Playlist Musicale

Alessandro Scibilia

Prof. Fraternali Piero

Anno Accademico 2022-2023

Specifica

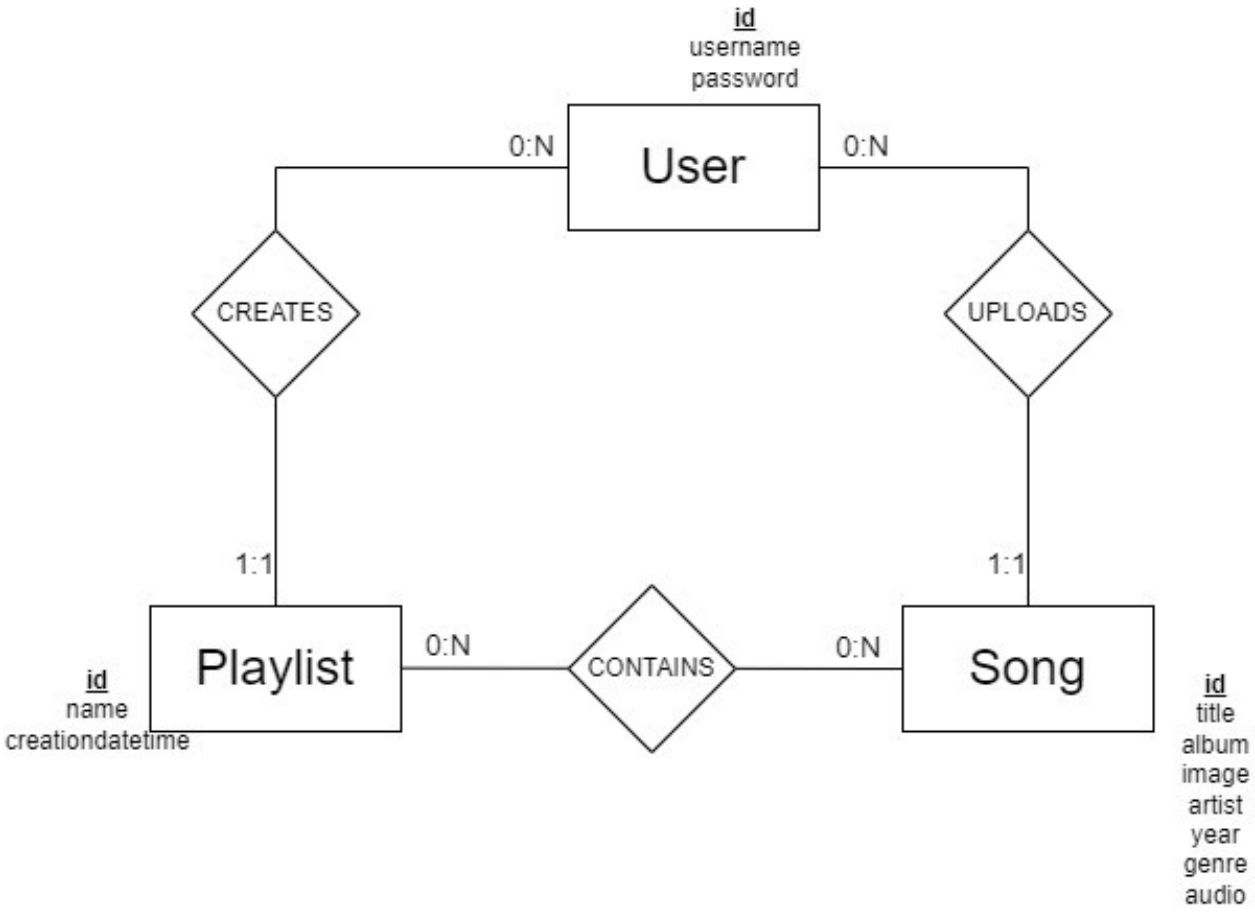
Un’applicazione web consente la gestione di una playlist di brani musicali. Playlist e brani sono personali di ogni utente e non condivisi. Ogni brano musicale è memorizzato nella base di dati mediante un titolo, l‘immagine e il titolo dell’album da cui il brano è tratto, il nome dell’interprete (singolo o gruppo) dell’album, l’anno di pubblicazione dell’album, il genere musicale (si supponga che i generi siano prefissati) e il file musicale. L’utente, previo login, può creare brani mediante il caricamento dei dati relativi e raggrupparli in playlist. Una playlist è un insieme di brani scelti tra quelli caricati dallo stesso utente ordinati per data decrescente dall’anno di pubblicazione dell’album. Lo stesso brano può essere inserito in più playlist. Una playlist ha un titolo e una data di creazione ed è associata al suo creatore. A seguito del login, l’utente accede all’HOME PAGE che presenta l’elenco delle proprie playlist, ordinate per data di creazione decrescente, una form per caricare un brano con tutti i dati relativi e una form per creare una nuova playlist. La creazione di una nuova playlist richiede di selezionare uno o più brani da includere. Quando l’utente clicca su una playlist nell’HOME PAGE, appare la pagina PLAYLIST PAGE che contiene inizialmente una tabella di una riga e cinque colonne. Ogni cella contiene il titolo di un brano e l’immagine dell’album da cui proviene. I brani sono ordinati da sinistra a destra per data decrescente dell’album di pubblicazione. Se la playlist contiene più di cinque brani, sono disponibili comandi per vedere il precedente e successivo gruppo di brani. Se la pagina PLAYLIST mostra il primo gruppo e ne esistono altri successivi nell’ordinamento, compare a destra della riga il bottone SUCCESSIVI, che permette di vedere il gruppo successivo. Se la pagina PLAYLIST mostra l’ultimo gruppo e ne esistono altri precedenti nell’ordinamento, compare a sinistra della riga il bottone PRECEDENTI, che permette di vedere i cinque brani precedenti. Se la pagina PLAYLIST mostra un blocco e esistono sia precedenti sia successivi, compare a destra della riga il bottone SUCCESSIVI e a sinistra il bottone PRECEDENTI. La pagina PLAYLIST contiene anche una form che consente di selezionare e aggiungere un brano alla playlist corrente, se non già presente nella playlist. A seguito dell’aggiunta di un brano alla playlist corrente, l’applicazione visualizza nuovamente la pagina a partire dal primo blocco della playlist. Quando l’utente seleziona il titolo di un brano, la pagina PLAYER mostra tutti i dati del brano scelto e il player audio per la riproduzione del brano.

Analisi dei dati

ENTITIES, ATTRIBUTES, RELATIONSHIPS

Un'applicazione web consente la gestione di una playlist di brani musicali. Playlist e brani sono personali di ogni utente e non condivisi. Ogni brano musicale è memorizzato nella base di dati mediante un titolo, l'immagine e il titolo dell'album da cui il brano è tratto, il nome dell'interprete (singolo o gruppo) dell'album, l'anno di pubblicazione dell'album, il genere musicale (si supponga che i generi siano prefissati) e il file musicale. L'utente, previo login, può creare brani mediante il caricamento dei dati relativi e raggrupparli in playlist. Una playlist è un insieme di brani scelti tra quelli caricati dallo stesso utente ordinati per data decrescente dall'anno di pubblicazione dell'album. Lo stesso brano può essere inserito in più playlist. Una playlist ha un titolo e una data di creazione ed è associata al suo creatore. A seguito del login, l'utente accede all'HOME PAGE che presenta l'elenco delle proprie playlist, ordinate per data di creazione decrescente, una form per caricare un brano con tutti i dati relativi e una form per creare una nuova playlist. La creazione di una nuova playlist richiede di selezionare uno o più brani da includere. Quando l'utente clicca su una playlist nell'HOME PAGE, appare la pagina PLAYLIST PAGE che contiene inizialmente una tabella di una riga e cinque colonne. Ogni cella contiene il titolo di un brano e l'immagine dell'album da cui proviene. I brani sono ordinati da sinistra a destra per data decrescente dell'album di pubblicazione. Se la playlist contiene più di cinque brani, sono disponibili comandi per vedere il precedente e successivo gruppo di brani. Se la pagina PLAYLIST mostra il primo gruppo e ne esistono altri successivi nell'ordinamento, compare a destra della riga il bottone SUCCESSIVI, che permette di vedere il gruppo successivo. Se la pagina PLAYLIST mostra l'ultimo gruppo e ne esistono altri precedenti nell'ordinamento, compare a sinistra della riga il bottone PRECEDENTI, che permette di vedere i cinque brani precedenti. Se la pagina PLAYLIST mostra un blocco e esistono sia precedenti sia successivi, compare a destra della riga il bottone SUCCESSIVI e a sinistra il bottone PRECEDENTI. La pagina PLAYLIST contiene anche una form che consente di selezionare e aggiungere un brano alla playlist corrente, se non già presente nella playlist. A seguito dell'aggiunta di un brano alla playlist corrente, l'applicazione visualizza nuovamente la pagina a partire dal primo blocco della playlist. Quando l'utente seleziona il titolo di un brano, la pagina PLAYER mostra tutti i dati del brano scelto e il player audio per la riproduzione del brano.

Diagramma E/R



Schema Database

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

Schema Database

```
CREATE TABLE playlist
(
  id int NOT NULL AUTO_INCREMENT,
  name varchar(45) NOT NULL,
  owner int NOT NULL,
  creationdatetime timestamp NOT NULL,
  PRIMARY KEY (id),
  KEY owneruser_idx (owner),
  CONSTRAINT owneruser FOREIGN KEY (owner)
     REFERENCES user (id)
);
```

Schema Database

```
CREATE TABLE song (
  id int NOT NULL AUTO_INCREMENT,
  title varchar(45) NOT NULL,
  album varchar(45) NOT NULL,
  image longblob NOT NULL,
  artist varchar(45) NOT NULL,
  year int NOT NULL,
  genre varchar(45) NOT NULL,
  audio longblob NOT NULL,
  owner int NOT NULL,
  PRIMARY KEY (id),
  KEY owneruser_idx (owner),
  CONSTRAINT songowneruser FOREIGN KEY (owner) REFERENCES user (id),
  CONSTRAINT song_chk_1 CHECK ((year between 0 and 9999)),
  CONSTRAINT song_chk_2 CHECK ((genre in ('Rock','Rap','Hip Hop', ,'Jazz',
      'Country','Blues','Metal','Soul','Punk','Reggae','Funk','Electronic','Pop')))
);
```

Schema Database

```
CREATE TABLE playlist_song (
  song_id int NOT NULL,
  playlist_id int NOT NULL,
  owner_id int NOT NULL,
  PRIMARY KEY (song_id,playlist_id),
  KEY playlist_id (playlist_id),
  KEY owner_id (owner_id),
  CONSTRAINT playlist_song_1 FOREIGN KEY (song_id)
     REFERENCES      song (id),
  CONSTRAINT playlist_song_2 FOREIGN KEY (playlist_id)
     REFERENCES playlist (id),
  CONSTRAINT playlist_song_3 FOREIGN KEY (owner_id)
     REFERENCES      user (id)
);
```

