React / JSX

**DEADLINE:** 06/11/2018

## FOLDER STRUCTURE

FL\_9\_13\_homework\_react/\*

   homework/\*

       src/\*

           assets/\*

material-icons/\*

*all files inside the folder are required*

scss/\*

\_material-icons.scss\*

index.scss\*

index.html\*

index.js\*

.babelrc\*

.eslintrc.json\*

package.json\*

package-lock.json\*

webpack.config.js\*

## PREREQUISITES

Requires `node.js` and `npm` (node.js includes npm by default) to be installed.

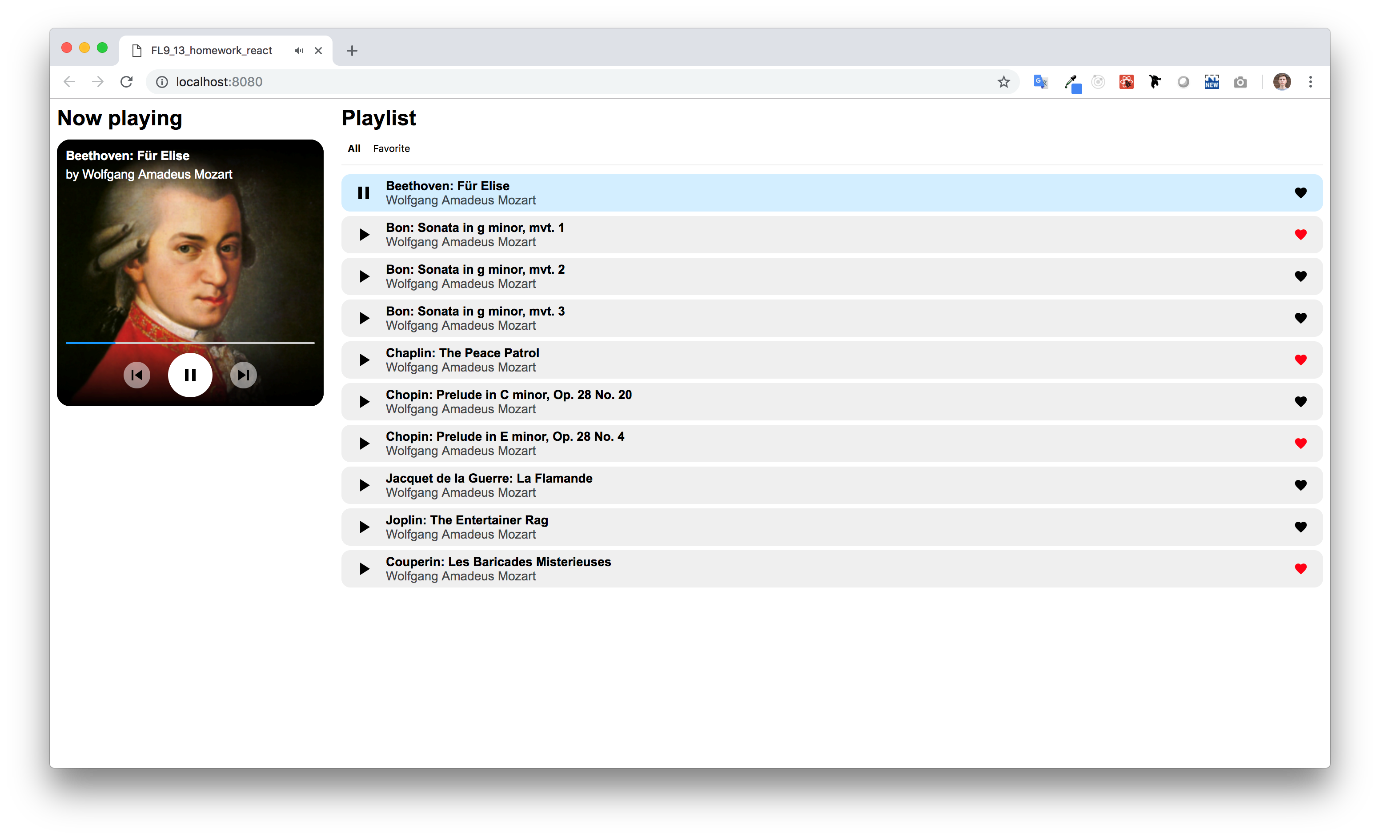
## HOW TO RUN THE PROJECT

1. Download homework
2. Extract files to your folder
3. Open cmd (terminal, bash)
4. Navigate to your folder which contains package.json file
5. Run `npm install` command
6. After all modules have been installed, use `npm run dev` command to run local server with project. (Please note, that `npm run dev` command requires webpack.config.js config file, don’t remove/move/change it)
7. The project should be opened automatically in your default web browser

## TASK

In this homework you need to create the music playlist using React.js library.

Take a look at the Figure 1.



##### Figure 1 — Music playlist page

This is the only one page that you need to create in the homework. On this page there are 3 main elements:

* The widget
* The playlist filter
* The list of music

*Don't worry if you can’t find these elements on the picture, there are many full-size images with examples under ./example folder, also demo video is presented in the same folder.*

Before starting to write any code, you need to understand the scope of the future work, try to find common (shared) logic between components and figure out how write clear and efficient code. *(*[*Think in React*](https://reactjs.org/docs/thinking-in-react.html) *article may be useful).*

After dividing the web page into components you are ready to start to write some code.

This homework requires some data to manipulate, in our case music and related metadata (e.g. author, title etc.). You can [fetch](https://developers.google.com/web/updates/2015/03/introduction-to-fetch) this data from the endpoint:

<https://fl-homework-api.firebaseio.com/mozart.json>

This endpoint returns a list of songs. Each item is the object which contains information about the song:

* id: number – the unique identifier
* author: string – the creator or originator
* title: string – the name of the song
* poster: string – the path to the poster image
* mp3: string – the path to the .mp3 file
* liked: boolean – the flag that shows if this song is included in the favorite playlist

You need to store this data somewhere in your container and use it where it needs.

The list of requirements is presented under ‘functional requirements’ title.

