

Form Programming

Add a new Playlist

Playlists	
Sonatas	Delete Playlist
Concertos	Delete Playlist

Title
<input type="text" value="Title"/>
Add Playlist

Form to Add a Playlist

Title

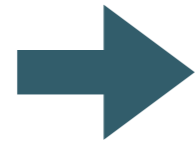

```
<form class="ui stacked segment form" action="/dashboard/addplaylist" method="POST">  
  <div class="field">  
    <label>Title</label>  
    <input placeholder="Title" type="text" name="title">  
  </div>  
  <button class="ui blue submit button">Add Playlist</button>  
</form>
```

action → **"/dashboard/addplaylist"**

Field Value → **name="title"**

Add Playlist

action



`"/dashboard/addplaylist"`

```
...  
router.post( '/dashboard/addplaylist', dashboard.addPlaylist );  
...
```

Router matches to action to a controller method

This method will be called when the 'Submit' button is
pressed

```
...  
router.post( '/dashboard/addplaylist', dashboard.addPlaylist);  
...
```

controller method to create a new playlist

```
const dashboard = {  
  ...  
  addPlaylist(request, response) {  
    const newPlayList = {  
      id: uuid(),  
      title: request.body.title,  
      songs: [],  
    };  
    playlistStore.addPlaylist(newPlayList);  
    response.redirect( '/dashboard' );  
  },  
};
```

Create a playlist object containing

- unique id
- title submitted in request
- empty songs array

```
const dashboard = {  
  ...  
  addPlaylist(request, response) {  
    const newPlayList = {  
      id: uuid(),  
      title: request.body.title,  
      songs: [],  
    };  
    playlistStore.addPlaylist(newPlayList);  
    response.redirect('/dashboard');  
  },  
};
```

add the new
playlist in the
store

```
const dashboard = {  
  ...  
  addPlaylist(request, response) {  
    const newPlayList = {  
      id: uuid(),  
      title: request.body.title,  
      songs: [],  
    };  
    playlistStore.addPlaylist(newPlayList);  
    response.redirect('/dashboard');  
  },  
};
```

Refresh the
Dashboard
(which will now
contain the
new playlist)

```
const dashboard = {  
  ...  
  addPlaylist(request, response) {  
    const newPlayList = {  
      id: uuid(),  
      title: request.body.title,  
      songs: [],  
    };  
    playlistStore.addPlaylist(newPlayList);  
    response.redirect('/dashboard');  
  },  
};
```


Add a new Song

Sonatas

Song	Artist	
Opus 23	Beethoven	Delete Song
Opus 33	Beethoven	Delete Song

Title

Artist

[Add Song](#)

Title Artist

Add Song

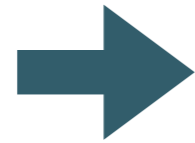
```
<form class="ui stacked segment form" action="/playlist/{{playlist.id}}/addsong" method="POST">
  <div class="two fields">
    <div class="field">
      <label>Title</label>
      <input placeholder="Title" type="text" name="title">
    </div>
    <div class="field">
      <label>Artist</label>
      <input placeholder="Artist" type="text" name="artist">
    </div>
  </div>
  <button class="ui blue submit button">Add Song</button>
</form>
```

action → `"/playlist/{{playlist.id}}/addsong"`

Field
Values → `name="title"`
`name="artist"`

Add Song

action



`"/playlist/{{playlist.id}}/addsong"`

```
...  
router.post('/playlist/:id/addsong', playlist.addSong);  
...
```

Router matches to action to a controller method

This method will be called when the 'Submit' button is pressed

The action contains the id of the playlist to which the song is to be added

Recover playlist id
from request

```
const playlist = {  
  ...  
  
  addSong(request, response) {  
    ← const playlistId = request.params.id;  
    const playlist = playlistStore.getPlaylist(playlistId);  
    const newSong = {  
      id: uuid(),  
      title: request.body.title,  
      artist: request.body.artist,  
    };  
    playlistStore.addSong(playlistId, newSong);  
    response.redirect('/playlist/' + playlistId);  
  },  
};
```

Get actual playlist
with this id from
store

```
const playlist = {  
  ...  
  
  addSong(request, response) {  
    const playlistId = request.params.id;  
    ← const playlist = playlistStore.getPlaylist(playlistId);  
    const newSong = {  
      id: uuid(),  
      title: request.body.title,  
      artist: request.body.artist,  
    };  
    playlistStore.addSong(playlistId, newSong);  
    response.redirect('/playlist/' + playlistId);  
  },  
};
```

Create a new Song object, with a unique id + the title and artist fields from the request

```
const playlist = {
  ...

  addSong(request, response) {
    const playlistId = request.params.id;
    const playlist = playlistStore.getPlaylist(playlistId);
    const newSong = {
      id: uuid(),
      title: request.body.title,
      artist: request.body.artist,
    };
    playlistStore.addSong(playlistId, newSong);
    response.redirect('/playlist/' + playlistId);
  },
};
```

Add this new
song to the store

```
const playlist = {
  ...

  addSong(request, response) {
    const playlistId = request.params.id;
    const playlist = playlistStore.getPlaylist(playlistId);
    const newSong = {
      id: uuid(),
      title: request.body.title,
      artist: request.body.artist,
    };
    playlistStore.addSong(playlistId, newSong);
    response.redirect('/playlist/' + playlistId);
  },
};
```

Refresh the
current view
(which will not
contain the new
song)

```
const playlist = {  
  ...  
  
  addSong(request, response) {  
    const playlistId = request.params.id;  
    const playlist = playlistStore.getPlaylist(playlistId);  
    const newSong = {  
      id: uuid(),  
      title: request.body.title,  
      artist: request.body.artist,  
    };  
    playlistStore.addSong(playlistId, newSong);  
    response.redirect('/playlist/' + playlistId);  
  },  
};
```