Analisi dei requisiti - 1

PAGES (VIEWS), VIEW COMPONENTS

Un'applicazione web consente la gestione di una playlist di brani musicali. Playlist e brani sono personali di ogni utente e non condivisi. Ogni brano musicale è memorizzato nella base di dati mediante un titolo, l'immagine e il titolo dell'album da cui il brano è tratto, il nome dell'interprete (singolo o gruppo) dell'album, l'anno di pubblicazione dell'album, il genere musicale (si supponga che i generi siano prefissati) e il file musicale. L'utente, previo login, può creare brani mediante il caricamento dei dati relativi e raggrupparli in playlist. Una playlist è un insieme di brani scelti tra quelli caricati dallo stesso utente ordinati per data decrescente dall'anno di pubblicazione dell'album. Lo stesso brano può essere inserito in più playlist. Una playlist ha un titolo e una data di creazione ed è associata al suo creatore. A seguito del login, l'utente accede all'HOME PAGE che presenta l'elenco delle proprie playlist, ordinate per data di creazione decrescente, una form per caricare un brano con tutti i dati relativi e una form per creare una nuova playlist. La creazione di una nuova playlist richiede di selezionare uno o più brani da includere. Quando l'utente clicca su una playlist nell'HOME PAGE, appare la pagina PLAYLIST PAGE che contiene inizialmente una tabella di una riga e cinque colonne. Ogni cella contiene il titolo di un brano e l'immagine dell'album da cui proviene. I brani sono ordinati da sinistra a destra per data decrescente dell'album di pubblicazione. Se la playlist contiene più di cinque brani, sono disponibili comandi per vedere il precedente e successivo gruppo di brani. Se la pagina PLAYLIST mostra il primo gruppo e ne esistono altri successivi nell'ordinamento, compare a destra della riga il bottone SUCCESSIVI, che permette di vedere il gruppo successivo. Se la pagina PLAYLIST mostra l'ultimo gruppo e ne esistono altri precedenti nell'ordinamento, compare a sinistra della riga il bottone PRECEDENTI, che permette di vedere i cinque brani precedenti. Se la pagina PLAYLIST mostra un blocco e esistono sia precedenti sia successivi, compare a destra della riga il bottone SUCCESSIVI e a sinistra il bottone PRECEDENTI. La pagina PLAYLIST contiene anche una form che consente di selezionare e aggiungere un brano alla playlist corrente, se non già presente nella playlist. A seguito dell'aggiunta di un brano alla playlist corrente, l'applicazione visualizza nuovamente la pagina a partire dal primo blocco della playlist. Quando l'utente seleziona il titolo di un brano, la pagina PLAYER mostra tutti i dati del brano scelto e il player audio per la riproduzione del brano.

Analisi dei requisiti - 2

EVENTS, ACTIONS

Un'applicazione web consente la gestione di una playlist di brani musicali. Playlist e brani sono personali di ogni utente e non condivisi. Ogni brano musicale è memorizzato nella base di dati mediante un titolo, l'immagine e il titolo dell'album da cui il brano è tratto, il nome dell'interprete (singolo o gruppo) dell'album, l'anno di pubblicazione dell'album, il genere musicale (si supponga che i generi siano prefissati) e il file musicale. L'utente, previo login, può creare brani mediante il caricamento dei dati relativi e raggrupparli in playlist. Una playlist è un insieme di brani scelti tra quelli caricati dallo stesso utente ordinati per data decrescente dall'anno di pubblicazione dell'album. Lo stesso brano può essere inserito in più playlist. Una playlist ha un titolo e una data di creazione ed è associata al suo creatore. A seguito del login, l'utente accede all'HOME PAGE che presenta l'elenco delle proprie playlist, ordinate per data di creazione decrescente, una form per caricare un brano con tutti i dati relativi e una form per creare una nuova playlist. La creazione di una nuova playlist richiede di selezionare uno o più brani da includere. Quando l'utente clicca su una playlist nell'HOME PAGE, appare la pagina PLAYLIST PAGE che contiene inizialmente una tabella di una riga e cinque colonne. Ogni cella contiene il titolo di un brano e l'immagine dell'album da cui proviene. I brani sono ordinati da sinistra a destra per data decrescente dell'album di pubblicazione. Se la playlist contiene più di cinque brani, sono disponibili comandi per vedere il precedente e successivo gruppo di brani. Se la pagina PLAYLIST mostra il primo gruppo e ne esistono altri successivi nell'ordinamento, compare a destra della riga il bottone SUCCESSIVI, che permette di vedere il gruppo successivo. Se la pagina PLAYLIST mostra l'ultimo gruppo e ne esistono altri precedenti nell'ordinamento, compare a sinistra della riga il bottone PRECEDENTI, che permette di vedere i cinque brani precedenti. Se la pagina PLAYLIST mostra un blocco e esistono sia precedenti sia successivi, compare a destra della riga il bottone SUCCESSIVI e a sinistra il bottone PRECEDENTI. La pagina PLAYLIST contiene anche una form che consente di selezionare e aggiungere un brano alla playlist corrente, se non già presente nella playlist. A seguito dell'aggiunta di un brano alla playlist corrente, l'applicazione visualizza nuovamente la pagina a partire dal primo blocco della playlist. Quando l'utente seleziona il titolo di un brano, la pagina PLAYER mostra tutti i dati del brano scelto e il player audio per la riproduzione del brano.

Versione HTML puro

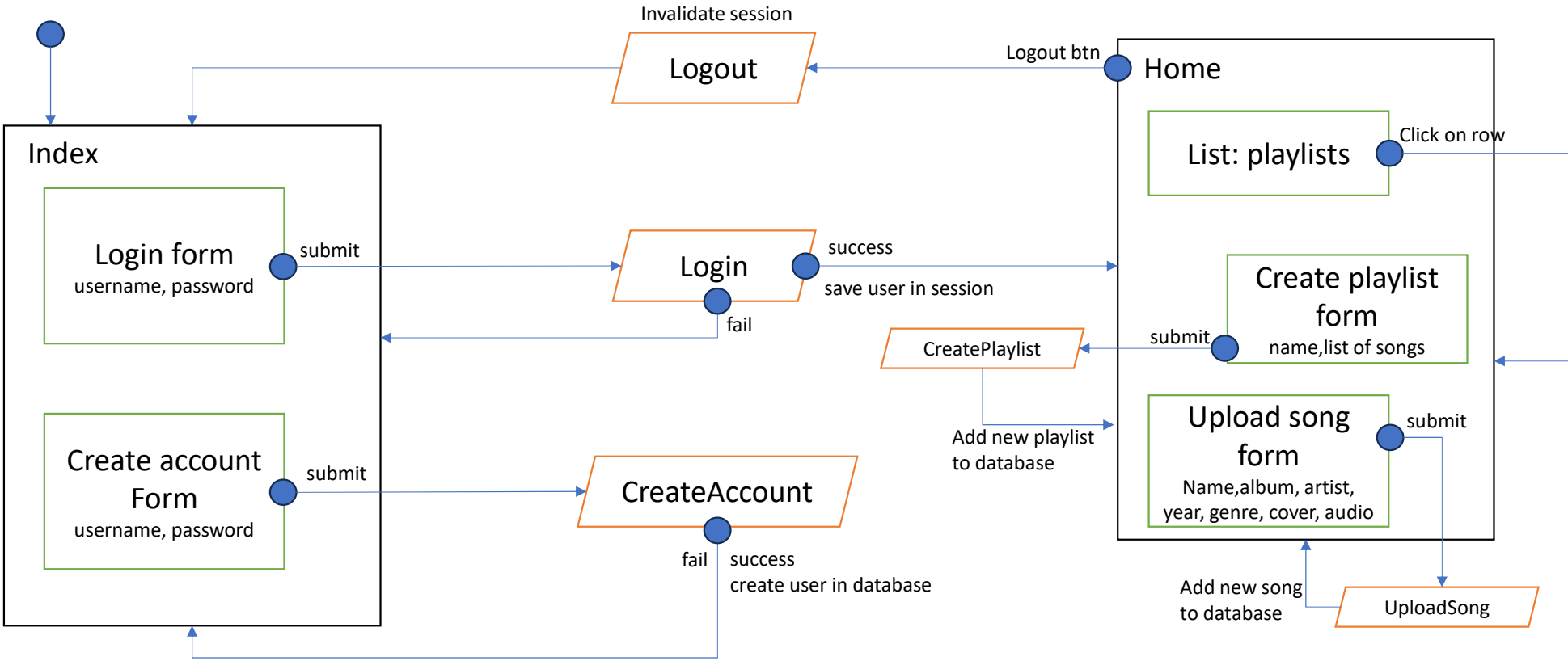
Componenti

- **Model objects** (JavaBeans)
  - Playlist
  - Song
  - User
- **Data Access Objects**
  - **UserDAO**
    - checkCredentials(usrn, pwd)
    - createAccount(usrn, pwd)
    - deleteAccount(userId)
    - findPlaylists(userId)
    - getAllUserSongs(userId, getAlbumCovers)
  - SongDAO
    - uploadSong(songName, albumName, artistName, year, genre, albumCover, audioFile, userId)
    - addSongToPlaylist(songId, playlistId, userId)
    - addSongsToPlaylist(songList, playlistId, userId)
    - getSong(songId, userId)
    - deleteSong(songId, userId)
    - checkSongOwner(songId, userId)
  - PlaylistDAO
    - createPlaylist(name, songList, userId)
    - getSongsFiltered(playlistId, userId, offset)
    - removeSongFromPlaylist(songId, playlistId, userId)
    - getPlaylistTitle(playlistId, userId)
    - deletePlaylist(playlistId, userId)
    - countSongs(playlistId, userId)
    - getSongsNotInPlaylist(userId, playlistId)

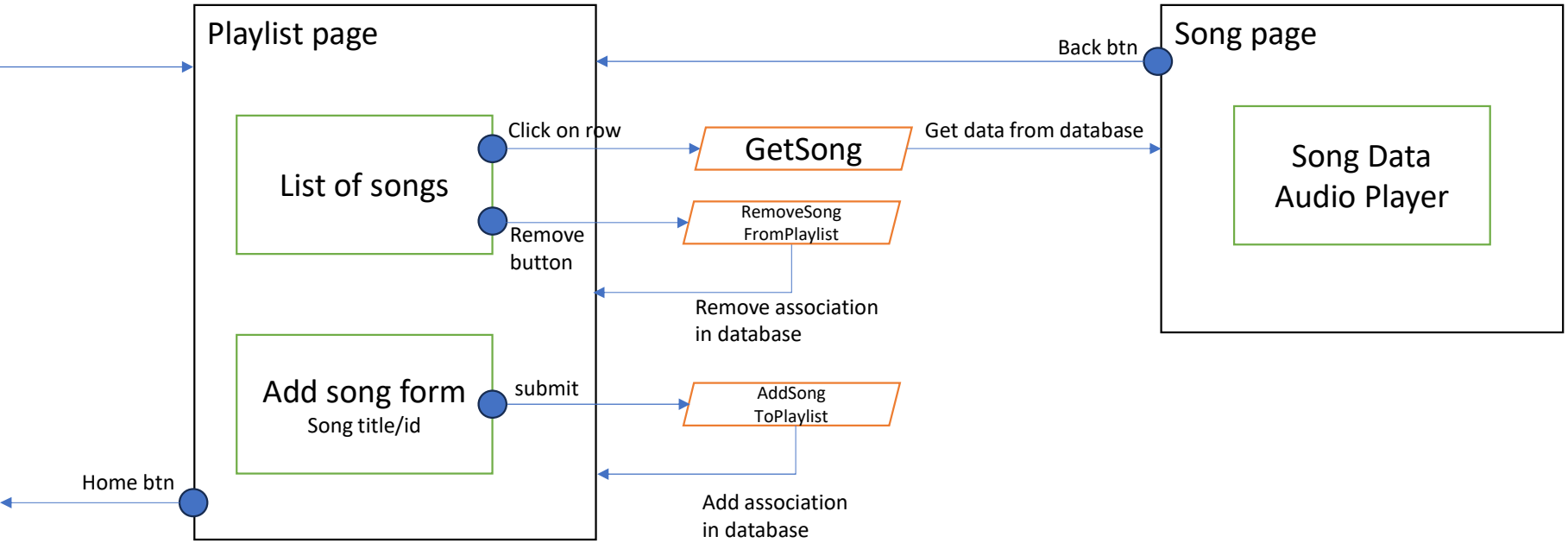
Componenti

- **Controllers** (servlets)
  - Authentication
    - Login
    - Logout
    - CreateAccount
    - DeleteAccount
  - Get content
    - AllSongs
    - GetPlaylist
    - GetSong
    - GoHome
  - Create content
    - AddSongsToPlaylist
    - CreatePlaylist
    - UploadSong
  - Delete content
    - DeletePlaylist
    - DeleteSong
    - RemoveSongFromPlaylist
- **Views** (templates)
  - Index *(login and signup)*
  - *Home*
  - *AllSongs*
  - *PlaylistPage*
  - *SongPage*
- **Utilities**
  - ConnectionHandler

