Steps for building Bloc Jams, a single-page application(SPA). Disadvantage of SPA is that programmed logic typically encoded in the backend now has to be provided in the frontend. The many advantages oF SPA's are perceived faster performance (absence of flicker with page transitions), actual faster performance because fewer requests need to be processed with each page transition, support animated page transitions to make user experience more interesting plus signal that view content is updated, and allow backend to focus on serving data rather than rendering a user interface.

1. move to desired directory at command line and run `create-react-app bloc-jams-react` to create the initial build:

```
> create-react-app bloc-jams-react
```

2. move to the new app's directory and spin up the DevServer to confirm successful build:

```
> cd bloc-jams-react
> npm start
```

3. initialize git local and remote repos for the new application. Name remote repo `bloc-jams-react`. Go to GitHub and copy the new repo's URL to be pasted in the commands below:

```
> git init
> git add .
> git commit -m "initial commit" //local repo established
> git remote add origin https://github.com/<username>/bloc
-jams-react.git
> git push -u origin master
```

4. create an `assets` directory within the `public` directory of the project, so that the DevServer can access images/music files necessary for the application. Place the `image` and `music` folders containing these files into the new `assets` directory

5. install React Router using Node Package manager. Use the save modifier so that Router gets stored as a dependency of the app, and is added to the app's `package.json` file. This will allow cloning of repository, and running `npm install` so that the app's dependencies can be downloaded based on the listed dependences in `package.json`.
   React Router will manage the client-side page transitions for the application. The application's HTML will load once, and then Router will determine which React components are visible at any given time through its provided `Link` component. This component also updates the displayed URL in the address bar along with the page transitions. For a web app, use `react-router-dom`. If we were building a mobile app, would use `react-router-native`.

```
> cd bloc-jams-react
> npm install -S react-router-dom
```

6. import React Router’s `BrowserRouter` router into the project by adding an import statement to the top of the root file of the application, which is `index.js`. This is the router that updates the address bar’s URL with page transitions.

```
import {BrowserRouter} from 'react-router-dom';
```

7. Wrap the rendering of the `App` component within the root file `index.js` with a `BrowserRouter` component:

```
ReactDOM.render(
<BrowserRouter>
<App />
</BrowserRouter>
, document.getElementById('root')
);
```

8. Go to `App.js` and remove the boilerplate code in its render method, as well as the import statement for the React logo. Have the `App.js` component return a root `<div>` element with a class of `App`. Within the root element, return a `<header>` element with an `<h1>` heading of `Bloc Jams`, and then return a `<main>` section which is empty. Make sure there is an export statement at the bottom of the document:

```
import React, {Component} from 'react';
import './App.css';

```

```
class App extends Component{
    render(){
        return(
        <div className='App'>
        <header>
            <h1>Bloc Jams</h1>
        </header>
        <main>
        </main>
        </div>
        );
    }
}

export default App;
```

9. Import the `Route` and `Link` components from React Router into `App.js`:

```
import React, {Component} from 'react';
import {Route, Link} from 'react-router-dom';
import './App.css';

class App extends Component{
    render(){
        return(
        <div className='App'>
        <header>
```

```
                <h1>Bloc Jams</h1>
            </header>
            <main>
            </main>
            </div>
            );
    }
}

export default App;
```

10. Create 3 routes in `App.js` within the `<main>` section, one each for the landing page, library page, and album page. Remember that `Route` components require a `path` and `component` property. When address bar matches one of the `Route` component's path properties, it will render the listed components for that path. Remember to use `exact` keyword for paths which are shortened relative to other specified paths, and when you don't want a route that ignores any specified subdirectories of the route in the URL:

```
import React, {Component} from 'react';
import {Route, Link} from 'react-router-dom';
import './App.css';

class App extends Component{
    render(){
        return(
        <div className='App'>
        <header>
            <h1>Bloc Jams</h1>
        </header>
        <main>
            <Route exact path='/' component={Landing} />
            <Route path='/library' component={Library} />
            <Route path='/album' component={Album} />
        </main>
        </div>
        );
    }
}

export default App;
```

11. Go to the `src` directory and create a `components` directory. Move to `components` and then create new files which will contain `Landing`, `Library`, and `Album` components:

```
> cd src
> mkdir components
> cd components
> touch Landing.js Library.js Album.js
```

12. Go to `Landing.js` and create a very basic component. This will entail an import statement for React, a function that returns JSX, and an export statement:

```
import React from 'react';

const Landing = () => (
<section className='landing'>
Landing page will go here
</section>
);

export default Landing;
```

13. Create a similar basic component for the `Library` component within `Library.js`:

```
import React from 'react';

const Library = () => (
<section className='library'>
Library will go here
</section>
);

export default Library;
```

14. Create a similar basic component for the `Album` component within `Library.js`:

```
import React from 'react';

```

```

const Album = () => (
<section className='album'>
Album will go here
</section>
);

export default Album;
```

15. Import the newly-created `Landing` and `Library` components into `App.js`:

```
import React, {Component} from 'react';
import {Route, Link} from 'react-router-dom';
import Landing from './components/Landing';
import Library from './components/Library';
import './App.css';

class App extends Component{
    render(){
        return(
        <div className='App'>
        <header>
            <h1>Bloc Jams</h1>
        </header>
        <main>
            <Route exact path='/' component={Landing} />
            <Route path='/library' component={Library} />
        </main>
        </div>
        );
    }
}

export default App;
```

16. Have `App.js` render navigation links to the Landing and Library pages. Use the `Link` component to do this, noting that syntax is similar to an `<a>` element except uses the `to` keyword and should point to one of the routes we have defined:

```
import React, {Component} from 'react';
import {Route, Link} from 'react-router-dom';
import Landing from './components/Landing';
import Library from './components/Library';
import './App.css';

class App extends Component{
    render(){
        return(
        <div className='App'>
        <header>
            <nav>
                <Link to='/'>Landing</Link>
                <Link to='/library'>Library</Link>
            </nav>
            <h1>Bloc Jams</h1>
        </header>
        <main>
            <Route exact path='/' component={Landing} />
            <Route path='/library' component={Library} />
        </main>
        </div>
        );
    }
}

export default App;
```

17. In the `Landing` component, add a slogan and 3 selling points, each selling point comprising a title and a description. Since this is a relatively simple component, we can keep this component structured as a function rather than converting it to a class component. Remember when applying classes to the JSX to use `className` rather than `class` keyword:

```
import React from 'react';

const Landing = () => (
    <section className='landing'>
        <h1 className>Turn the music up!</h1>
    <section className='selling-points'>
        <div className='point'>
            <h2 className='point-title>Choose your music</
h2>
            <p className='point-description'>The world is
full of music; why should you have to listen to music that
 someone else chose?</p>
        </div>
        <div className='point'>
            <h2 className='point-title>Unlimited, streamin
g, ad-free</h2>
            <p className='point-description'>No arbitrary
limits. No distractions.</p>
        </div>
        <div className='point'>
            <h2 className='point-title>Mobile enabled</h2>
            <p className='point-description'>Listen to you
r music on the go. This streaming service is available on
all mobile platforms.</p>
        </div>
    </section>
    </section>
);

export default Landing;
```

18. Convert the `Library` component from a function to a React class-based component. The steps will be changing the import statement so that we are importing both React and its `Component`, and changing the `Library` function to a class that

includes a `render()` function that returns a single root element:

```
import React, {Component} from 'react';

class Library extends Component{
render(){
    return(
    <section className='library'>
Library will go here
</section>
    );
}
}

export default Library;
```

19. simulate a call to an API to retrieve album data by creating a new subdirectory within the `src` directory called `data`. Then create a file inside the `data` directory called `albums.js`. Paste data into the `albums.js` file which includes paths to the music files the application will use. Use the `export` keyword so that this data can be imported by the `Library` component:

```
> cd src
> mkdir data
> cd data
> touch albums.js
```

```
export default [{
title: 'The Colors',
  artist: 'Pablo Picasso',
  releaseInfo: '1909 Spanish Records',
  albumCover: '/assets/images/album_covers/01.jpg',
  slug: 'the-colors',
  songs: [
      { title: 'Blue', duration: '161.71', audioSrc: '/assets/music/blue.mp3' },
      { title: 'Green', duration: '103.96', audioSrc: '/assets/music/green.mp3' },
      { title: 'Red', duration: '268.45', audioSrc: '/assets/music/red.mp3' },
      { title: 'Pink', duration: '153.14', audioSrc: '/assets/music/pink.mp3' },
      { title: 'Magenta', duration: '374.22', audioSrc: '/assets/music/magenta.mp3' }
  ]
}, {
    title: 'The Telephone',
    artist: 'Guglielmo Marconi',
    releaseInfo: '1909 EM',
    albumCover: '/assets/images/album_covers/02.jpg',
    slug: 'the-telephone',
    songs: [
      { title: 'Blue', duration: '161.71', audioSrc: '/assets/music/blue.mp3' },
      { title: 'Green', duration: '103.96', audioSrc: '/assets/music/green.mp3' },
      { title: 'Red', duration: '268.45', audioSrc: '/assets/music/red.mp3' },
      { title: 'Pink', duration: '153.14', audioSrc: '/assets/music/pink.mp3' },
      { title: 'Magenta', duration: '374.22', audioSrc: '/assets/music/magenta.mp3' }
    ]
}];
```

