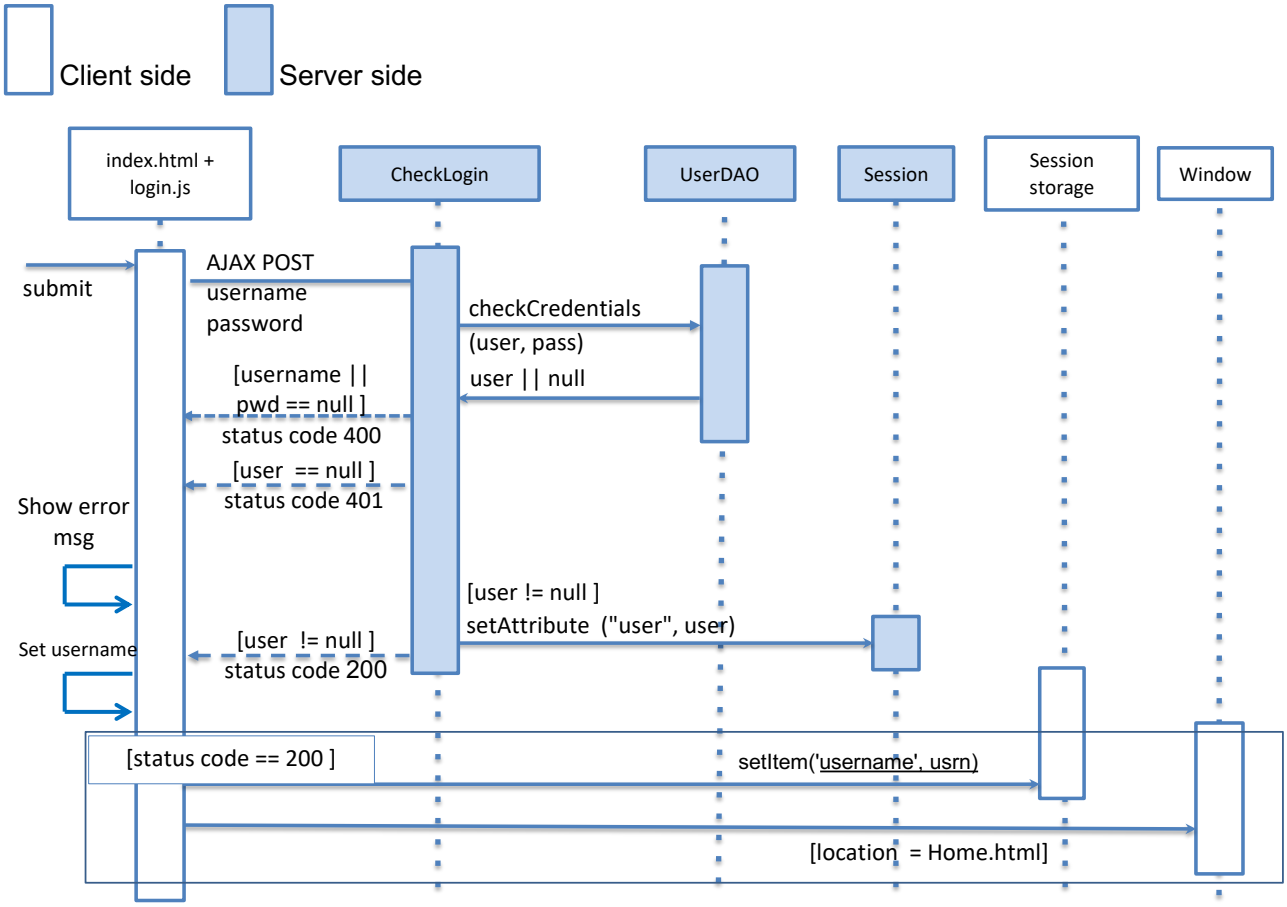
CLIENT SIDE

- Index
  - LoginWizard
    - registerEvents(): associa al componente le funzioni per gestirne gli eventi
    - makeCall(): con i dati del login
  - RegisterWizard
    - registerEvents(): associa al componente le funzioni per gestirne gli eventi
    - makeCall(): con i dati della registrazione