Application design (login/create account)

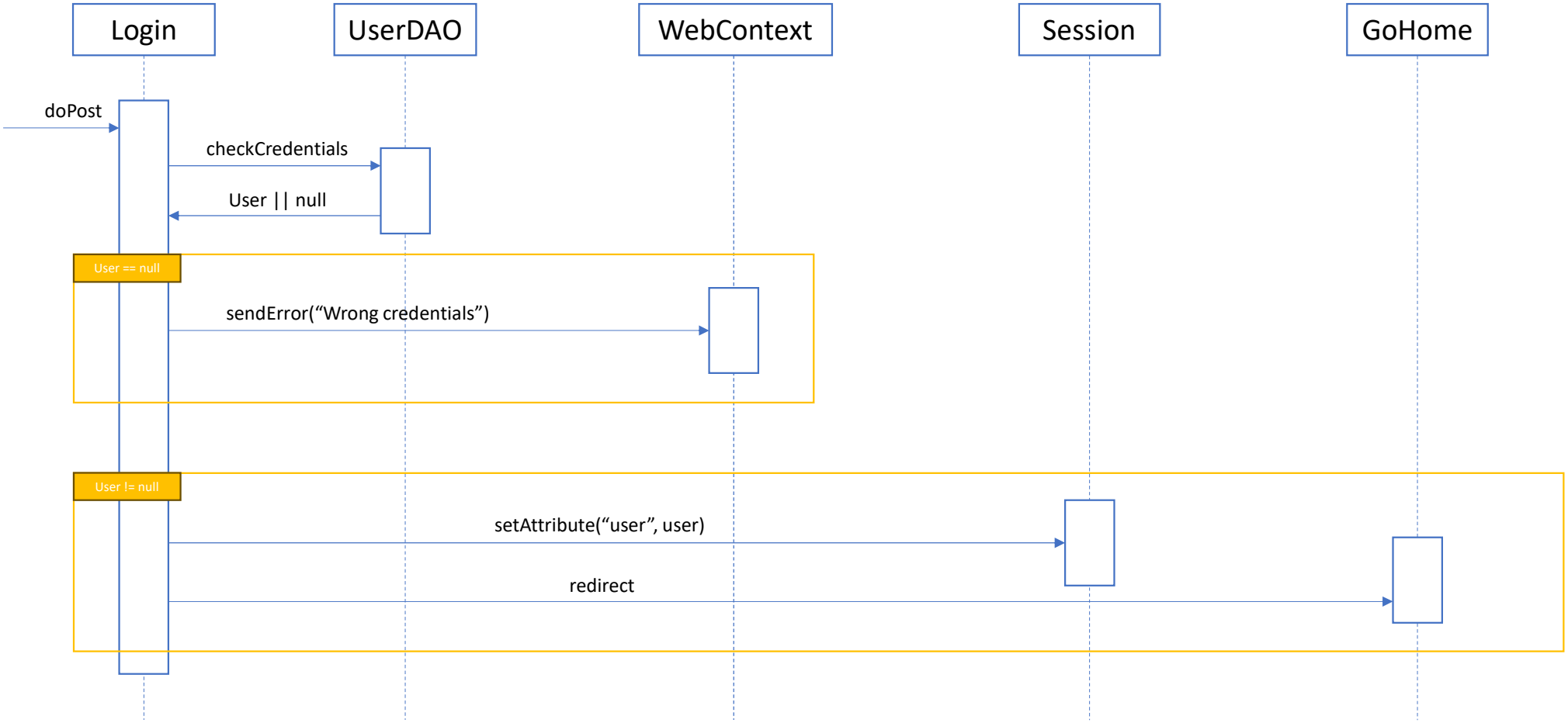


Application design: playlist/song



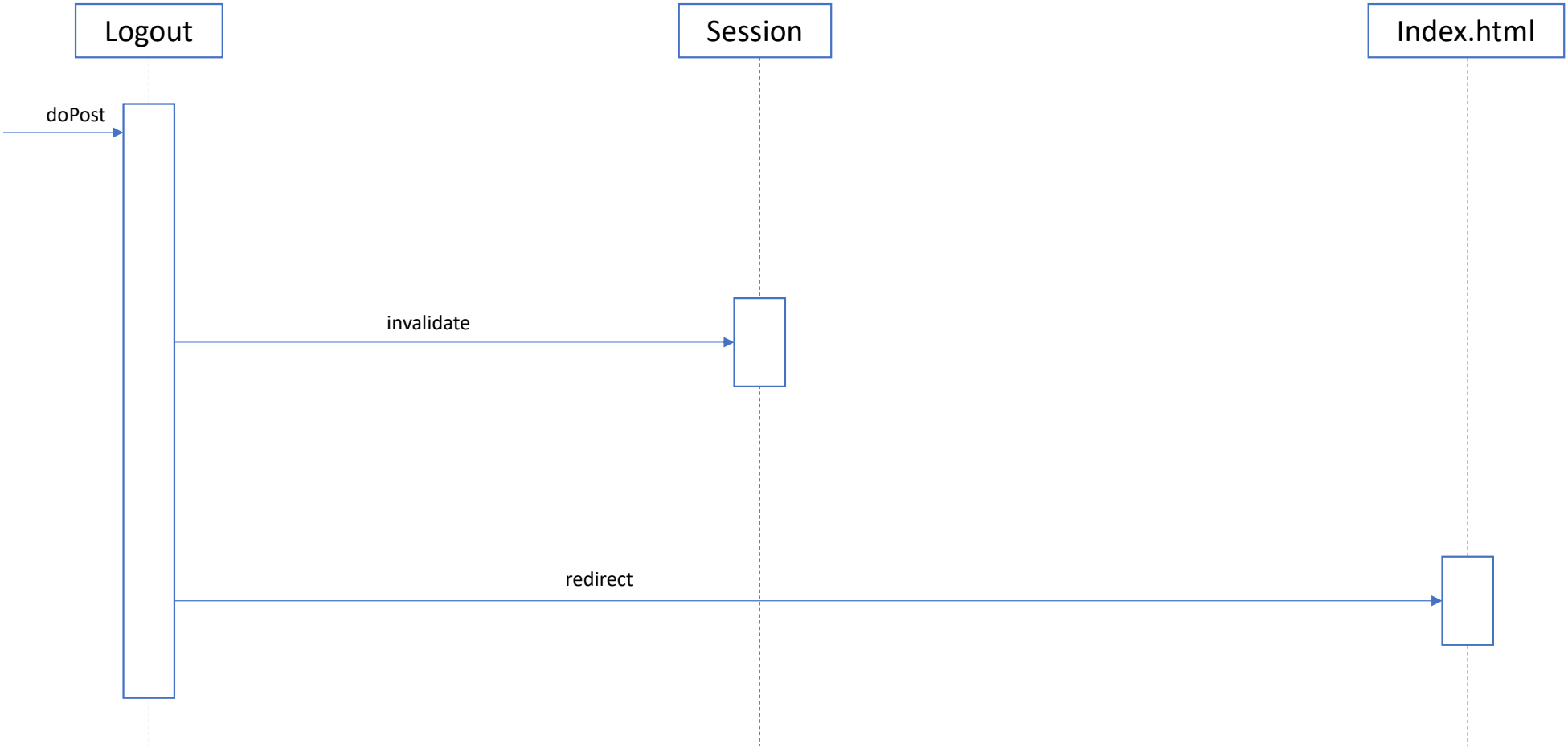
Event: login

POST request from index.html
Parameters: username, password

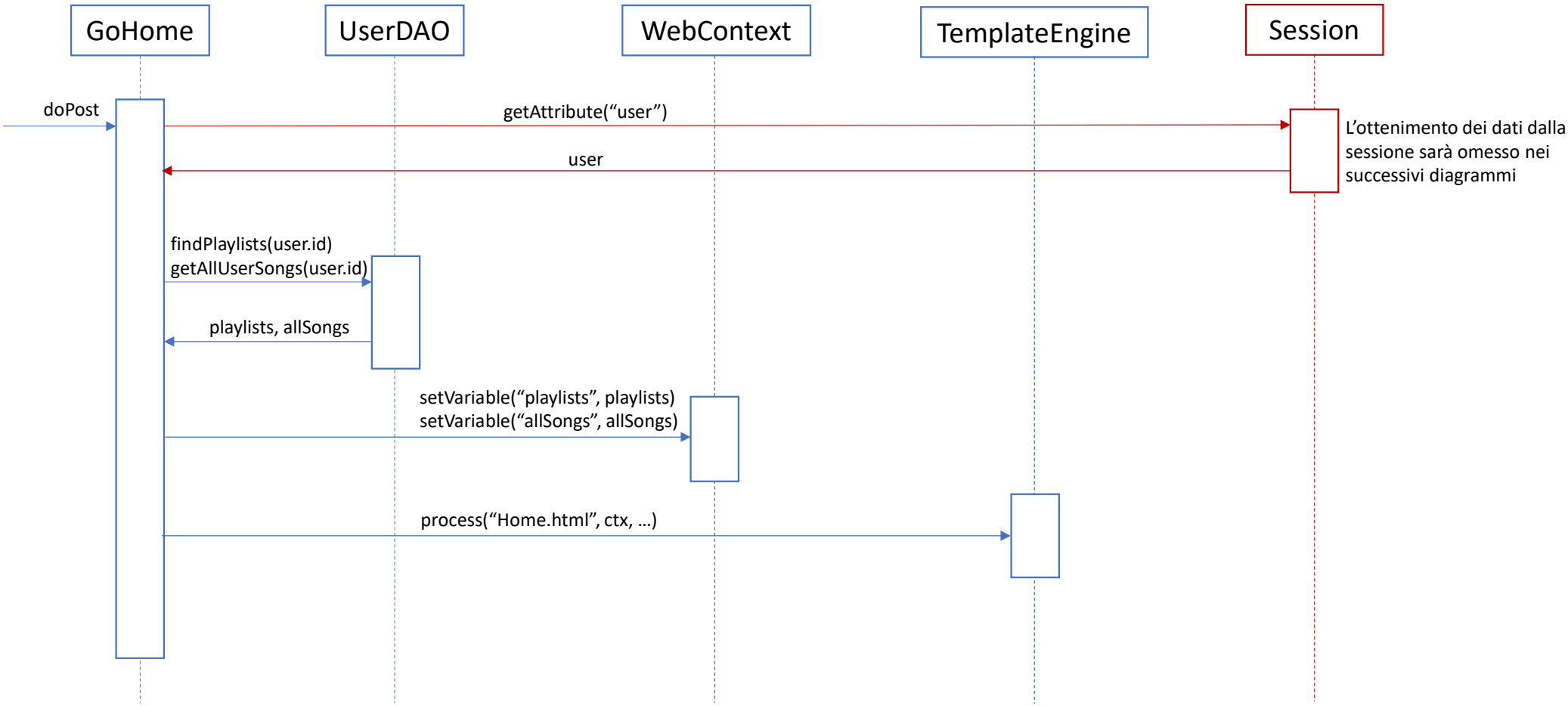


Event: logout

POST request from Home.html