20. import the data in `album.js` into the `Library.js` component:

```
import React, {Component} from 'react';
import albumData from './../data/albums';

class Library extends Component{
render(){
    return(
    <section className='library'>
Library will go here
</section>
    );
}
}

export default Library;
```

21. add constructor function to `Library.js` component, and have this component maintain state for the list of albums in the `albums.js` data file:

```
import React, {Component} from 'react';
import albumData from './../data/albums';

class Library extends Component{
    constructor(props){
        super(props);
        this.state={albums: albumData};
    }
    render(){
        return(
            <section className='library'>
                Library will go here
            </section>
        );
    }
}

export default Library;
```

22. have the `Library` component render its `albums` state property using the `.map()` function to iterate through the array. Display the album image, album title, artist, and number of songs on the album:

```
import React, {Component} from 'react';
import albumData from './../data/albums';

class Library extends Component{
    constructor(props){
        super(props);
        this.state={albums: albumData};
    }
    render(){
        return(
            <section className='library'>
                {
                    this.state.albums.map((album, index) =
>
                        <div key={index}>
                            <img src={album.albumCover} al
t={album.title} />
                            <div>{album.title}</div>
                            <div>{album.artist}</div>
                            <div>{album.songs.length} song
s</div>
                        </div>
                    )
                }
            </section>
        );
    }
}

export default Library;
```

23. We want to be able to click on a displayed album within the `Library` component to take us to that album’s page. Because there are multiple albums and each album page will have its own route, best to formulate the album route as being dynamic, where the subdirectory of the album URL is the name of the album. Accomplish this by changing the `Route` component for album in `App.js` to a URL parameter. Syntax will be a semi-colon followed by the word `slug`; `slug` is used to define text formatted to be acceptable in a different format, and will also serve as a unique id for each album:

```
import React, {Component} from 'react';
import {Route, Link} from 'react-router-dom';
import Landing from './components/Landing';
import Library from './components/Library';
import './App.css';

class App extends Component{
    render(){
        return(
        <div className='App'>
        <header>
            <nav>
                <Link to='/'>Landing</Link>
```

```
                    <Link to='/library'>Library</Link>
                </nav>
                <h1>Bloc Jams</h1>
            </header>
            <main>
                <Route exact path='/' component={Landing} />
                <Route path='/library' component={Library} />
                <Route path='/album/:slug' component={Album} /
>
            </main>
            </div>
            );
        }
}

export default App;
```

24. Modify the `Album.js` component to a class component so that it can work with the `slug` URL parameter. Remember conversion from function to class component requires importing both React and `Component`, making the class declaration, and making sure class includes a `render()` function that returns a single root element:

```
import React, {Component} from 'react';

class Album extends Component {
render(){
    return(
        <section className='album'>
            Album will go here
        </section>
    );
}
}

export default Album;
```

25. Allow `Album.js` component to access the URL parameter specified in its parent `App.js` component by passing the `slug` through props using the `this.props.match.params` object:

```
import React, {Component} from 'react';

class Album extends Component {
render(){
    return(
        <section className='album'>
            {this.props.match.params.slug} Album will go h
ere
        </section>
    );
}
}

export default Album;
```

26. Add links to each album displayed in the `Library` component using the `Link` component of React Router. This will require importing the `Link` component to the `Library` component and then wrapping the rendered information with the `Link` component, using the `slug` URL parameter. Define the path for each album using a template literal:

```
import React, {Component} from 'react';
import {Link} from 'react-router-dom';
import albumData from './../data/albums';

class Library extends Component{
    constructor(props){
        super(props);
        this.state={albums: albumData};
    }
    render(){
        return(
            <section className='library'>
                {
                    this.state.albums.map((album, index) =
>
                        <Link to={'/album/${album.slug}' k
ey={index}>
                            <img src={album.albumCover} al
t={album.title} />
                            <div>{album.title}</div>
                            <div>{album.artist}</div>
                            <div>{album.songs.length} song
s</div>
                        </Link>
                    )
                }
            </section>
        );
    }
}

export default Library;
```

27. Set state on `Album` component to the matching album object. First import `albumData` into `Album` component. Then add `constructor` method.

```
import React, {Component} from 'react';
import albumData from './../data/albums';

class Album extends Component {
constructor(props){
   super(props);
}
render(){
   return(
       <section className='album'>
           {this.props.match.params.slug} Album will go he
re
        </section>
    );
}
}

export default Album;
```

28. Find the album object in albumData with the `slug` property matching the URL parameter `this.props.match.params.slug`, so that we can set an `album` property on `Album.js` state. Use the `.find()` array method and then use `.setState()` method to assign this album object to the `album` state property:

```
import React, {Component} from 'react';
import albumData from './../data/albums';

class Album extends Component {
constructor(props){
    super(props);
    const album = albumData.find( album => {
      return album.slug === this.props.match.params.slug
    });
    this.state = {album: album};
    }

render(){
    return(
        <section className='album'>
            {this.props.match.params.slug} Album will go he
re
        </section>
    );
}
}

export default Album;
```

29. add the HTML to `Album.js` to render the album image, title, artist, and release information:

```
import React, {Component} from 'react';
import albumData from './../data/albums';

class Album extends Component {
constructor(props){
   super(props);
   const album = albumData.find( album => {
     return album.slug === this.props.match.params.slug
   });
   this.state = {album: album};
   }

render(){
   return(
        <section className='album'>
```

```
            <section id='album-info'>
                <img id='album-cover-art' />
                <div className='album-details'>
                    <h1 id='album-title'></h1>
                    <h2 className='artist'></h2>
                    <div id='release-info'></div>
                </div>
            </section>
        </section>
    );
}
}

export default Album;
```

30. Use the `albumCover`, `title`, `artist`, and `releaseInfo` properties of the album held in state by `Album.js` to add the necessary JSX to fully render the album information by `Album.js`. Remember to use curly braces to reference these object properties:

```
import React, {Component} from 'react';
import albumData from './../data/albums';

class Album extends Component {
constructor(props){
   super(props);
   const album = albumData.find( album => {
     return album.slug === this.props.match.params.slug
   });
   this.state = {album: album};
   }

render(){
   return(
       <section className='album'>
           <section id='album-info'>
               <img id='album-cover-art' src={this.state.a
lbum.albumCover}/>
               <div className='album-details'>
                   <h1 id='album-title'>{this.state.album.
title}</h1>
                   <h2 className='artist'>{this.state.albu
m.artist}</h2>
                   <div id='release-info'>{this.state.albu
m.releaseInfo}</div>
               </div>
           </section>
       </section>
   );
}
}

export default Album;
```

31. Create a table that will contain the list of songs for the album:

```
import React, {Component} from 'react';
import albumData from './../data/albums';

class Album extends Component {
constructor(props){
   super(props);
   const album = albumData.find( album => {
     return album.slug === this.props.match.params.slug
   });
   this.state = {album: album};
   }

render(){
   return(
       <section className='album'>
           <section id='album-info'>
               <img id='album-cover-art' src={this.state.album.albumCover}/>
               <div className='album-details'>
                   <h1 id='album-title'>{this.state.album.title}</h1>
                   <h2 className='artist'>{this.state.album.artist}</h2>
                   <div id='release-info'>{this.state.album.releaseInfo}</div>
               </div>
```

```
            </section>
            <table id='song-list'>
                <colgroup>
                    <col id='song-number-column' />
                    <col id='song-title-column' />
                    <col id='song-duration-column' />
                </colgroup>
                <tbody>
                </tbody>
            </table>
        </section>
    );
}
}

export default Album;
```

32. Add code to the `Album.js` component to display the song list inside the `<tbody>` element. Use the `.map()` method and give each row a unique `key` value:

```
import React, {Component} from 'react';
import albumData from './../data/albums';

class Album extends Component {
constructor(props){
    super(props);
    const album = albumData.find( album => {
      return album.slug === this.props.match.params.slug
    });
    this.state = {album: album};
    }

render(){
    return(
        <section className='album'>
            <section id='album-info'>
                <img id='album-cover-art' src={this.state.
album.albumCover}/>
                <div className='album-details'>
                    <h1 id='album-title'>{this.state.album
.title}</h1>
                    <h2 className='artist'>{this.state.alb
um.artist}</h2>
                    <div id='release-info'>{this.state.alb
um.releaseInfo}</div>
                </div>
            </section>
            <table id='song-list'>
                <colgroup>
                    <col id='song-number-column' />
                    <col id='song-title-column' />
                    <col id='song-duration-column' />
                </colgroup>
                <tbody>
                    {this.state.album.songs.map((song, ind
ex) => {
                        <tr key={index}>
                            <td>
                                {index+1}
                            </td>
                            <td>
                                {song.title}
                            </td>
                            <td>
                                {song.duration}
                            </td>
                        </tr>
                    }
                    )
                    }
                </tbody>
            </table>
        </section>
    );
}
}

export default Album;
```