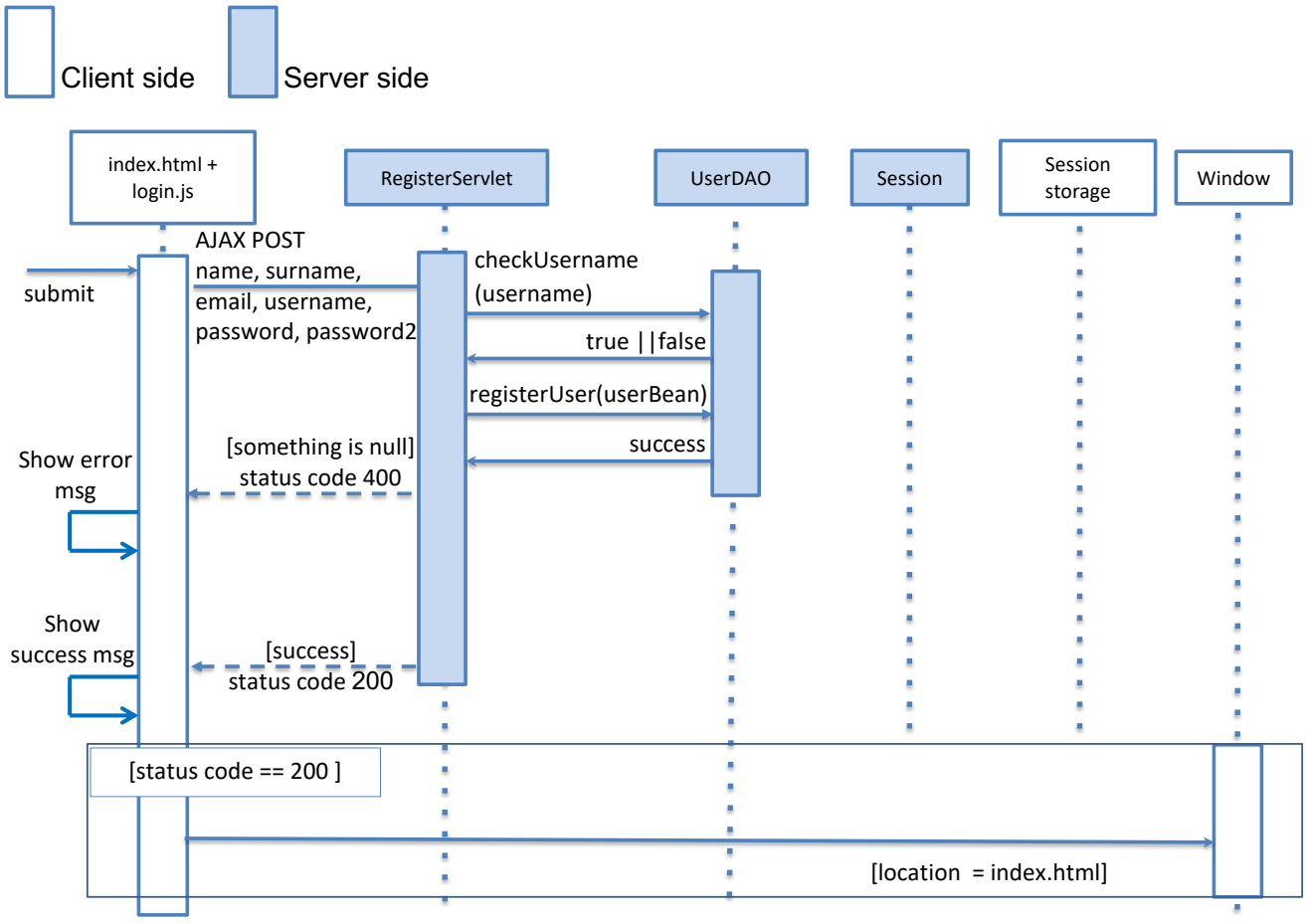
- Home
  - ConferenceList Lista conferenze create
    - Show(): richiede al server i dati dell'elenco conferenze e fa la update
    - Update(): aggiorna la lista delle conferenze e la rende visibile
    - Reset(): nasconde alla vista le conferenze
  - ConferenceList2 Lista conferenze alle quali si è stati invitati
    - Show(): richiede al server i dati dell'elenco conferenze e fa la update
    - Update(): aggiorna la lista delle conferenze e la rende visibile
    - Reset(): nasconde alla vista le conferenze
  - UserList Lista utenti disponibili
    - show(): chiama la update
    - update(): aggiorna la lista degli utenti la rende visibile
    - reset(): nasconde alla vista la lista di utenti
  - Wizard
    - registerEvents(): associa al componente le funzioni per gestirne gli eventi
    - makeCall(): con i dati della nuova conferenza