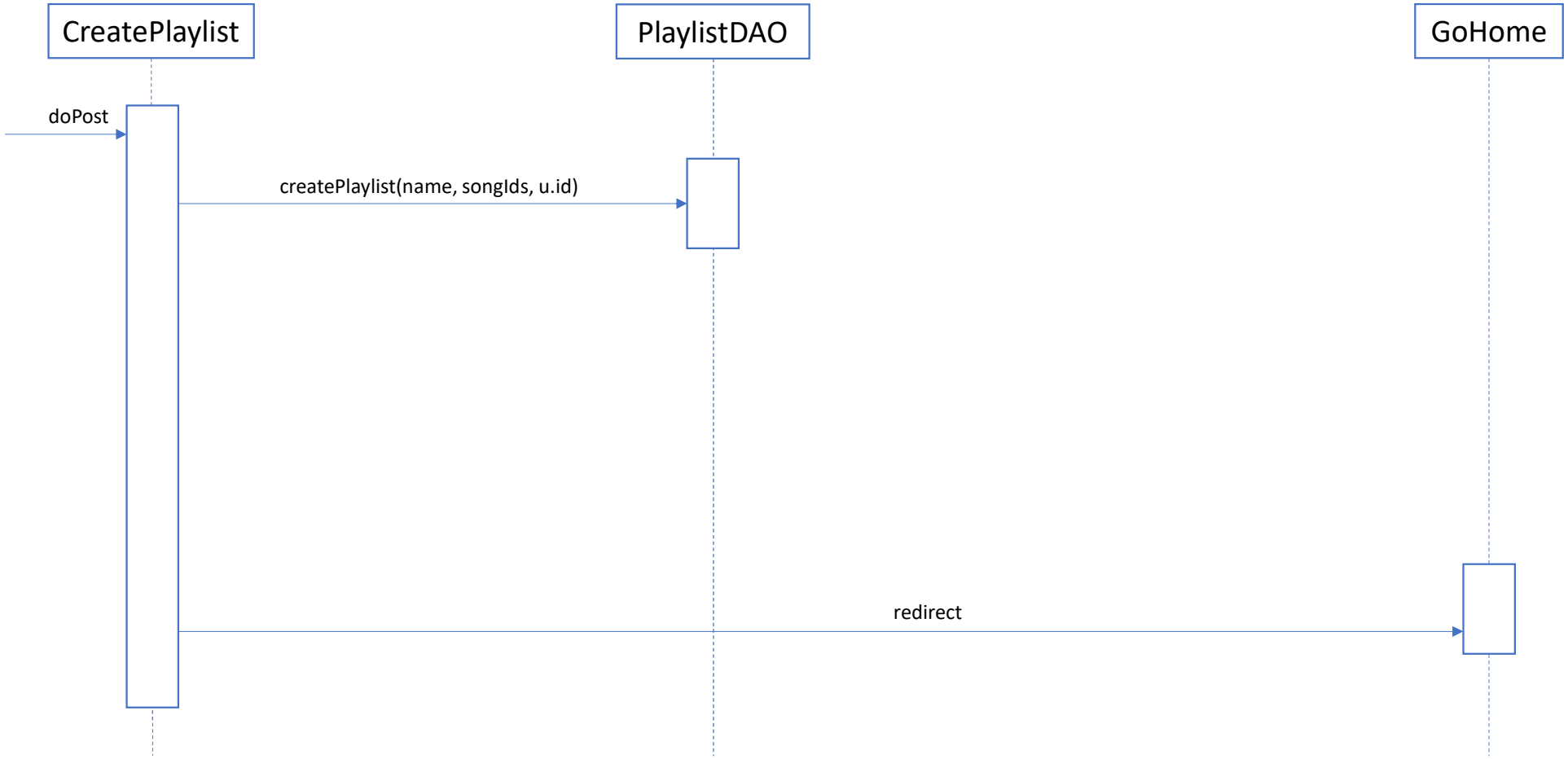


View: GoHome GET request



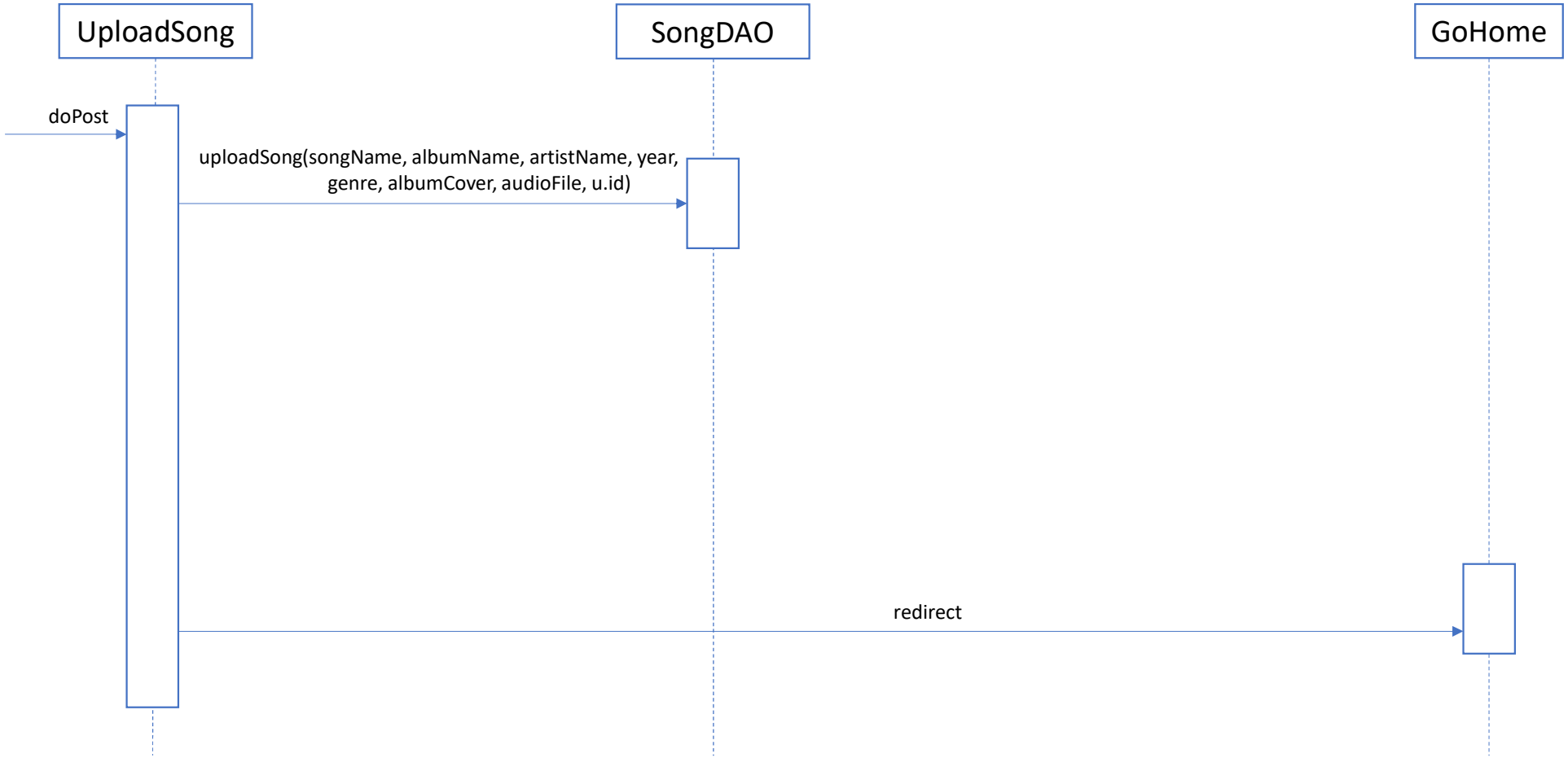
Event: CreatePlaylist

POST request from Home.html
Parameters: name, songIds



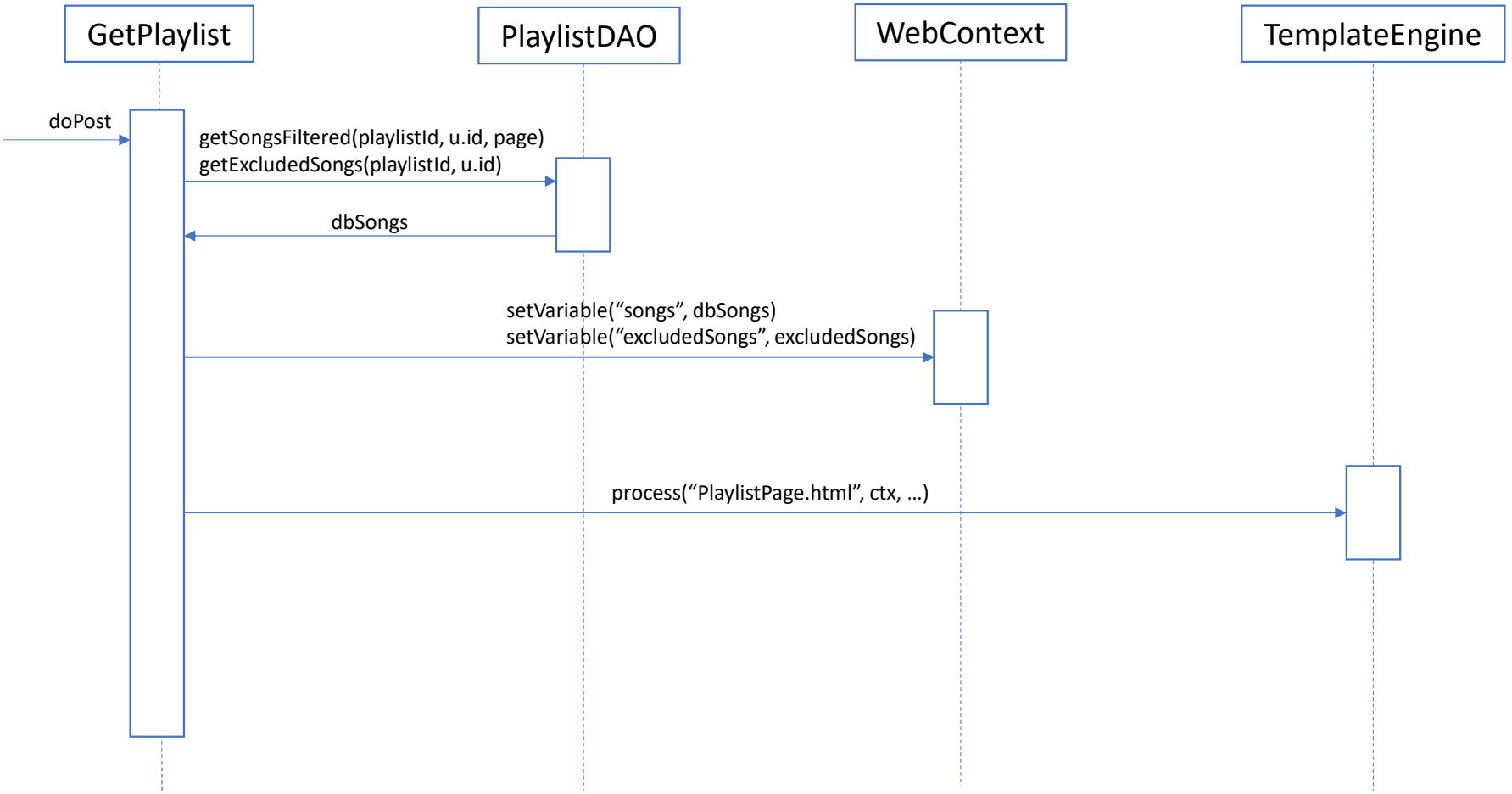
Event: UploadSong

POST request from Home.html
Parameters: songName, albumName, artistName, year, genre, albumCover, audioFile