33. Create `<Audio>` element in `Album.js` within `constructor()` method. Since only playing one file at a time, only need one element. Note that this element will NOT be held in state. This

type of element has numerous properties but will use the following:

- `src` property: URL of the audio file to play. Change URL to switch song
- `volume` property: volume level ranging from 0.0 to 1.0
- `currentTime` property: current playback time in seconds. Can change to skip forward or backward
- `timeupdate` event: as `currentTime` changes, this event triggers. Use to update player bar with current time as song plays
- `durationchange` event: when `duration` changes, this event triggers. Use to update the duration when song changes
- `play()` method: begin playback of audio starting at `currentTime`
- `pause()` method: pauses playback
- `volumecontrol` event: triggered when volume level is changed

```
import React, {Component} from 'react';
import albumData from './../data/albums';

class Album extends Component {
constructor(props){
    super(props);
    const album = albumData.find( album => {
      return album.slug === this.props.match.params.slug
    });
    this.state = {album: album};
    this.audioElement = document.createElement('audio');
    }

render(){
    return(
        <section className='album'>
            <section id='album-info'>
                <img id='album-cover-art' src={this.state.
album.albumCover}/>
                <div className='album-details'>
                    <h1 id='album-title'>{this.state.album
.title}</h1>
                    <h2 className='artist'>{this.state.alb
um.artist}</h2>
                    <div id='release-info'>{this.state.alb
um.releaseInfo}</div>
                </div>
            </section>
            <table id='song-list'>
                <colgroup>
                    <col id='song-number-column' />
                    <col id='song-title-column' />
                    <col id='song-duration-column' />
                </colgroup>
                <tbody>
                    {this.state.album.songs.map((song, ind
ex) => {
```

```
                        <tr key={index}>
                            <td>
                                {index+1}
                            </td>
                            <td>
                                {song.title}
                            </td>
                            <td>
                                {song.duration}
                            </td>
                        </tr>
                    }
                    )
                    }
                </tbody>
            </table>
        </section>
    );
}
}

export default Album;
```

34. Set `src` property of `this.audioElement` to the audio source of the first song on the album:

```
import React, {Component} from 'react';
import albumData from './../data/albums';
```

```
class Album extends Component {
constructor(props){
    super(props);
    const album = albumData.find( album => {
      return album.slug === this.props.match.params.slug
    });
    this.state = {album: album};
    this.audioElement = document.createElement('audio');
    this.audioElement.src = album.songs[0].audioSrc;
    }

render(){
    return(
        <section className='album'>
            <section id='album-info'>
                <img id='album-cover-art' src={this.state.
album.albumCover}/>
                <div className='album-details'>
                    <h1 id='album-title'>{this.state.album
.title}</h1>
                    <h2 className='artist'>{this.state.alb
um.artist}</h2>
                    <div id='release-info'>{this.state.alb
um.releaseInfo}</div>
                </div>
            </section>
            <table id='song-list'>
```

```
                <colgroup>
                    <col id='song-number-column' />
                    <col id='song-title-column' />
                    <col id='song-duration-column' />
                </colgroup>
                <tbody>
                    {this.state.album.songs.map((song, ind
ex) => {
                        <tr key={index}>
                            <td>
                                {index+1}
                            </td>
                            <td>
                                {song.title}
                            </td>
                            <td>
                                {song.duration}
                            </td>
                        </tr>
                    }
                    )
                    }
                </tbody>
            </table>
        </section>
    );
}
}
```

```
export default Album;
```

35. We want to hold the song data and whether the song is playing in state. Add these two properties to state (`currentSong` and `isPlaying`). Set default values for these to first song on album, and to false respectively:

```
import React, {Component} from 'react';
import albumData from './../data/albums';

class Album extends Component {
constructor(props){
    super(props);
    const album = albumData.find( album => {
      return album.slug === this.props.match.params.slug
    });
    this.state = {
    album: album,
    currentSong: album.songs[0],
    isPlaying: false
    };
    this.audioElement = document.createElement('audio');
    this.audioElement.src = album.songs[0].audioSrc;
    }

render(){
    return(
        <section className='album'>
            <section id='album-info'>
                <img id='album-cover-art' src={this.state.album.albumCover}/>
                <div className='album-details'>
                    <h1 id='album-title'>{this.state.album.title}</h1>
                    <h2 className='artist'>{this.state.album.artist}</h2>
                    <div id='release-info'>{this.state.album.releaseInfo}</div>
                </div>
            </section>
            <table id='song-list'>
                <colgroup>
                    <col id='song-number-column' />
                    <col id='song-title-column' />
                    <col id='song-duration-column' />
                </colgroup>
                <tbody>
                    {this.state.album.songs.map((song, index) => {
                        <tr key={index}>
                            <td>
                                {index+1}
                            </td>
                            <td>
                                {song.title}
                            </td>
                            <td>
                                {song.duration}
                            </td>
                        </tr>
                    }
                    )
                    }
                </tbody>
            </table>
        </section>
    );
}
}

export default Album;
```

36. Create a `play()` method for `audioElement` such that we can play the selected song and also update the state of `isPlaying`. Also add a `pause()` method for pausing a song and updating state of `isPlaying`:

```
import React, {Component} from 'react';
import albumData from './../data/albums';

class Album extends Component {
constructor(props){
    super(props);
```

```
     const album = albumData.find( album => {
       return album.slug === this.props.match.params.slug
     });
     this.state = {
     album: album,
     currentSong: album.songs[0],
     isPlaying: false
     };
     this.audioElement = document.createElement('audio');
     this.audioElement.src = album.songs[0].audioSrc;
     }


     play(){
         this.audioElement.play();
         this.setState({isPlaying: true});
     }


     pause(){
         this.audioElement.pause();
         this.setState({isPlaying: false});
     }


     setSong(song){
         this.audioElement.src = song.audioSrc;
         this.setState({currentSong: song});
     }


render(){
     return(
         <section className='album'>
             <section id='album-info'>
                 <img id='album-cover-art' src={this.state.
 album.albumCover}/>
                 <div className='album-details'>
                     <h1 id='album-title'>{this.state.album
 .title}</h1>
                     <h2 className='artist'>{this.state.alb
 um.artist}</h2>
                     <div id='release-info'>{this.state.alb
 um.releaseInfo}</div>
                 </div>
             </section>
             <table id='song-list'>
                 <colgroup>
                     <col id='song-number-column' />
                     <col id='song-title-column' />
                     <col id='song-duration-column' />
                 </colgroup>
                 <tbody>
                     {this.state.album.songs.map((song, ind
 ex) => {
                         <tr key={index}>
                             <td>
                                 {index+1}
                             </td>
                             <td>
```

```
                                        {song.title}
                                    </td>
                                    <td>
                                        {song.duration}
                                    </td>
                                </tr>
                            }
                            )
                            }
                        </tbody>
                    </table>
                </section>
        );
}
}

export default Album;
```

37. Create a method `setSong()` in the `Album` component that takes the song object as a parameter and updates the `src` property of `audioElement` and updates the state of `currentSong`:

```
import React, {Component} from 'react';
import albumData from './../data/albums';

class Album extends Component {
constructor(props){
    super(props);
    const album = albumData.find( album => {
      return album.slug === this.props.match.params.slug
    });
    this.state = {
    album: album,
    currentSong: album.songs[0],
    isPlaying: false
    };
    this.audioElement = document.createElement('audio');
    this.audioElement.src = album.songs[0].audioSrc;
    }

    play(){
        this.audioElement.play();
        this.setState({isPlaying: true});
    }

    pause(){
        this.audioElement.pause();
        this.setState({isPlaying: false});
    }

    setSong(song){
        this.audioElement.src = song.
        this.setState({currentSong: song });
    }

```

```
render(){
    return(
        <section className='album'>
            <section id='album-info'>
                <img id='album-cover-art' src={this.state.
album.albumCover}/>
                <div className='album-details'>
                    <h1 id='album-title'>{this.state.album
.title}</h1>
                    <h2 className='artist'>{this.state.alb
um.artist}</h2>
                    <div id='release-info'>{this.state.alb
um.releaseInfo}</div>
                </div>
            </section>
            <table id='song-list'>
                <colgroup>
                    <col id='song-number-column' />
                    <col id='song-title-column' />
                    <col id='song-duration-column' />
                </colgroup>
                <tbody>
                    {this.state.album.songs.map((song, ind
ex) => {
                        <tr key={index}>
                            <td>
                                {index+1}
                            </td>
                            <td>
                                {song.title}
                            </td>
                            <td>
                                {song.duration}
                            </td>
                        </tr>
                    }
                    )
                    }
                </tbody>
            </table>
        </section>
    );
}
}

export default Album;
```