## FUNCTIONAL REQUIREMENTS

Feature: The playlist page

**Scenario: User navigates to the page for the first time**

Given: The page is opened for the first time

When: NA

Then: The playlist is available

And: The playlist filter is ‘All’ and available

And: The full list of music is available

**Scenario: User navigates to the page for the first time (data waiting)**

Given: The page is opened for the first time

When: User waiting for data fetching

Then: The ‘Playlist’ title is available

And: The ‘Loading music…’ message is displayed

**Scenario: User navigates to the page for the first time (data failure)**

Given: The page is opened for the first time

When: Data is failed to load

Then: The ‘Playlist’ title is available

And: The ‘Failed to load music’ message is displayed

And: The ‘Retry’ button is available

**Scenario: User navigates to the page for the first time (data failure -> retry)**

Given: The page is opened for the first time

When: Data failed to load and user presses ‘Retry’ button

Then: The ‘Playlist’ title is available

And: The ‘Loading music…’ message is displayed

**Scenario: User navigates to the page to listen the music**

Given: The page is opened for the first time

When: User press the ‘Play’ button in the playlist

Then: The ‘Now playing’ title is available

And: The music widget is available

And: The music progress bar in widget is available and increases every second

And: The music is playing

And: The current playing music is highlighted in the playlist

**Scenario: User stops the playing music by playlist controls**

Given: The music is playing

When: User press the ‘Pause’ button in the playlist

Then: The ‘Now playing’ title is available

And: The music widget is available

And: The music is paused and not playing

And: The current playing music still highlighted in the playlist

**Scenario: User stops the playing music by widget controls**

Given: The music is playing

When: User press the ‘Pause’ button in the widget

Then: The ‘Now playing’ title is available

And: The music widget is available

And: The music is paused and not playing

And: The current playing music is highlighted in the playlist

**Scenario: User resumes music by widget controls**

Given: The music is paused

When: User press the ‘Play’ button in the widget

Then: The ‘Now playing’ title is available

And: The music widget is available

And: The music is playing

And: The current playing music is highlighted in the playlist

**Scenario: User skips to the next track**

Given: The music is playing or paused

When: User press the ‘Skip next’ button in the widget

Then: The ‘Now playing’ title is available

And: The music widget is available

And: The next music is selected and highlighted in the playlist

And: The music is playing

**Scenario: User skips to the previous track**

Given: The music is playing or paused

When: User press the ‘Skip previous’ button in the widget

Then: The ‘Now playing’ title is available

And: The music widget is available

And: The previous music is selected and highlighted in the playlist

And: The music is playing

**Scenario: User skips to the next track when current playing track is the last in the list**

Given: The music is playing or paused

When: The current paying music is last in the playlist and user press the ‘Skip next’ button in the widget

Then: The ‘Now playing’ title is available

And: The music widget is available

And: The first music in the list is selected and highlighted in the playlist

And: The music is playing

**Scenario: User skips to the previous track when current playing track is the first in the list**

Given: The music is playing or paused

When: The current paying music is first in the playlist and user press the ‘Skip previous’ button in the widget

Then: The ‘Now playing’ title is available

And: The music widget is available

And: The last music in the list is selected and highlighted in the playlist

And: The music is playing

**Scenario: User marks the track as favorite**

Given: The page is opened for the first time

When: User clicks the ‘Favorite’ button in the playlist (e.g. Heart)

Then: The ‘Heart’ highlighted in red color

And: The track is added to the ‘Liked’ playlist

**Scenario: User remove track from favorite (filter – all)**

Given: The playlist with ‘All’ filter is available and some tracks are marked as favorite

When: User clicks the ‘Favorite’ button the second time on already favorite track (e.g. Heart)

Then: The ‘Heart’ highlighted in black color

And: The track is removed from the ‘Liked’ playlist

**Scenario: User navigates to the page for the first time (filter – liked)**

Given: The playlist with ‘Liked’ filter is available and some tracks are marked as favorite

When: User clicks the ‘Favorite’ button the second time on already favorite track (e.g. Heart)

Then: The ‘Heart’ highlighted in black color

And: The track is removed from the ‘Liked’ playlist and disappears from the `Liked` playlist

**Scenario: User navigates to the page for the first time and select ‘Liked’ playlist**

Given: The page is opened for the first time

When: User click the ‘Liked’ button in the filter

Then: ‘The playlist is empty’ is shown

## PLEASE NOTE

* Pixel perfect is not required
* The font for text is ‘Arial’

## RESTRICTIONS

* Using 3rd party tools is forbidden
* Using CSS frameworks is forbidden (e.g. Bootstrap, Materialize).
* Using JS libraries is forbidden.
* Using CSS is forbidden.
* The installation of additional packages is forbidden.

## HOW TO

* To run linter for the project use `npm run lint`

## BEFORE SUBMIT

* Remove all unnecessary files that you might have included by mistake
* Verify that all functionality is implemented according to requirements
* Add comments if the code is difficult to understand
* Fix warnings/errors in the browser console
* Verify that the name of the folders and files meet the requirements
* Make sure there are no errors/warnings in the browser console
* Run the linter and fix all warnings and errors

## SUBMIT

* The folder should be uploaded to github repository 'fl-9' into master branch

## USEFUL LINKS

* <https://reactjs.org/docs/hello-world.html>
* <https://medium.com/@dan_abramov/smart-and-dumb-components-7ca2f9a7c7d0>
* <https://medium.com/@thejasonfile/dumb-components-and-smart-components-e7b33a698d43>
* <https://blogs.perficient.com/2017/12/19/how-to-customize-your-own-html5-audio-player/>