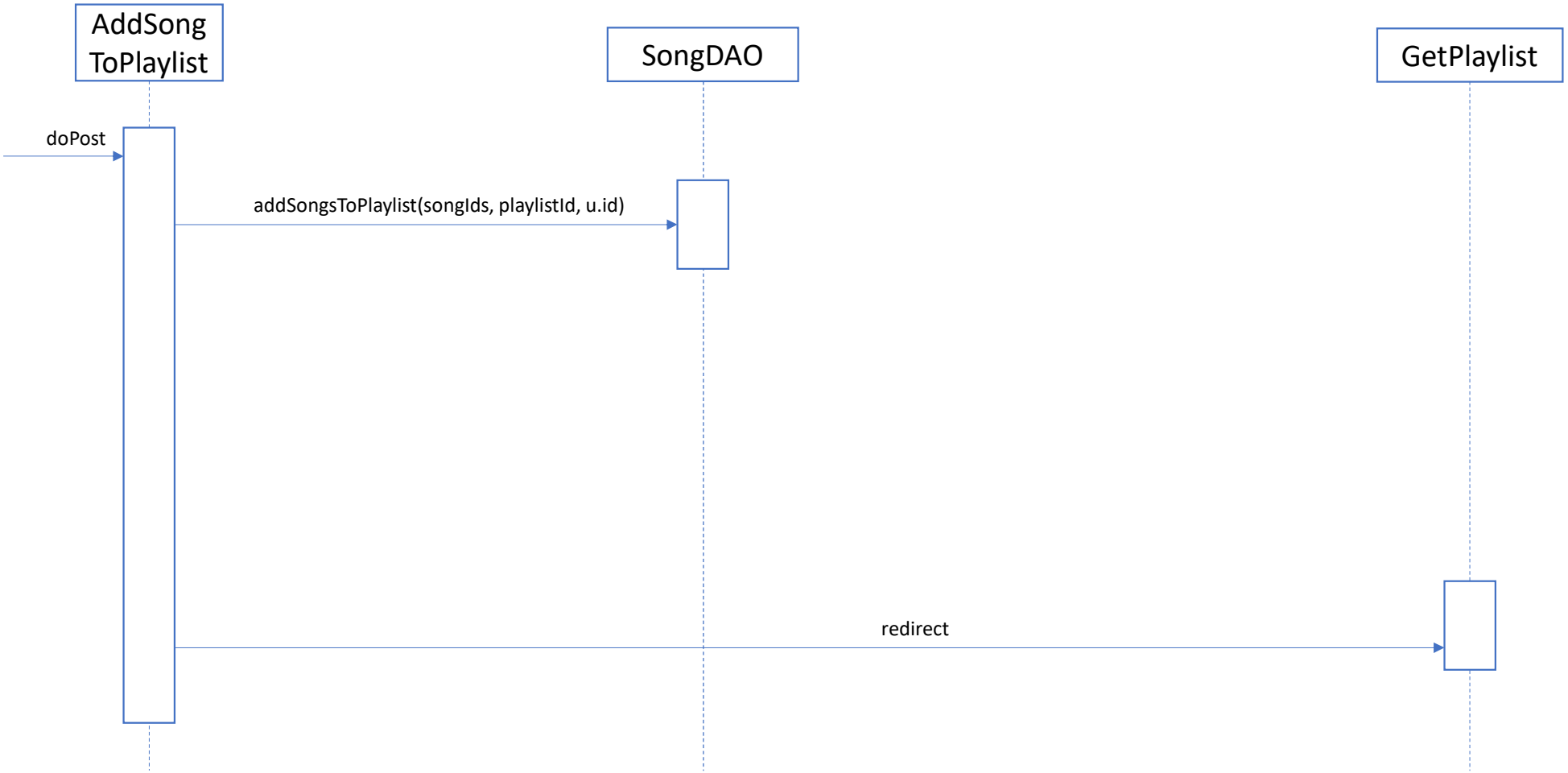
View: getPlaylist

GET request from Home.html
Parameters: playlistId



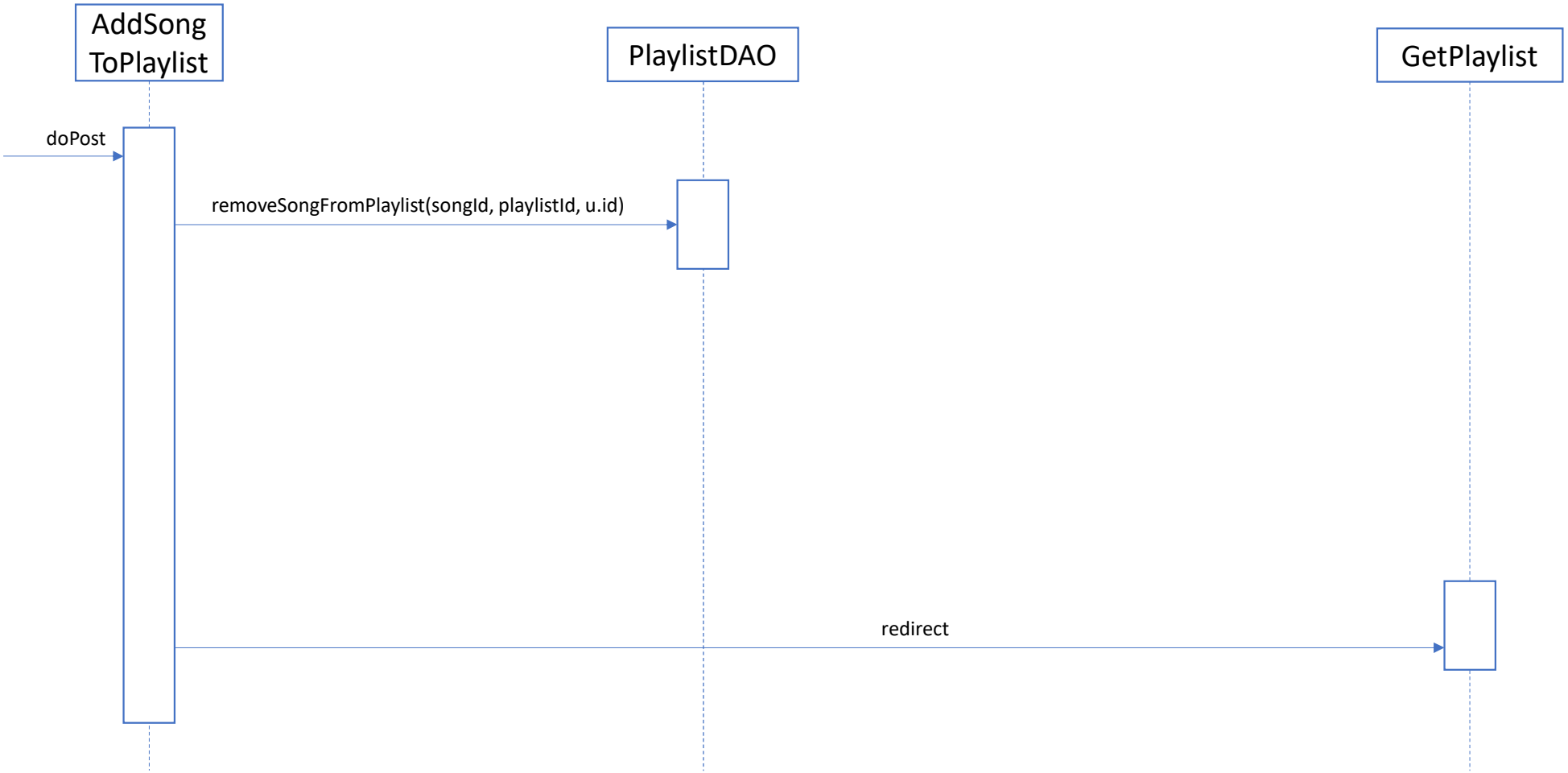
Event: AddSongsToPlaylist

POST request from PlaylistPage.html
Parameters: songIds, playlistId



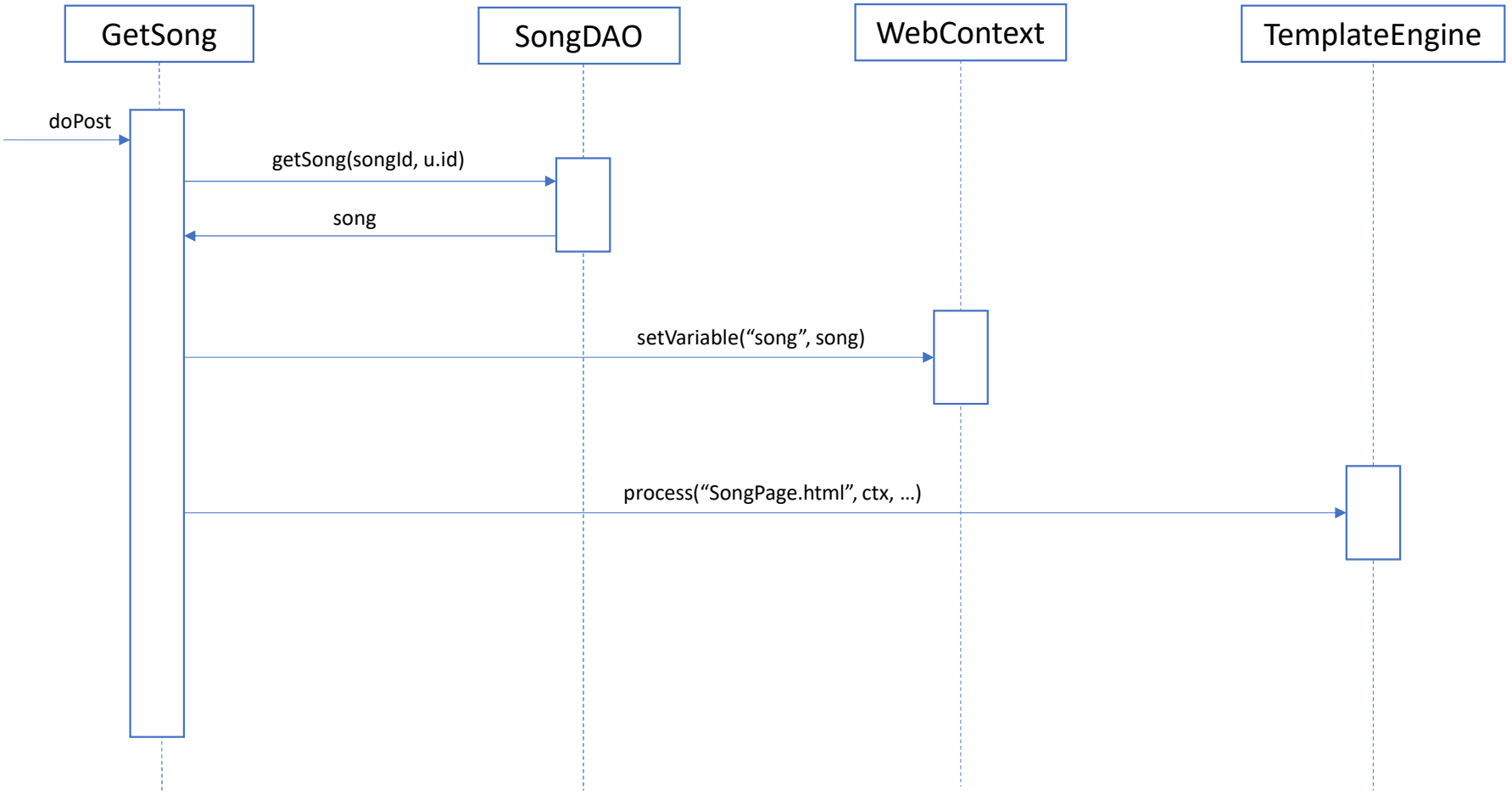
Event: RemoveSongFromPlaylist

POST request from PlaylistPage.html
Parameters: songId, playlistId



View: getSong

GET request from PlaylistPage.html
Parameters: songId



Versione JavaScript

Specifiche aggiuntive

- 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.
- L'evento di visualizzazione del blocco precedente/successivo è gestito a lato client senza generare una richiesta al server.
- L'applicazione deve consentire all'utente di riordinare le playlist con un criterio diverso da quello di default (data decrescente). Dalla HOME con un link associato a ogni playlist page si accede a una pagina RIORDINO, che mostra la lista completa dei brani della playlist e permette all'utente di trascinare il titolo di un brano nell'elenco e di collocarlo in una posizione diversa per realizzare l'ordinamento che desidera, senza invocare il server. Quando l'utente ha raggiunto l'ordinamento desiderato, usa un bottone "salva ordinamento", per memorizzare la sequenza sul server. Ai successivi accessi, l'ordinamento personalizzato è usato al posto di quello di default.