38. Create a `handleSongClick()` method in `Album` component that functions depending on whether user clicks current song and whether a song is currently playing. We want player to pause if user clicks on current song that is playing and to play if user clicks on currently-paused song; if user clicks on a different song, want this different song to play instead. Construct by first determining if user is clicking on `currentSong`. Then depending on whether song `isPlaying` and if current song, can

code whether to pause or to play:

```
import React, {Component} from 'react';
import albumData from './../data/albums';

class Album extends Component {
constructor(props){
    super(props);
    const album = albumData.find( album => {
      return album.slug === this.props.match.params.slug
    });
    this.state = {
    album: album,
    currentSong: album.songs[0],
    isPlaying: false
    };
    this.audioElement = document.createElement('audio');
    this.audioElement.src = album.songs[0].audioSrc;
    }

    play(){
        this.audioElement.play();
        this.setState({isPlaying: true});
    }

    pause(){
        this.audioElement.pause();
        this.setState({isPlaying: false});
    }

    setSong(song){
        this.audioElement.src = song.
        this.setState({currentSong: song });
    }

    handleSongClick(song){
        const isSameSong = this.state.currentSong === song
;
        if(this.state.isPlaying && isSameSong){
            this.pause();
        } else {
            this.play();
        }
    }

render(){
    return(
        <section className='album'>
            <section id='album-info'>
                <img id='album-cover-art' src={this.state.
album.albumCover}/>
                <div className='album-details'>
                    <h1 id='album-title'>{this.state.album
.title}</h1>
                    <h2 className='artist'>{this.state.alb
um.artist}</h2>
```

```
                    <div id='release-info'>{this.state.alb
um.releaseInfo}</div>
                </div>
            </section>
            <table id='song-list'>
                <colgroup>
                    <col id='song-number-column' />
                    <col id='song-title-column' />
                    <col id='song-duration-column' />
                </colgroup>
                <tbody>
                    {this.state.album.songs.map((song, ind
ex) => {
                        <tr key={index}>
                            <td>
                                {index+1}
                            </td>
                            <td>
                                {song.title}
                            </td>
                            <td>
                                {song.duration}
                            </td>
                        </tr>
                    }
                    )
                    }
                </tbody>
            </table>
        </section>
    );
}
}

export default Album;
```

39. Add event listener to the `<tr>` element for each listed song that will invoke the `handleSongClick()` event-handler function:

```
import React, {Component} from 'react';
import albumData from './../data/albums';

class Album extends Component {
constructor(props){
    super(props);
    const album = albumData.find( album => {
      return album.slug === this.props.match.params.slug
    });
    this.state = {
    album: album,
    currentSong: album.songs[0],
    isPlaying: false
    };
    this.audioElement = document.createElement('audio');
    this.audioElement.src = album.songs[0].audioSrc;
    }

    play(){
        this.audioElement.play();
        this.setState({isPlaying: true});
    }

    pause(){
        this.audioElement.pause();
        this.setState({isPlaying: false});
    }

    setSong(song){
        this.audioElement.src = song.
        this.setState({currentSong: song });
    }

    handleSongClick(song){
        const isSameSong = this.state.currentSong === song
;
        if(this.state.isPlaying && isSameSong){
            this.pause();
        } else {
            this.play();
        }
    }

render(){
    return(
        <section className='album'>
            <section id='album-info'>
                <img id='album-cover-art' src={this.state.
album.albumCover}/>
                <div className='album-details'>
                    <h1 id='album-title'>{this.state.album
.title}</h1>
                    <h2 className='artist'>{this.state.alb
um.artist}</h2>
                    <div id='release-info'>{this.state.alb
um.releaseInfo}</div>
                </div>
            </section>
            <table id='song-list'>
                <colgroup>
                    <col id='song-number-column' />
                    <col id='song-title-column' />
                    <col id='song-duration-column' />
                </colgroup>
                <tbody>
                    {this.state.album.songs.map((song, ind
ex) => {
                        <tr key={index}
                            onClick={()=> this.handleSongC
lick(song)}
                        >
                            <td>
                                {index+1}
                            </td>
                            <td>
                                {song.title}
                            </td>
                            <td>
                                {song.duration}
                            </td>
                        </tr>
                    }
                    )
                    }
                </tbody>
            </table>
        </section>
    );
}
}

export default Album;
```

40. If user clicks different song at this point, player doesn’t switch to the new song and begin playing. Add a line to the `handleSongClick()` function so that the current song is changed to the new song:

```
import React, {Component} from 'react';
import albumData from './../data/albums';

```

```
class Album extends Component {
constructor(props){
    super(props);
    const album = albumData.find( album => {
      return album.slug === this.props.match.params.slug
    });
    this.state = {
    album: album,
    currentSong: album.songs[0],
    isPlaying: false
    };
    this.audioElement = document.createElement('audio');
    this.audioElement.src = album.songs[0].audioSrc;
    }

    play(){
        this.audioElement.play();
        this.setState({isPlaying: true});
    }

    pause(){
        this.audioElement.pause();
        this.setState({isPlaying: false});
    }

    setSong(song){
        this.audioElement.src = song.
        this.setState({currentSong: song });
    }

    handleSongClick(song){
        const isSameSong = this.state.currentSong === song
;
        if(this.state.isPlaying && isSameSong){
            this.pause();
        } else {
            if(!isSameSong){this.setSong(song);}
            this.play();
        }
    }

render(){
    return(
        <section className='album'>
            <section id='album-info'>
                <img id='album-cover-art' src={this.state.
album.albumCover}/>
                <div className='album-details'>
                    <h1 id='album-title'>{this.state.album
.title}</h1>
                    <h2 className='artist'>{this.state.alb
um.artist}</h2>
                    <div id='release-info'>{this.state.alb
um.releaseInfo}</div>
                </div>
            </section>
            <table id='song-list'>
                <colgroup>
                    <col id='song-number-column' />
                    <col id='song-title-column' />
                    <col id='song-duration-column' />
                </colgroup>
                <tbody>
                    {this.state.album.songs.map((song, ind
ex) => {
                        <tr key={index}
                            onClick={()=> this.handleSongC
lick(song)}
                        >
                            <td>
                                {index+1}
                            </td>
                            <td>
                                {song.title}
                            </td>
                            <td>
                                {song.duration}
                            </td>
                        </tr>
                    }
                    )
                    }
                </tbody>
            </table>
```

```
        </section>
    );
}
}

export default Album;
```

41. Add features such that when cursor hovers over a song, it displays a play button in place of the song number. Also add feature that will display a pause button in place of song number for the currently-playing song. Finally, add feature so that a paused song displays a play button in place of the song number. Use the Ionicons to add the icons to the project, utilizing the CDN by adding this to the `public/index.html` page public/index.html:

```
<html lang="en">
  <head>
    <meta charset="utf-8" />
    <link rel="shortcut icon" href="%PUBLIC_URL%/favicon.i
co" />
    <meta
      name="viewport"
      content="width=device-width, initial-scale=1, shrink
-to-fit=no"
    />
    <meta name="theme-color" content="#000000" />
    <!--
```

```
      manifest.json provides metadata used when your web app is installed on a
      user's mobile device or desktop. See https://developers.google.com/web/fundamentals/web-app-manifest/
    -->
    <link rel="manifest" href="%PUBLIC_URL%/manifest.json" />
    <!--
      Notice the use of %PUBLIC_URL% in the tags above.
      It will be replaced with the URL of the `public` folder during the build.
      Only files inside the `public` folder can be referenced from the HTML.
      Unlike "/favicon.ico" or "favicon.ico", "%PUBLIC_URL%/favicon.ico" will
      work correctly both with client-side routing and a non-root public URL.
      Learn how to configure a non-root public URL by running `npm run build`.
    -->
    <link href="http://code.ionicframework.com/ionicons/2.0.1/css/ionicons.min.css" rel="stylesheet" type="text/css">
    <title>React App</title>
  </head>
  <body>
    <noscript>You need to enable JavaScript to run this app.</noscript>
    <div id="root"></div>
    <!--
      This HTML file is a template.
      If you open it directly in the browser, you will see an empty page.
      You can add webfonts, meta tags, or analytics to this file.
      The build step will place the bundled scripts into the <body> tag.
      To begin the development, run `npm start` or `yarn start`.
      To create a production bundle, use `npm run build` or `yarn build`.
    -->
  </body>
</html>
```