      salva i dati degli utenti dalla request
  - WizardUsers
    - registerEvents(): associa al componente le funzioni per gestirne gli eventi
    - makeCall(): con i dati degli utenti invitati

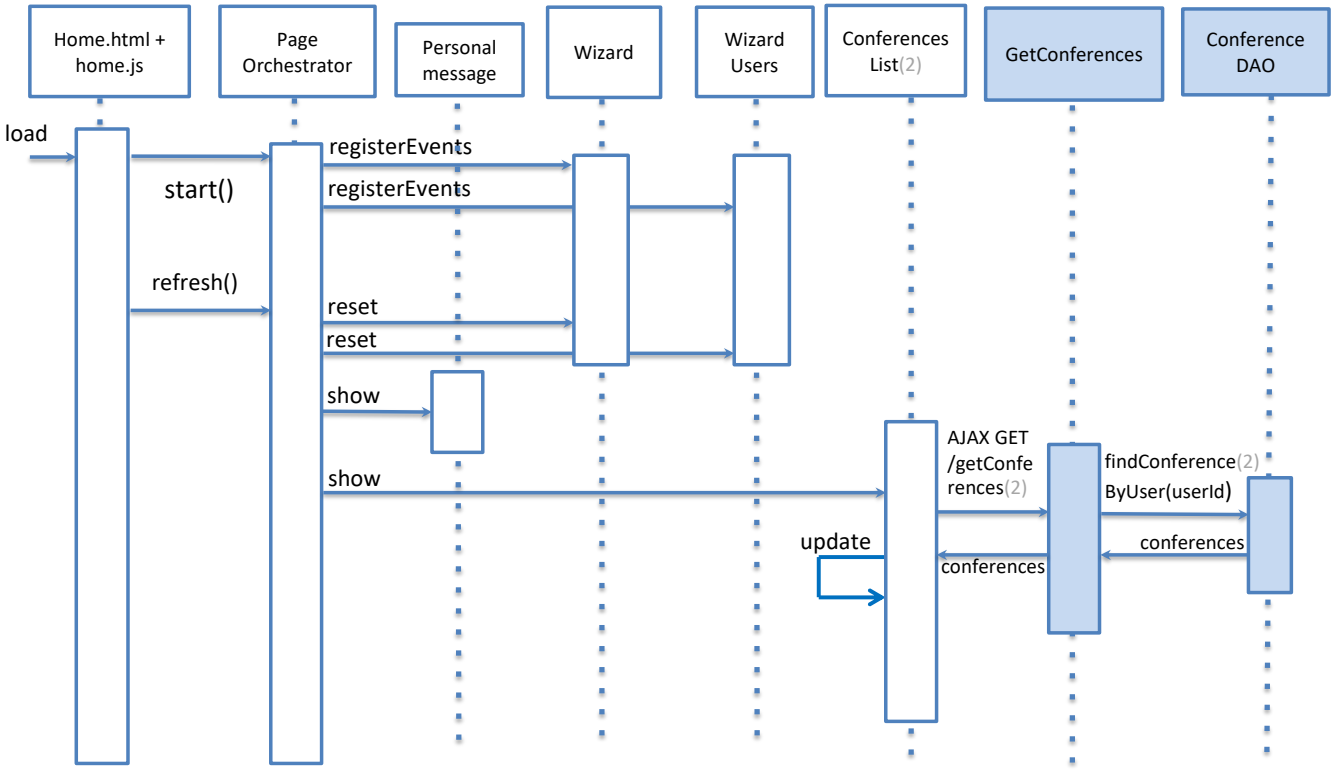
EVENT: LOGIN



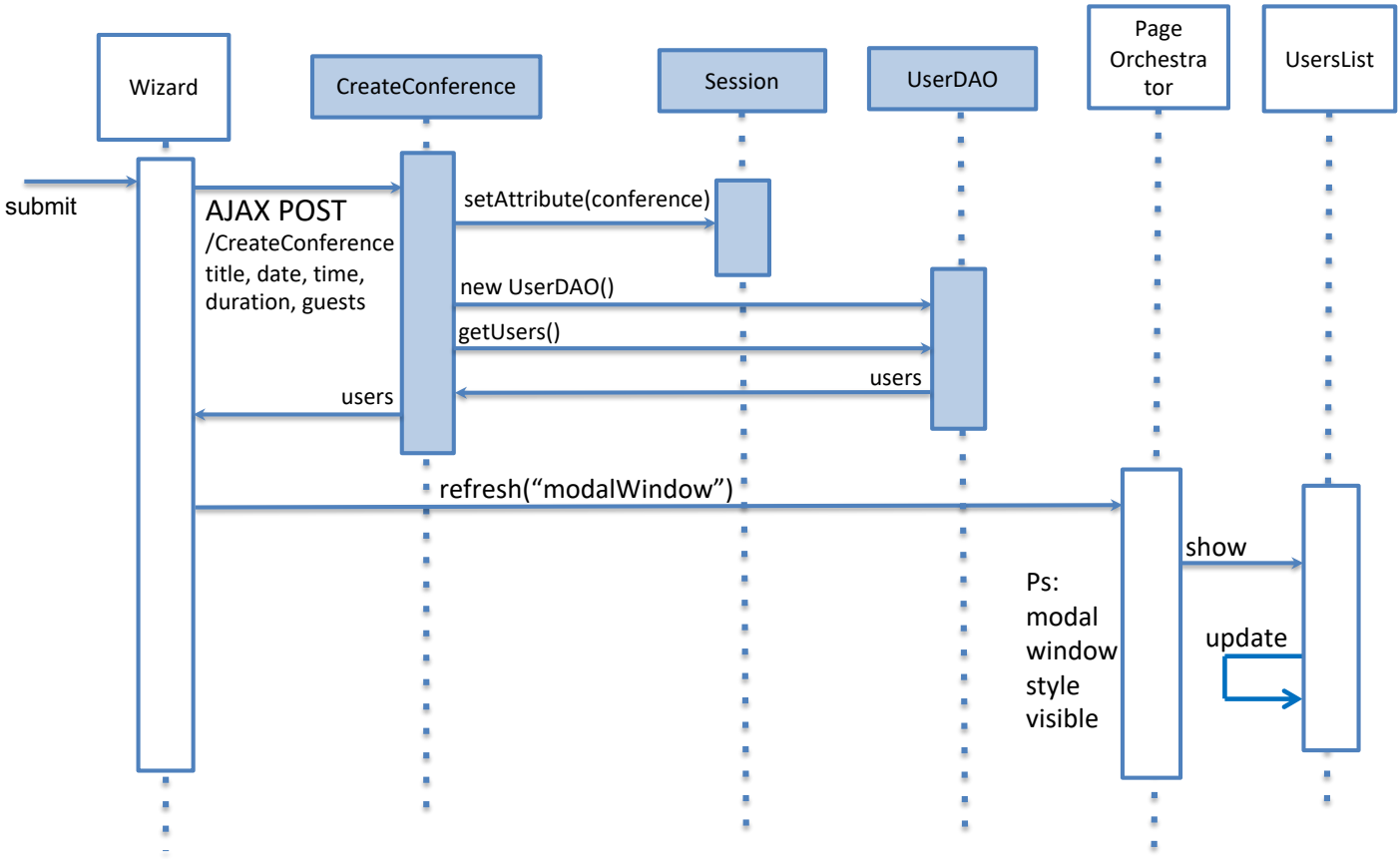