Cambiamenti alla base di dati:

```
CREATE TABLE playlist_song (
  song_id int NOT NULL,
  playlist_id int NOT NULL,
  owner_id int NOT NULL,
  position int DEFAULT NULL,
  PRIMARY KEY (song_id,playlist_id),
  KEY playlist_id (playlist_id),
  KEY owner_id (owner_id),
  CONSTRAINT playlist_song_1 FOREIGN KEY (song_id)    REFERENCES
      song (id),
  CONSTRAINT playlist_song_2 FOREIGN KEY (playlist_id)
      REFERENCES playlist (id),
  CONSTRAINT playlist_song_3 FOREIGN KEY (owner_id)   REFERENCES
      user (id)
);
```

E’ stata aggiunta la Colonna “position” per abilitare la funzionalità di ordinamento manuale della playlist.
La stessa base di dati può essere utilizzata anche per la versione HTML puro, in quanto il dato aggiuntivo non interferisce in alcun modo con il funzionamento dell’applicazione.

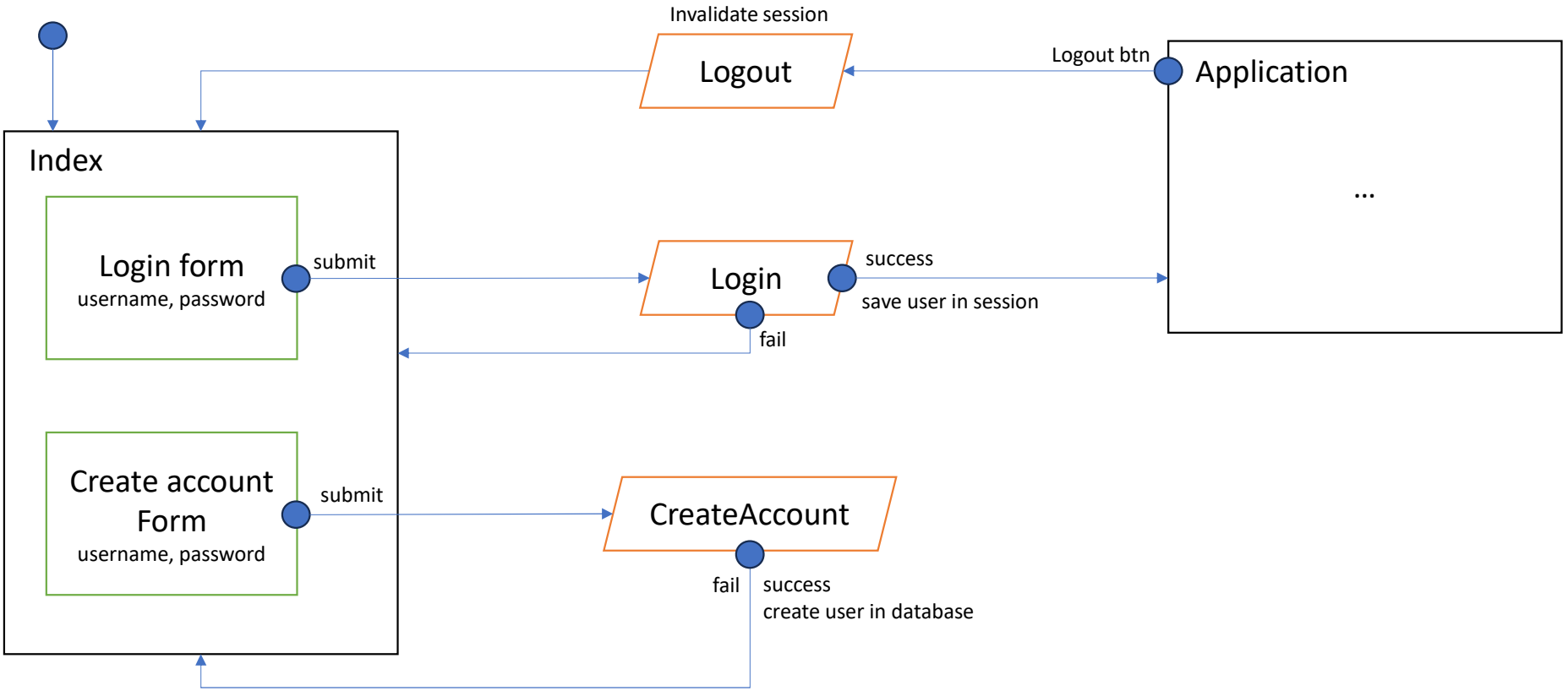
Componenti

- **Model objects** (JavaBeans)
  - Playlist
  - Song
  - User
- **Data Access Objects**
  - **UserDAO**
    - checkCredentials(usrn, pwd)
    - createAccount(usrn, pwd)
    - deleteAccount(userId)
    - findPlaylists(userId)
    - getAllUserSongs(userId, getAlbumCovers)
  - SongDAO
    - uploadSong(songName, albumName, artistName, year, genre, albumCover, audioFile, userId)
    - addSongToPlaylist(songId, playlistId, userId)
    - addSongsToPlaylist(songList, playlistId, userId)
    - getSong(songId, userId)
    - deleteSong(songId, userId)
    - checkSongOwner(songId, userId)
  - PlaylistDAO
    - createPlaylist(name, songList, userId)
    - getSongs(playlistId, userId, offset)
    - removeSongFromPlaylist(songId, playlistId, userId)
    - getPlaylistTitle(playlistId, userId)
    - deletePlaylist(playlistId, userId)
    - countSongs(playlistId, userId)
    - getSongsNotInPlaylist(userId, playlistId)

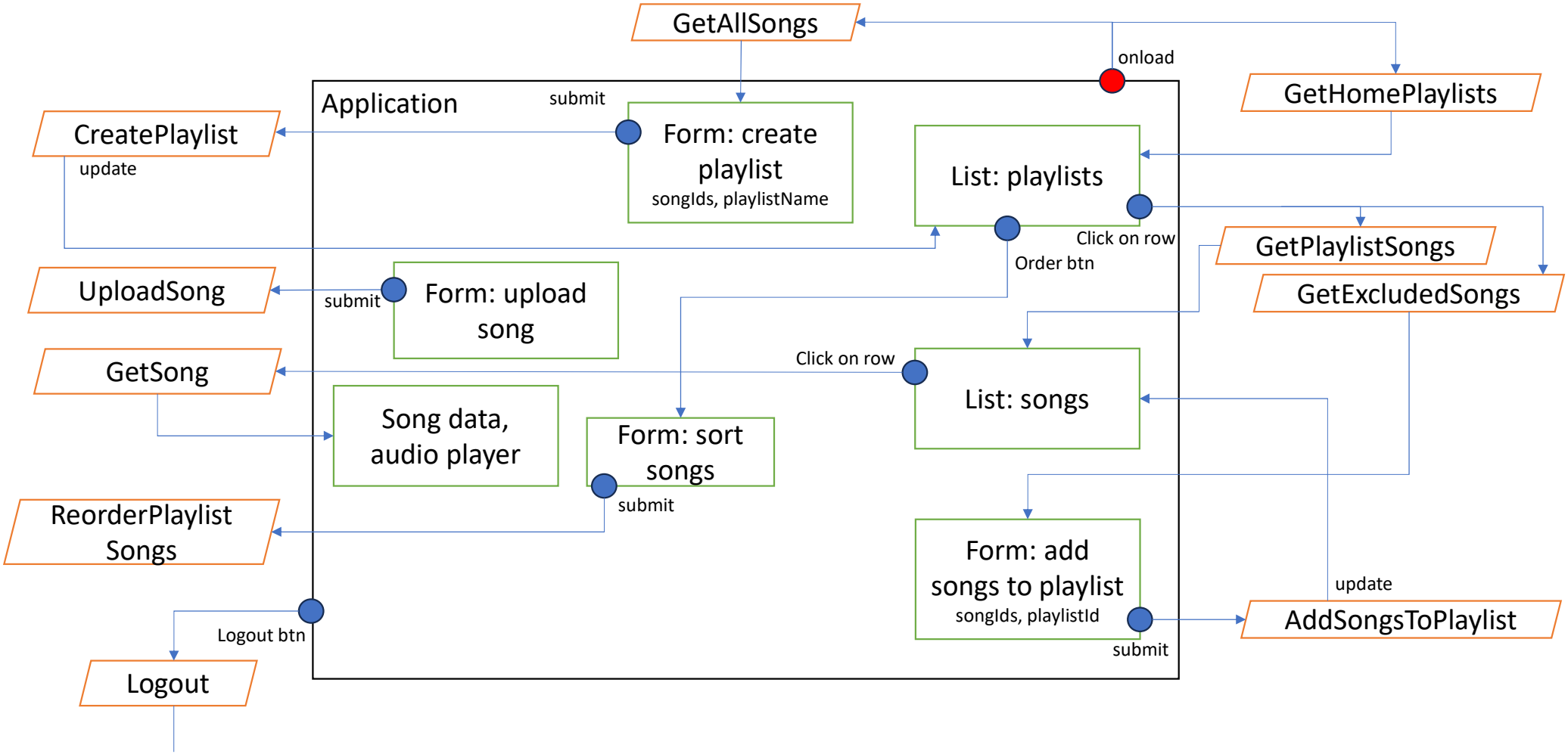
Componenti

- **Controllers** (servlets)
  - Authentication
    - Login
    - Logout
    - CreateAccount
    - DeleteAccount
  - Get content
    - GetAllSongs
    - GetExcludedSongs
    - GetHomePlaylists
    - GetPlaylistSongs
    - GetSong
  - Create content
    - AddSongsToPlaylist
    - CreatePlaylist
    - ReorderPlaylistSongs
    - UploadSong
  - Delete content
    - DeletePlaylist
    - DeleteSong
    - RemoveSongFromPlaylist
- **Views** (templates)
  - Index *(login and signup)*
  - *Home*
- **Utilities**
  - ConnectionHandler
  - LocalDateTimeAdapter