Album.js:

```
import React, {Component} from 'react';
import albumData from './../data/albums';
class Album extends Component {
  constructor(props){
    super(props);
    const album = albumData.find( album => {
       return album.slug === this.props.match.params.slug
     });
```

```
      this.state = {
        album: album,
        currentSong: album.songs[0],
        hoveredSongIndex: null,
        hoveredSong: null,
        isPlaying: false
      };

      this.audioElement = document.createElement('audio');
      this.audioElement.src = album.songs[0].audioSrc;
   }
   play(){
     this.audioElement.play();
     this.setState({isPlaying:true});
   }
   pause(){
     this.audioElement.pause();
     this.setState({isPlaying:false});
   }
   setSong(song){
     this.audioElement.src = song.audioSrc;
     this.setState({currentSong: song});
   }
   handleSongClick(song){
     const isSameSong = this.state.currentSong === song;
     if(this.state.isPlaying && isSameSong){
       this.pause();
     } else {
       if(!isSameSong) {
         this.setSong(song);
       }
       this.play();
     }
   }

   handleSongHover(song, index) {

     this.setState({hoveredSong: song})
     this.setState({hoveredSongIndex: index})
   };

   handleSongUnhover(song) {

     this.setState( {hoveredSongIndex: null} )

   };

   render() {
     return (
       <section className='album'>
       <section id="album-info">
         <img id="album-cover-art" src={this.state.album.al
 bumCover} alt={this.state.album.title}/>/>
          <div className="album-details">
          <h1 id="album-title">{this.state.album.title}</h1
 >
```

```
           <h2 className="artist">{this.state.album.artist
}</h2>
           <div id="release-info">{this.state.album.releas
eInfo}</div>
         </div>
       </section>
       <table align='center' id="song-list">
           <colgroup>
             <col id="song-number-column" />
             <col id="song-title-column" />
             <col id="song-duration-column" />
           </colgroup>
           <tbody>
           {
             this.state.album.songs.map( (song, index) =>
            <tr className='song'
            key={index}
            onClick={() => this.handleSongClick(song)}
            onMouseEnter={ ()=> this.handleSongHover(song,
index)}
            onMouseLeave={ ()=> this.handleSongUnhover(ind
ex)}
            >
               <td>{
                 (this.state.isPlaying && (this.state.curr
entSong === song)) ?
                  <span className='ion-pause'></span> :
                  (this.state.hoveredSongIndex === index)
?
                  <span className='ion-play'></span> : ind
ex+1
                }
               </td>
               <td>{song.title}</td>
               <td>{song.duration}</td>
            </tr>
          )
          }
           </tbody>
         </table>
      </section>
    );
  }
}
export default Album;
```

42. Create new component called `PlayerBar`. This will be located in the `src/components` folder and will hold the player controls for the application:

```
cd ~/bloc/bloc-jams-react/src/components
touch PlayerBar.js
```

43. Create basics of the new PlayerBar component within `PlayerBar.js`:

```
import React, {Component} from 'react';

class PlayerBar extends Component{
    render(){
        return(
        <section className='player-bar'>
            player bar goes here
        </section>
        );
    }
}

export default PlayerBar;
```

44. Import `PlayerBar` into the `Album` component:

```
import PlayerBar from './PlayerBar';
```

45. Render `PlayerBar` in the `Album` component below the table containing the song list:

```
import React, {Component} from 'react';
import albumData from './../data/albums';
import PlayerBar from './PlayerBar';
class Album extends Component {
  constructor(props){
    super(props);

    const album = albumData.find( album => {
       return album.slug === this.props.match.params.slug
     });
     this.state = {
       album: album,
       currentSong: album.songs[0],
       hoveredSongIndex: null,
       hoveredSong: null,
       isPlaying: false
     };

     this.audioElement = document.createElement('audio');
     this.audioElement.src = album.songs[0].audioSrc;
  }
  play(){
    this.audioElement.play();
    this.setState({isPlaying:true});
  }
  pause(){
    this.audioElement.pause();
    this.setState({isPlaying:false});
  }
  setSong(song){
    this.audioElement.src = song.audioSrc;
    this.setState({currentSong: song});
  }
  handleSongClick(song){
    const isSameSong = this.state.currentSong === song;
    if(this.state.isPlaying && isSameSong){
      this.pause();
    } else {
      if(!isSameSong) {
        this.setSong(song);
      }
      this.play();
    }
  }

  handleSongHover(song, index) {

    this.setState({hoveredSong: song})
    this.setState({hoveredSongIndex: index})
  };

  handleSongUnhover(song) {

    this.setState( {hoveredSongIndex: null} )

  };

  render() {
    return (
      <section className='album'>
      <section id="album-info">
        <img id="album-cover-art" src={this.state.album.al
bumCover} alt={this.state.album.title}/>/>
```

```
         <div className="album-details">
         <h1 id="album-title">{this.state.album.title}</h1
>
           <h2 className="artist">{this.state.album.artist
}</h2>
           <div id="release-info">{this.state.album.releas
eInfo}</div>
         </div>
       </section>
       <table align='center' id="song-list">
           <colgroup>
             <col id="song-number-column" />
             <col id="song-title-column" />
             <col id="song-duration-column" />
           </colgroup>
           <tbody>
           {
             this.state.album.songs.map( (song, index) =>
            <tr className='song'
            key={index}
            onClick={() => this.handleSongClick(song)}
            onMouseEnter={ ()=> this.handleSongHover(song,
index)}
            onMouseLeave={ ()=> this.handleSongUnhover(ind
ex)}
            >
               <td>{
                 (this.state.isPlaying && (this.state.curr
entSong === song)) ?
                  <span className='ion-pause'></span> :
                  (this.state.hoveredSongIndex === index)
?
                  <span className='ion-play'></span> : ind
ex+1
                }
               </td>
               <td>{song.title}</td>
               <td>{song.duration}</td>
            </tr>
          )
          }
           </tbody>
         </table>
         <PlayerBar />
      </section>
    );
  }
}
export default Album;
```

46. Add controls to `PlayerBar`: a previous button, play/pause button, next button, time control slider, and volume control slider:

```
import React, {Component} from 'react';

```

```
class PlayerBar extends Component{
    render(){
        return(
        <section className='player-bar'>
            <section id="buttons">
           <button id="previous">
             <span className="ion-skip-backward"></span>
           </button>
           <button id="play-pause">
             <span className="ion-play"></span>
             <span className="ion-pause"></span>
           </button>
           <button id="next">
             <span className="ion-skip-forward"></span>
           </button>
         </section>
         <section id="time-control">
           <div className="current-time">--:--</div>
           <input type="range" className="seek-bar" value=
"0" />
           <div className="total-time">--:--</div>
         </section>
         <section id="volume-control">
           <div className="icon ion-volume-low"></div>
           <input type="range" className="seek-bar" value=
"80" />
           <div className="icon ion-volume-high"></div>
         </section>
        </section>
        );
    }
}

export default PlayerBar;
```

47. Pass down state for `isPlaying` and `currentSong` to `PlayerBar` as props:

```
import React, {Component} from 'react';
import albumData from './../data/albums';
import PlayerBar from './PlayerBar';
class Album extends Component {
  constructor(props){
    super(props);
    const album = albumData.find( album => {
       return album.slug === this.props.match.params.slug
     });
     this.state = {
       album: album,
       currentSong: album.songs[0],
       hoveredSongIndex: null,
       hoveredSong: null,
       isPlaying: false
     };

     this.audioElement = document.createElement('audio');
```

```
      this.audioElement.src = album.songs[0].audioSrc;
   }
   play(){
     this.audioElement.play();
     this.setState({isPlaying:true});
   }
   pause(){
     this.audioElement.pause();
     this.setState({isPlaying:false});
   }
   setSong(song){
     this.audioElement.src = song.audioSrc;
     this.setState({currentSong: song});
   }
   handleSongClick(song){
     const isSameSong = this.state.currentSong === song;
     if(this.state.isPlaying && isSameSong){
       this.pause();
     } else {
       if(!isSameSong) {
         this.setSong(song);
       }
       this.play();
     }
   }

   handleSongHover(song, index) {

     this.setState({hoveredSong: song})
     this.setState({hoveredSongIndex: index})
   };

   handleSongUnhover(song) {

     this.setState( {hoveredSongIndex: null} )

   };

   render() {
     return (
       <section className='album'>
       <section id="album-info">
         <img id="album-cover-art" src={this.state.album.al
 bumCover} alt={this.state.album.title}/>/>
          <div className="album-details">
          <h1 id="album-title">{this.state.album.title}</h1
 >
            <h2 className="artist">{this.state.album.artist
 }</h2>
            <div id="release-info">{this.state.album.releas
 eInfo}</div>
          </div>
        </section>
        <table align='center' id="song-list">
            <colgroup>
              <col id="song-number-column" />
              <col id="song-title-column" />
              <col id="song-duration-column" />
            </colgroup>
            <tbody>
            {
              this.state.album.songs.map( (song, index) =>
             <tr className='song'
             key={index}
             onClick={() => this.handleSongClick(song)}
             onMouseEnter={ ()=> this.handleSongHover(song,
 index)}
             onMouseLeave={ ()=> this.handleSongUnhover(ind
 ex)}
             >
                <td>{
                  (this.state.isPlaying && (this.state.curr
 entSong === song)) ?
                   <span className='ion-pause'></span> :
                   (this.state.hoveredSongIndex === index)
 ?
                   <span className='ion-play'></span> : ind
 ex+1
                 }
                </td>
                <td>{song.title}</td>
                <td>{song.duration}</td>
             </tr>
           )
```

```
           }
            </tbody>
          </table>
          <PlayerBar
          isPlaying={this.state.isPlaying}
          currentSong={this.state.currentSong}
          />
       </section>
     );
   }
 }
export default Album;
```