EVENT: REGISTER



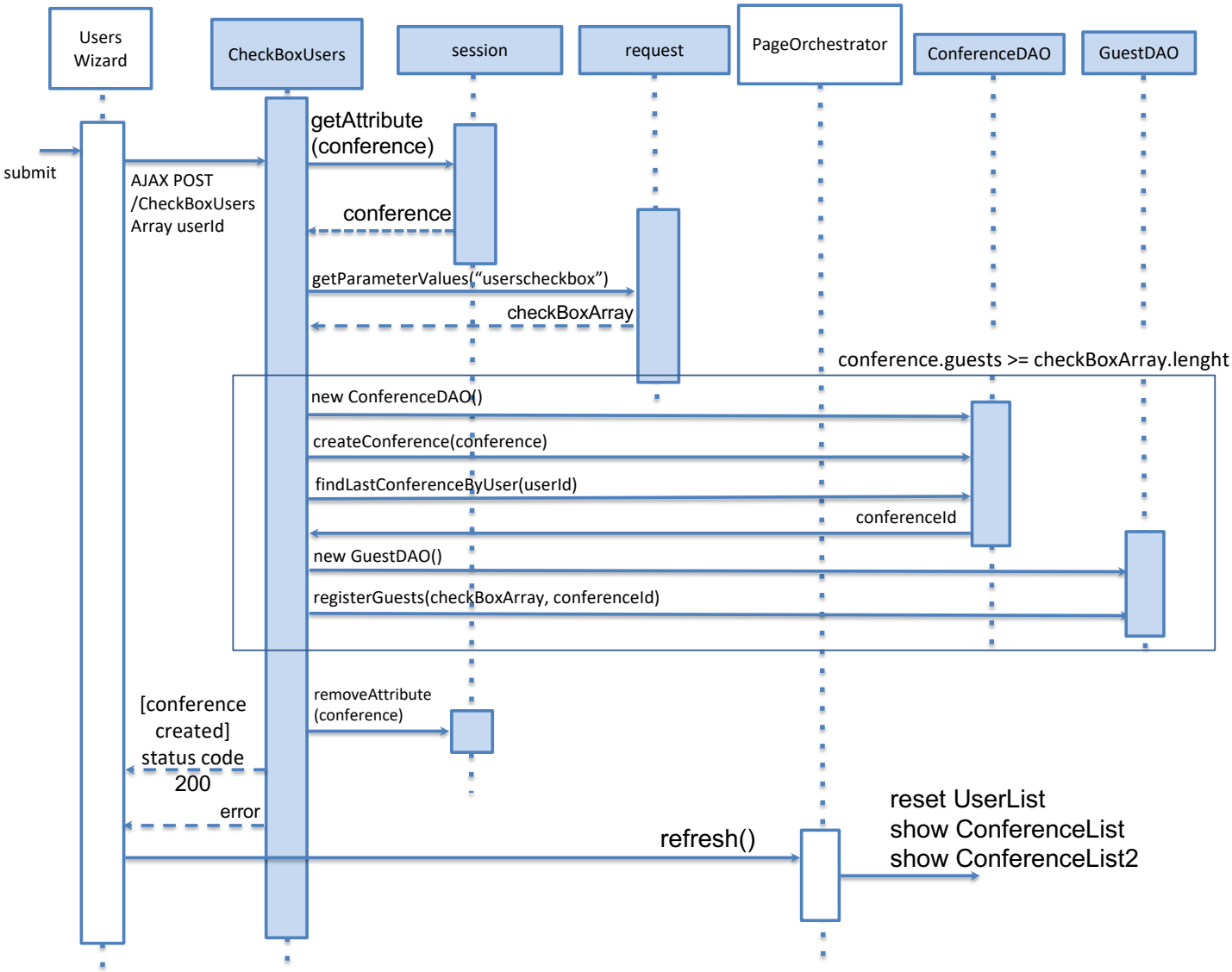
EVENT: GO TO HOME PAGE



EVENT: CREATE CONFERENCE



EVENT: SELECT GUESTS



EVENT: LOGOUT