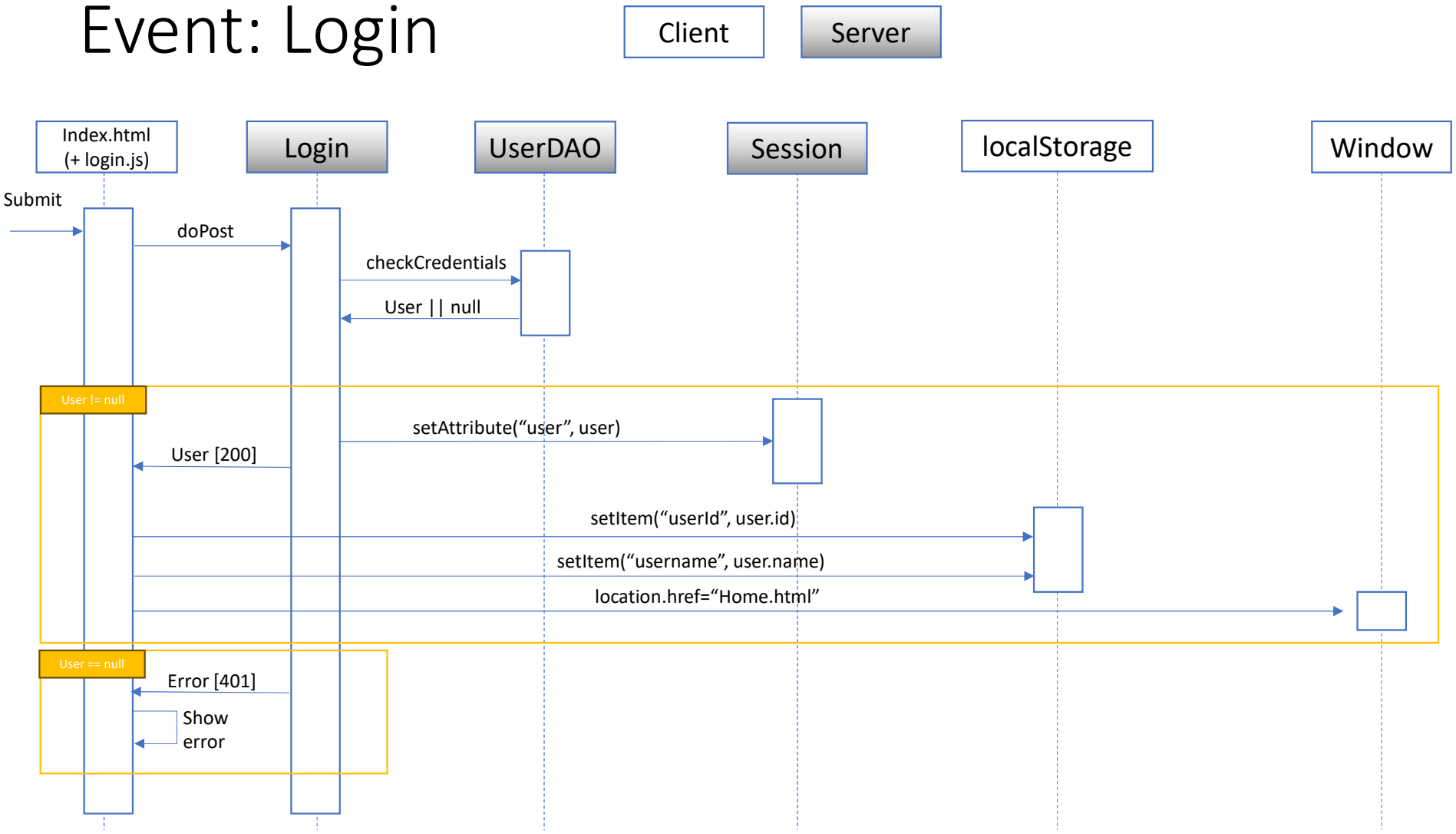
Application design (login/create account)



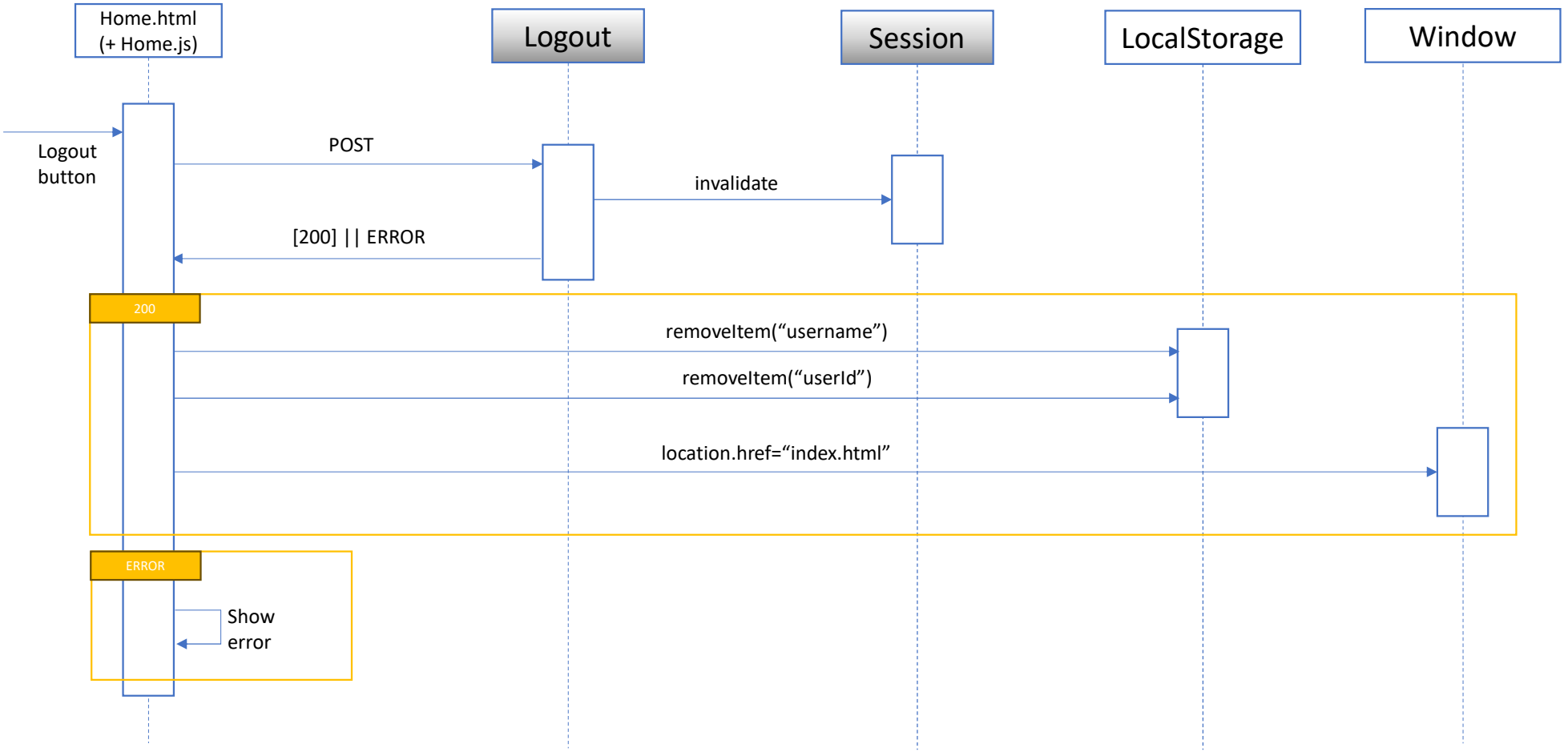
Application design



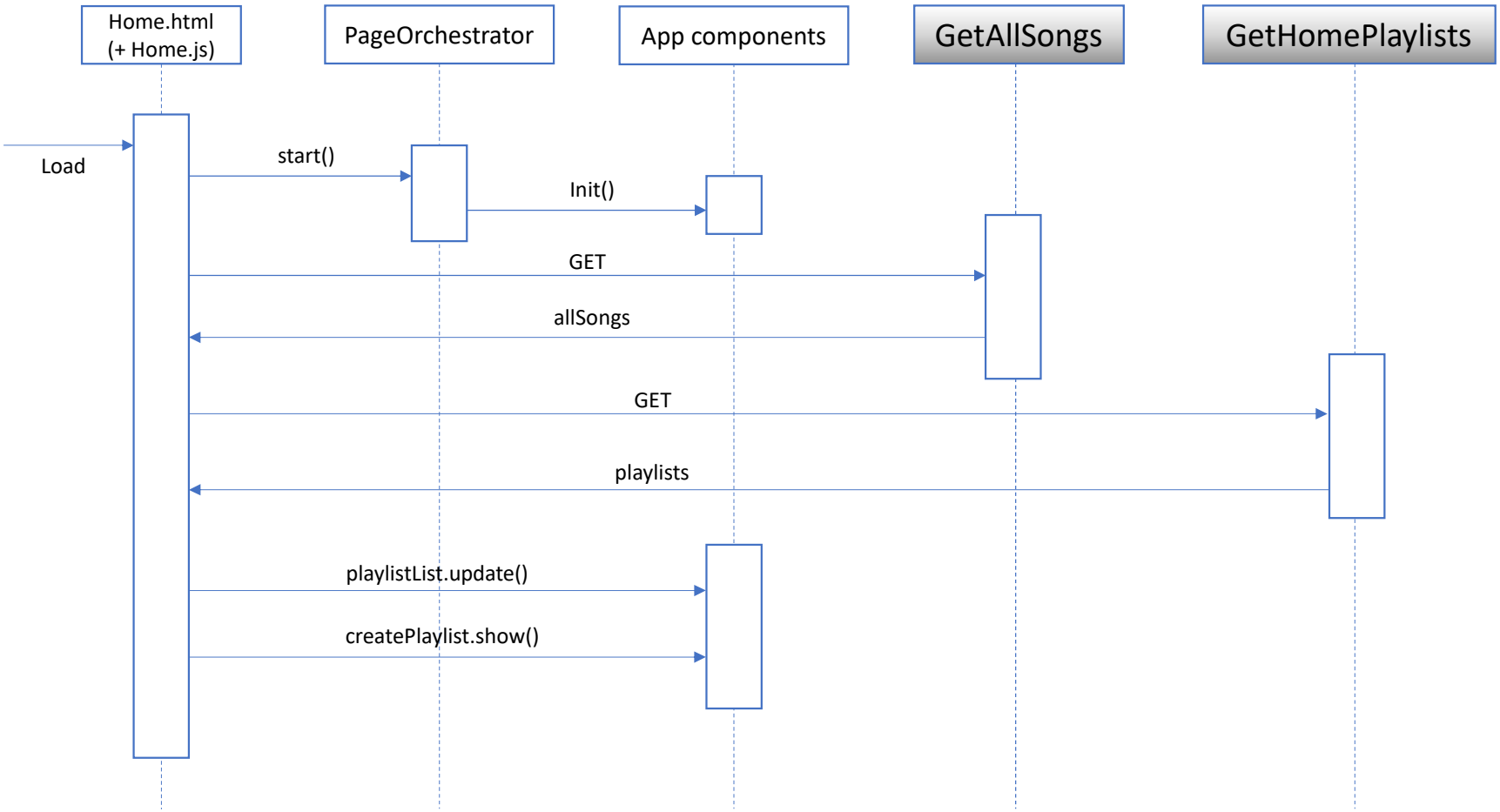
Event: Login


Client
Server
Index.html
(+ login.js)
Login
UserDAO
Session
localStorage
Window
Submit
doPost
checkCredentials
User || null
User != null
setAttribute(“user”, user)
User [200]
setItem(“userId”, user.id)
setItem(“username”, user.name)
location.href=“Home.html”
User == null
Error [401]
Show
error