48. Refactor play/pause icon to reflect state of play, using a ternary operator:

```
import React, {Component} from 'react';

class PlayerBar extends Component{
    render(){
        return(
        <section className='player-bar'>
            <section id="buttons">
           <button id="previous">
             <span className="ion-skip-backward"></span>
           </button>
           <button id="play-pause">
             <span
             className={this.props.isPlaying ? 'ion-pause'
 : 'ion-play'}>
             </span>
           </button>
           <button id="next">
             <span className="ion-skip-forward"></span>
           </button>
         </section>
         <section id="time-control">
           <div className="current-time">--:--</div>
           <input type="range" className="seek-bar" value=
"0" />
           <div className="total-time">--:--</div>
         </section>
         <section id="volume-control">
           <div className="icon ion-volume-low"></div>
           <input type="range" className="seek-bar" value=
"80" />
           <div className="icon ion-volume-high"></div>
         </section>
        </section>
        );
    }
}

export default PlayerBar;
```

49. Further adjust `PlayerBar` rendering in `Album` component so that `handleSongClick()` method can be passed to `PlayerBar`:

```
import React, {Component} from 'react';
import albumData from './../data/albums';
import PlayerBar from './PlayerBar';
class Album extends Component {
  constructor(props){
    super(props);
    const album = albumData.find( album => {
       return album.slug === this.props.match.params.slug
     });
     this.state = {
       album: album,
       currentSong: album.songs[0],
       hoveredSongIndex: null,
       hoveredSong: null,
       isPlaying: false
     };

     this.audioElement = document.createElement('audio');
     this.audioElement.src = album.songs[0].audioSrc;
  }
  play(){
    this.audioElement.play();
    this.setState({isPlaying:true});
  }
  pause(){
    this.audioElement.pause();
    this.setState({isPlaying:false});
  }
  setSong(song){
    this.audioElement.src = song.audioSrc;
    this.setState({currentSong: song});
  }
  handleSongClick(song){
    const isSameSong = this.state.currentSong === song;
    if(this.state.isPlaying && isSameSong){
      this.pause();
    } else {
      if(!isSameSong) {
        this.setSong(song);
      }
      this.play();
    }
  }

  handleSongHover(song, index) {

    this.setState({hoveredSong: song})
    this.setState({hoveredSongIndex: index})
  };

  handleSongUnhover(song) {

    this.setState( {hoveredSongIndex: null} )
  };

  render() {
    return (
      <section className='album'>
      <section id="album-info">
        <img id="album-cover-art" src={this.state.album.al
bumCover} alt={this.state.album.title}/>/>
          <div className="album-details">
          <h1 id="album-title">{this.state.album.title}</h1
>
            <h2 className="artist">{this.state.album.artist
}</h2>
            <div id="release-info">{this.state.album.releas
eInfo}</div>
          </div>
       </section>
       <table align='center' id="song-list">
            <colgroup>
              <col id="song-number-column" />
              <col id="song-title-column" />
              <col id="song-duration-column" />
            </colgroup>
            <tbody>
            {
              this.state.album.songs.map( (song, index) =>
             <tr className='song'
            key={index}
            onClick={() => this.handleSongClick(song)}
            onMouseEnter={ ()=> this.handleSongHover(song,
index)}
            onMouseLeave={ ()=> this.handleSongUnhover(ind
ex)}
            >
               <td>{
                 (this.state.isPlaying && (this.state.curr
entSong === song)) ?
                  <span className='ion-pause'></span> :
                  (this.state.hoveredSongIndex === index)
?
                  <span className='ion-play'></span> : ind
ex+1
                 }
               </td>
               <td>{song.title}</td>
               <td>{song.duration}</td>
            </tr>
          )
          }
           </tbody>
         </table>
         <PlayerBar
         isPlaying={this.state.isPlaying}
         currentSong={this.state.currentSong}
         handleSongClick={this.handleSongClick(this.state.
currentSong)}
         />
      </section>
    );
  }
}
export default Album;
```

50. add event listener to `PlayerBar` so that clicking on play-pause button calls `handleSongClick()`:

```
import React, {Component} from 'react';

class PlayerBar extends Component{
    render(){
        return(
        <section className='player-bar'>
            <section id="buttons">
           <button id="previous">
             <span className="ion-skip-backward"></span>
           </button>
           <button id="play-pause" onClick={this.props.han
dleSongClick}>
             <span
             className={this.props.isPlaying ? 'ion-pause'
 : 'ion-play'}>
             </span>
           </button>
           <button id="next">
             <span className="ion-skip-forward"></span>
           </button>
         </section>
         <section id="time-control">
           <div className="current-time">–:——</div>
           <input type="range" className="seek-bar" value=
"0" />
           <div className="total-time">–:——</div>
         </section>
         <section id="volume-control">
           <div className="icon ion-volume-low"></div>
           <input type="range" className="seek-bar" value=
"80" />
           <div className="icon ion-volume-high"></div>
         </section>
        </section>
        );
    }
}

export default PlayerBar;
```

51. Now add previous button functionality. Want to switch to previous song when the user clicks on the previous button. Write `handlePrevClick()` method, pass it down to `PlayerBar` as prop, and then add an event listener to the previous button to

trigger `handlePrevClick()`:

```
import React, {Component} from 'react';
import albumData from './../data/albums';
import PlayerBar from './PlayerBar';
class Album extends Component {
  constructor(props){
    super(props);
    const album = albumData.find( album => {
       return album.slug === this.props.match.params.slug
     });
     this.state = {
       album: album,
       currentSong: album.songs[0],
       hoveredSongIndex: null,
       hoveredSong: null,
       isPlaying: false
     };

     this.audioElement = document.createElement('audio');
     this.audioElement.src = album.songs[0].audioSrc;
  }
  play(){
    this.audioElement.play();
    this.setState({isPlaying:true});
  }
  pause(){
    this.audioElement.pause();
    this.setState({isPlaying:false});
  }
  setSong(song){
    this.audioElement.src = song.audioSrc;
    this.setState({currentSong: song});
  }
  handleSongClick(song){
    const isSameSong = this.state.currentSong === song;
    if(this.state.isPlaying && isSameSong){
      this.pause();
    } else {
      if(!isSameSong) {
        this.setSong(song);
      }
      this.play();
    }
  }

  handlePrevClick(){
      const currentIndex = this.state.album.songs.findInde
x(song => this.state.currentSong === song);
      const newIndex = Math.max(0, currentIndex - 1);
      const newSong = this.state.album.songs[newIndex];
      this.setSong(newSong);
      this.play();
  }

  handleSongHover(song, index) {

    this.setState({hoveredSong: song})
    this.setState({hoveredSongIndex: index})
  };

  handleSongUnhover(song) {

    this.setState( {hoveredSongIndex: null} )

  };

  render() {
    return (
      <section className='album'>
      <section id="album-info">
        <img id="album-cover-art" src={this.state.album.al
bumCover} alt={this.state.album.title}/>/>
         <div className="album-details">
         <h1 id="album-title">{this.state.album.title}</h1
>
           <h2 className="artist">{this.state.album.artist
}</h2>
           <div id="release-info">{this.state.album.releas
eInfo}</div>
         </div>
       </section>
       <table align='center' id="song-list">
           <colgroup>
```

```
              <col id="song-number-column" />
              <col id="song-title-column" />
              <col id="song-duration-column" />
            </colgroup>
            <tbody>
            {
              this.state.album.songs.map( (song, index) =>
             <tr className='song'
             key={index}
             onClick={() => this.handleSongClick(song)}
             onMouseEnter={ ()=> this.handleSongHover(song,
index)}
             onMouseLeave={ ()=> this.handleSongUnhover(ind
ex)}
             >
                <td>{
                  (this.state.isPlaying && (this.state.curr
entSong === song)) ?
                   <span className='ion-pause'></span> :
                   (this.state.hoveredSongIndex === index)
?
                   <span className='ion-play'></span> : ind
ex+1
                  }
                </td>
                <td>{song.title}</td>
                <td>{song.duration}</td>
             </tr>
```

```
           )
           }
            </tbody>
          </table>
          <PlayerBar
          isPlaying={this.state.isPlaying}
          currentSong={this.state.currentSong}
          handleSongClick={this.handleSongClick(this.state.
currentSong)}
          handlePrevClick={this.handlePrevClick()}
          />
       </section>
    );
  }
}
export default Album;
```

```
import React, {Component} from 'react';

class PlayerBar extends Component{
    render(){
        return(
        <section className='player-bar'>
            <section id="buttons">
           <button id="previous" onClick={this.props.handl
ePrevClick}>
             <span className="ion-skip-backward"></span>
           </button>
           <button id="play-pause" onClick={this.props.handleSongClick}>
             <span
             className={this.props.isPlaying ? 'ion-pause' : 'ion-play'}>
             </span>
           </button>
           <button id="next">
             <span className="ion-skip-forward"></span>
           </button>
         </section>
         <section id="time-control">
           <div className="current-time">—:——</div>
           <input type="range" className="seek-bar" value="0" />
           <div className="total-time">—:——</div>
         </section>
         <section id="volume-control">
           <div className="icon ion-volume-low"></div>
           <input type="range" className="seek-bar" value="80" />
           <div className="icon ion-volume-high"></div>
         </section>
        </section>
        );
    }
}
```

```
export default PlayerBar;
```

52. Similar to the previous button functionality, establish the next button functionality:

```
import React, {Component} from 'react';
import albumData from './../data/albums';
import PlayerBar from './PlayerBar';
class Album extends Component {
  constructor(props){
    super(props);
    const album = albumData.find( album => {
       return album.slug === this.props.match.params.slug
     });
     this.state = {
       album: album,
       currentSong: album.songs[0],
       hoveredSongIndex: null,
       hoveredSong: null,
       isPlaying: false
     };

     this.audioElement = document.createElement('audio');
     this.audioElement.src = album.songs[0].audioSrc;
  }
  play(){
    this.audioElement.play();
    this.setState({isPlaying:true});
  }
  pause(){
    this.audioElement.pause();
    this.setState({isPlaying:false});
  }
  setSong(song){
    this.audioElement.src = song.audioSrc;
    this.setState({currentSong: song});
  }
  handleSongClick(song){
    const isSameSong = this.state.currentSong === song;
    if(this.state.isPlaying && isSameSong){
      this.pause();
    } else {
      if(!isSameSong) {
        this.setSong(song);
      }
      this.play();
    }
  }

  handlePrevClick(){
      const currentIndex = this.state.album.songs.findInde
x(song => this.state.currentSong === song);
      const newIndex = Math.max(0, currentIndex - 1);
      const newSong = this.state.album.songs[newIndex];
      this.setSong(newSong);
      this.play();
  }

    handleNextClick(){
       const currentIndex = this.state.album.songs.findInd
ex(song => this.state.currentSong === song);
        const newIndex = Math.min(this.state.album.songs.l
ength-1, currentIndex+1);
        const newSong = this.state.album.songs[newIndex];
        this.setSong(newSong);
        this.play();
    }

  handleSongHover(song, index) {

    this.setState({hoveredSong: song})
    this.setState({hoveredSongIndex: index})
  };

  handleSongUnhover(song) {

    this.setState( {hoveredSongIndex: null} )

  };

  render() {
    return (
      <section className='album'>
      <section id="album-info">
        <img id="album-cover-art" src={this.state.album.al
bumCover} alt={this.state.album.title}/>/>
         <div className="album-details">
         <h1 id="album-title">{this.state.album.title}</h1
>
           <h2 className="artist">{this.state.album.artist
}</h2>
           <div id="release-info">{this.state.album.releas
eInfo}</div>
         </div>
       </section>
       <table align='center' id="song-list">
           <colgroup>
             <col id="song-number-column" />
             <col id="song-title-column" />
             <col id="song-duration-column" />
           </colgroup>
           <tbody>
           {
             this.state.album.songs.map( (song, index) =>
            <tr className='song'
            key={index}
            onClick={() => this.handleSongClick(song)}
            onMouseEnter={ ()=> this.handleSongHover(song,
index)}
            onMouseLeave={ ()=> this.handleSongUnhover(ind
ex)}
```

```
             >
                <td>{
                  (this.state.isPlaying && (this.state.curr
entSong === song)) ?
                   <span className='ion-pause'></span> :
                   (this.state.hoveredSongIndex === index)
?
                   <span className='ion-play'></span> : ind
ex+1
                 }
                </td>
                <td>{song.title}</td>
                <td>{song.duration}</td>
             </tr>
           )
           }
            </tbody>
          </table>
          <PlayerBar
          isPlaying={this.state.isPlaying}
          currentSong={this.state.currentSong}
          handleSongClick={this.handleSongClick(this.state.
currentSong)}
          handlePrevClick={this.handlePrevClick()}
          handleNextClick={this.handleNextClick()}
          />
       </section>
    );
```

```
  }
}
export default Album;
```

```
import React, {Component} from 'react';

class PlayerBar extends Component{
    render(){
        return(
        <section className='player-bar'>
            <section id="buttons">
           <button id="previous" onClick={this.props.handl
ePrevClick}>
             <span className="ion-skip-backward"></span>
           </button>
           <button id="play-pause" onClick={this.props.han
dleSongClick}>
             <span
             className={this.props.isPlaying ? 'ion-pause'
 : 'ion-play'}>
             </span>
           </button>
           <button id="next" onClick={this.props.handleNex
tClick}>
             <span className="ion-skip-forward"></span>
           </button>
         </section>
         <section id="time-control">
           <div className="current-time">–:––</div>
           <input type="range" className="seek-bar" value=
"0" />
           <div className="total-time">–:––</div>
         </section>
         <section id="volume-control">
           <div className="icon ion-volume-low"></div>
           <input type="range" className="seek-bar" value=
"80" />
           <div className="icon ion-volume-high"></div>
         </section>
        </section>
        );
    }
}

export default PlayerBar;
```

53. Set initial state for current song time and duration of song in `Album.js`. Then pass these states to `PlayerBar` component as props. Finally, render `currentTime` and `duration` in `PlayerBar`, providing a fallback value to the range value of 0 to prevent any undefined value from invalidated the range value:

```
import React, {Component} from 'react';
import albumData from './../data/albums';
import PlayerBar from './PlayerBar';
```

```
class Album extends Component {
  constructor(props){
    super(props);
    const album = albumData.find( album => {
       return album.slug === this.props.match.params.slug
     });
     this.state = {
       album: album,
       currentSong: album.songs[0],
       hoveredSongIndex: null,
       hoveredSong: null,
       currentTime: 0,
       duration: album.songs[0].duration,
       isPlaying:false
     };

     this.audioElement = document.createElement('audio');
     this.audioElement.src = album.songs[0].audioSrc;
  }
  play(){
    this.audioElement.play();
    this.setState({isPlaying:true});
  }
  pause(){
    this.audioElement.pause();
    this.setState({isPlaying:false});
  }
  setSong(song){
    this.audioElement.src = song.audioSrc;
    this.setState({currentSong: song});
  }
  handleSongClick(song){
    const isSameSong = this.state.currentSong === song;
    if(this.state.isPlaying && isSameSong){
      this.pause();
    } else {
      if(!isSameSong) {
        this.setSong(song);
      }
      this.play();
    }
  }

  handlePrevClick(){
      const currentIndex = this.state.album.songs.findInde
x(song => this.state.currentSong === song);
      const newIndex = Math.max(0, currentIndex - 1);
      const newSong = this.state.album.songs[newIndex];
      this.setSong(newSong);
      this.play();
  }

    handleNextClick(){
       const currentIndex = this.state.album.songs.findInd
ex(song => this.state.currentSong === song);
        const newIndex = Math.min(this.state.album.songs.l
ength-1, currentIndex+1);
        const newSong = this.state.album.songs[newIndex];
        this.setSong(newSong);
        this.play();
    }

  handleSongHover(song, index) {

    this.setState({hoveredSong: song})
    this.setState({hoveredSongIndex: index})
  };

  handleSongUnhover(song) {

    this.setState( {hoveredSongIndex: null} )

  };

  render() {
    return (
      <section className='album'>
      <section id="album-info">
        <img id="album-cover-art" src={this.state.album.al
bumCover} alt={this.state.album.title}/>/>
         <div className="album-details">
         <h1 id="album-title">{this.state.album.title}</h1
>
           <h2 className="artist">{this.state.album.artist
```

```
}</h2>
           <div id="release-info">{this.state.album.releas
eInfo}</div>
         </div>
       </section>
       <table align='center' id="song-list">
           <colgroup>
             <col id="song-number-column" />
             <col id="song-title-column" />
             <col id="song-duration-column" />
           </colgroup>
           <tbody>
           {
             this.state.album.songs.map( (song, index) =>
            <tr className='song'
            key={index}
            onClick={() => this.handleSongClick(song)}
            onMouseEnter={ ()=> this.handleSongHover(song,
index)}
            onMouseLeave={ ()=> this.handleSongUnhover(ind
ex)}
            >
               <td>{
                 (this.state.isPlaying && (this.state.curr
entSong === song)) ?
                  <span className='ion-pause'></span> :
                  (this.state.hoveredSongIndex === index)
?
```

```
                   <span className='ion-play'></span> : ind
ex+1
                 }
                </td>
                <td>{song.title}</td>
                <td>{song.duration}</td>
             </tr>
           )
           }
            </tbody>
          </table>
          <PlayerBar
          isPlaying={this.state.isPlaying}
          currentSong={this.state.currentSong}
          handleSongClick={this.handleSongClick(this.state.
currentSong)}
          handlePrevClick={this.handlePrevClick()}
          handleNextClick={this.handleNextClick()}
          currentTime={this.audioElement.currentTime}
          duration={this.audioElement.duration}
          />
       </section>
     );
  }
}
export default Album;
```

```
import React, {Component} from 'react';

class PlayerBar extends Component{
    render(){
        return(
        <section className='player-bar'>
            <section id="buttons">
           <button id="previous" onClick={this.props.handl
ePrevClick}>
             <span className="ion-skip-backward"></span>
           </button>
           <button id="play-pause" onClick={this.props.han
dleSongClick}>
             <span
             className={this.props.isPlaying ? 'ion-pause'
 : 'ion-play'}>
             </span>
           </button>
           <button id="next" onClick={this.props.handleNex
tClick}>
             <span className="ion-skip-forward"></span>
           </button>
         </section>
         <section id="time-control">
           <div className="current-time">{this.props.curre
ntTime}</div>
           <input
```

```
             type="range"
             className="seek-bar"
             value={(this.props.currentTime / this.props.duration) || 0}
             max="1"
             min="0"
             step="0.01"
           />
           <div className="total-time">{this.props.duration}</div>
         </section>
         <section id="volume-control">
           <div className="icon ion-volume-low"></div>
           <input type="range" className="seek-bar" value="80" />
           <div className="icon ion-volume-high"></div>
         </section>
        </section>
        );
    }
}

export default PlayerBar;
```