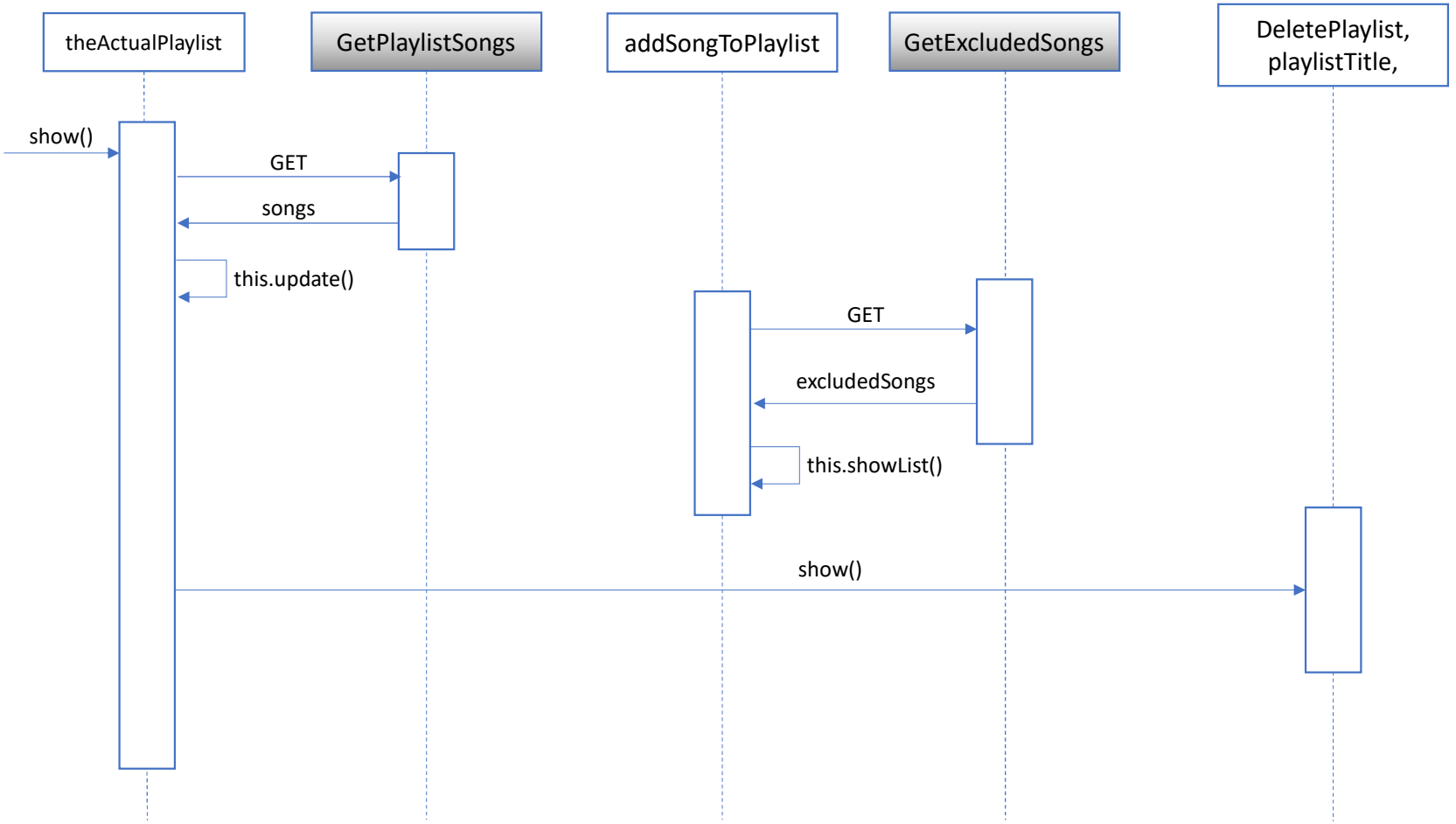
Event: Logout



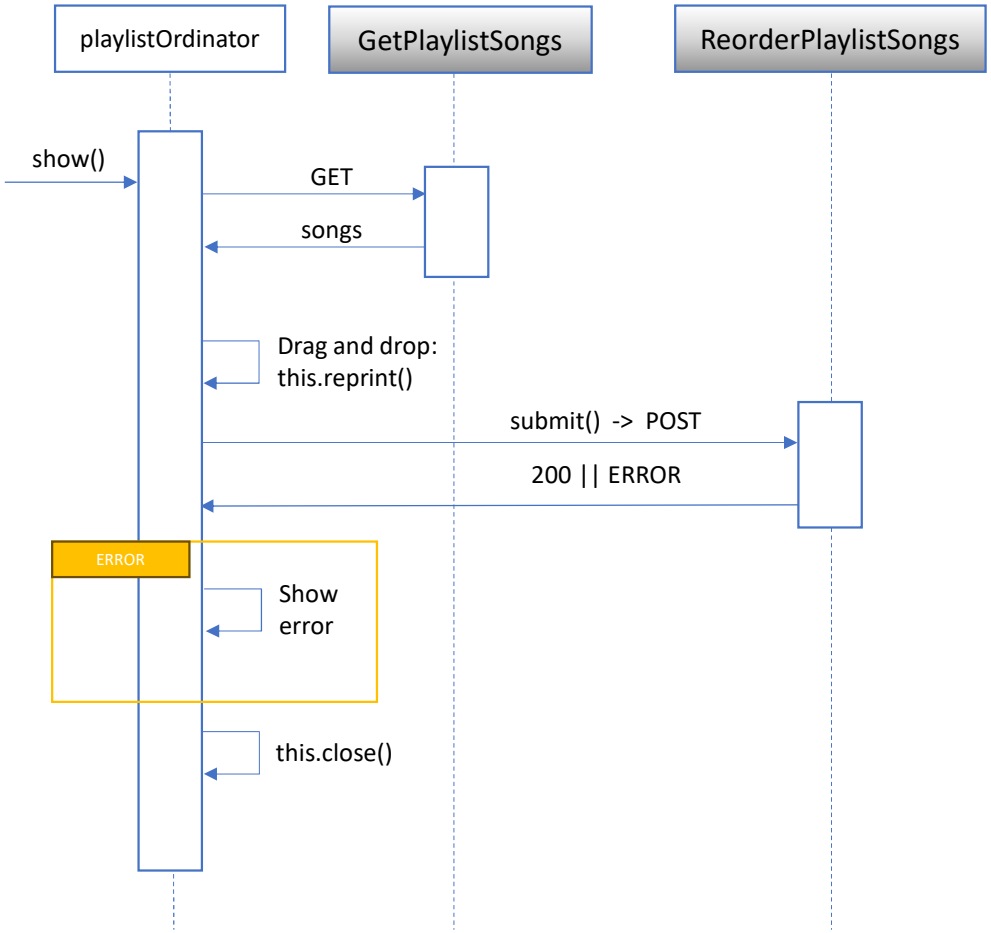
View: Application - onload



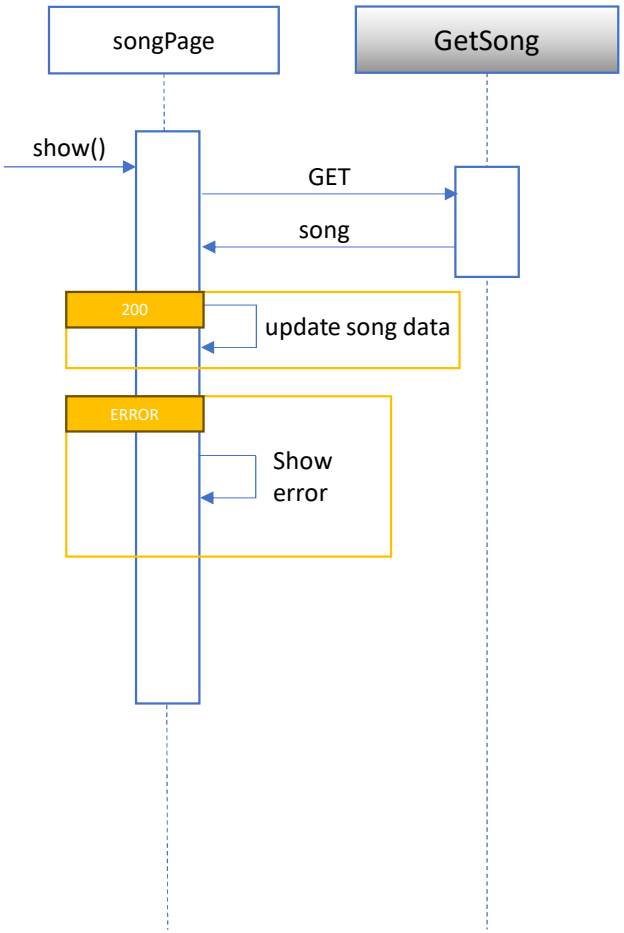
Event: click on playlist



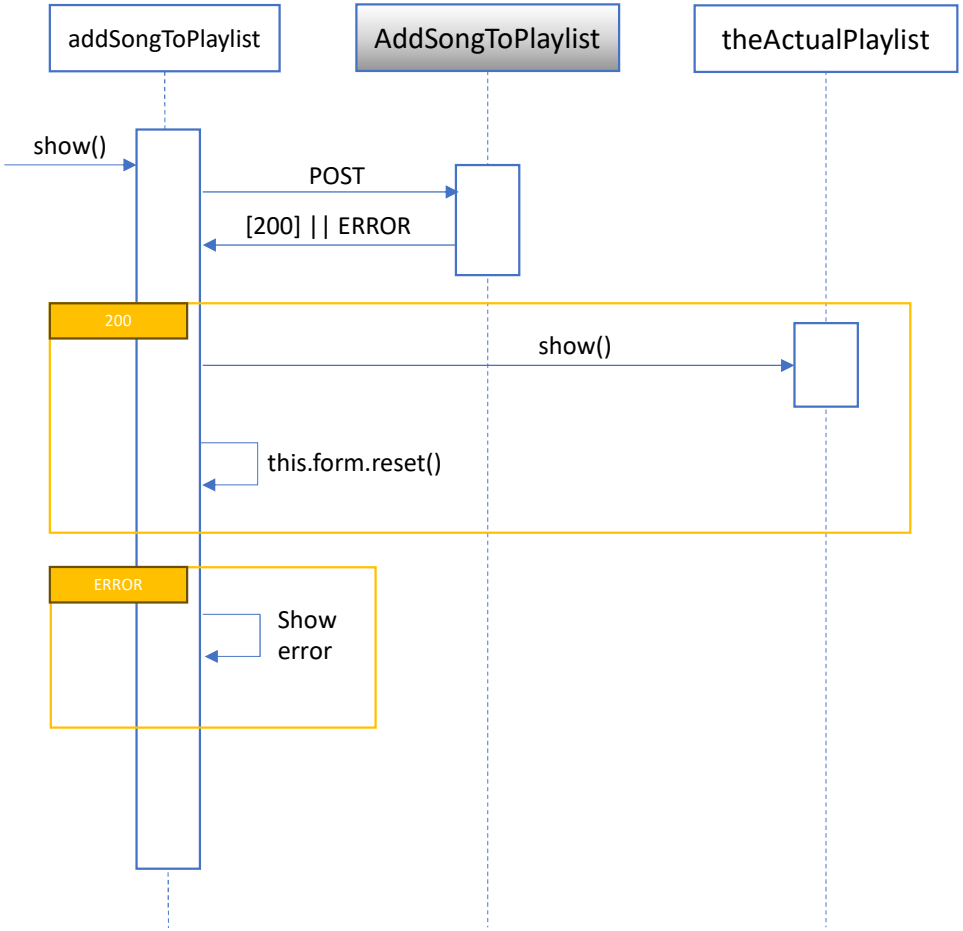
Event: reorder playlist



Event: click on song



Event: add song to playlist



Event: upload song