54. Go back to `Album` component and add the `componentDidMount()` lifecycle method. This should hold 2 event listeners for updating the time of the current song as well

as the current song’s duration. Update the state values for these 2 parameters within `Album`:

```
import React, {Component} from 'react';
import albumData from './../data/albums';
import PlayerBar from './PlayerBar';
class Album extends Component {
  constructor(props){
    super(props);
    const album = albumData.find( album => {
       return album.slug === this.props.match.params.slug
     });
     this.state = {
       album: album,
       currentSong: album.songs[0],
       hoveredSongIndex: null,
       hoveredSong: null,
       currentTime: 0,
       duration: album.songs[0].duration,
       isPlaying:false
     };

     this.audioElement = document.createElement('audio');
     this.audioElement.src = album.songs[0].audioSrc;
  }
  play(){
    this.audioElement.play();
    this.setState({isPlaying:true});
  }
  pause(){
    this.audioElement.pause();
    this.setState({isPlaying:false});
  }
  setSong(song){
    this.audioElement.src = song.audioSrc;
    this.setState({currentSong: song});
  }
  handleSongClick(song){
    const isSameSong = this.state.currentSong === song;
    if(this.state.isPlaying && isSameSong){
      this.pause();
    } else {
      if(!isSameSong) {
        this.setSong(song);
      }
      this.play();
    }
  }

  handlePrevClick(){
      const currentIndex = this.state.album.songs.findInde
x(song => this.state.currentSong === song);
      const newIndex = Math.max(0, currentIndex - 1);
      const newSong = this.state.album.songs[newIndex];
      this.setSong(newSong);
      this.play();
  }

    handleNextClick(){
       const currentIndex = this.state.album.songs.findInd
ex(song => this.state.currentSong === song);
        const newIndex = Math.min(this.state.album.songs.l
ength-1, currentIndex+1);
        const newSong = this.state.album.songs[newIndex];
        this.setSong(newSong);
        this.play();
    }

    componentDidMount(){
        this.audioElement.addEventListener('timeupdate', (
e)=> {
            this.setState({currentTime: this.audioElement.
currentTime});
        });
        this.audioElement.addEventListener('durationchange
', (e) => {
       this.setState({ duration: this.audioElement.duratio
n });
     });
    }

  handleSongHover(song, index) {

    this.setState({hoveredSong: song})
    this.setState({hoveredSongIndex: index})
  };

  handleSongUnhover(song) {

    this.setState( {hoveredSongIndex: null} )

  };

  render() {
    return (
      <section className='album'>
      <section id="album-info">
        <img id="album-cover-art" src={this.state.album.albumCover} alt={this.state.album.title}/>/>
         <div className="album-details">
         <h1 id="album-title">{this.state.album.title}</h1>
           <h2 className="artist">{this.state.album.artist}</h2>
           <div id="release-info">{this.state.album.releaseInfo}</div>
         </div>
       </section>
       <table align='center' id="song-list">
           <colgroup>
             <col id="song-number-column" />
             <col id="song-title-column" />
```

```
              <col id="song-duration-column" />
            </colgroup>
            <tbody>
            {
              this.state.album.songs.map( (song, index) =>
             <tr className='song'
             key={index}
             onClick={() => this.handleSongClick(song)}
             onMouseEnter={ ()=> this.handleSongHover(song,
index)}
             onMouseLeave={ ()=> this.handleSongUnhover(ind
ex)}
             >
                <td>{
                  (this.state.isPlaying && (this.state.curr
entSong === song)) ?
                   <span className='ion-pause'></span> :
                   (this.state.hoveredSongIndex === index)
?
                   <span className='ion-play'></span> : ind
ex+1
                 }
                </td>
                <td>{song.title}</td>
                <td>{song.duration}</td>
             </tr>
           )
           }
```

```
            </tbody>
          </table>
          <PlayerBar
          isPlaying={this.state.isPlaying}
          currentSong={this.state.currentSong}
          handleSongClick={this.handleSongClick(this.state.
currentSong)}
          handlePrevClick={this.handlePrevClick()}
          handleNextClick={this.handleNextClick()}
          currentTime={this.audioElement.currentTime}
          duration={this.audioElement.duration}
          />
      </section>
    );
  }
}
export default Album;
```

55. Refactor `componentDidMount` in `Album` component so that we remove the event listeners when the component is unmounted to prevent errors. Also terminate playback if user leaves album page by adding a `componentWillUnmount()` method. Finally, refactor code so that callbacks are being stored on the `this` keyword:

```
import React, {Component} from 'react';
import albumData from './../data/albums';
import PlayerBar from './PlayerBar';
```

```
class Album extends Component {
  constructor(props){
    super(props);
    const album = albumData.find( album => {
       return album.slug === this.props.match.params.slug
     });
     this.state = {
       album: album,
       currentSong: album.songs[0],
       hoveredSongIndex: null,
       hoveredSong: null,
       currentTime: 0,
       duration: album.songs[0].duration,
       isPlaying:false
     };

     this.audioElement = document.createElement('audio');
     this.audioElement.src = album.songs[0].audioSrc;
  }
  play(){
    this.audioElement.play();
    this.setState({isPlaying:true});
  }
  pause(){
    this.audioElement.pause();
    this.setState({isPlaying:false});
  }
  setSong(song){
    this.audioElement.src = song.audioSrc;
    this.setState({currentSong: song});
  }
  handleSongClick(song){
    const isSameSong = this.state.currentSong === song;
    if(this.state.isPlaying && isSameSong){
      this.pause();
    } else {
      if(!isSameSong) {
        this.setSong(song);
      }
      this.play();
    }
  }

  handlePrevClick(){
      const currentIndex = this.state.album.songs.findInde
x(song => this.state.currentSong === song);
      const newIndex = Math.max(0, currentIndex - 1);
      const newSong = this.state.album.songs[newIndex];
      this.setSong(newSong);
      this.play();
  }

    handleNextClick(){
       const currentIndex = this.state.album.songs.findInd
ex(song => this.state.currentSong === song);
        const newIndex = Math.min(this.state.album.songs.l
ength-1, currentIndex+1);
        const newSong = this.state.album.songs[newIndex];
        this.setSong(newSong);
        this.play();
    }

    componentDidMount(){
        this.eventListeners = {
       timeupdate: e => {
         this.setState({ currentTime: this.audioElement.cu
rrentTime });
       },
       durationchange: e => {
         this.setState({ duration: this.audioElement.durat
ion });
       }
     };
     this.audioElement.addEventListener('timeupdate', this
.eventListeners.timeupdate);
     this.audioElement.addEventListener('durationchange',
this.eventListeners.durationchange);
   }

   componentWillUnmount() {
     this.audioElement.src = null;
     this.audioElement.removeEventListener('timeupdate', t
his.eventListeners.timeupdate);
     this.audioElement.removeEventListener('durationchange
', this.eventListeners.durationchange);
   }

  handleSongHover(song, index) {

    this.setState({hoveredSong: song})
    this.setState({hoveredSongIndex: index})
  };

  handleSongUnhover(song) {

    this.setState( {hoveredSongIndex: null} )

  };

  render() {
    return (
      <section className='album'>
      <section id="album-info">
        <img id="album-cover-art" src={this.state.album.al
bumCover} alt={this.state.album.title}/>/>
         <div className="album-details">
         <h1 id="album-title">{this.state.album.title}</h1
>
           <h2 className="artist">{this.state.album.artist
}</h2>
           <div id="release-info">{this.state.album.releas
eInfo}</div>
```

```
         </div>
       </section>
       <table align='center' id="song-list">
           <colgroup>
             <col id="song-number-column" />
             <col id="song-title-column" />
             <col id="song-duration-column" />
           </colgroup>
           <tbody>
           {
             this.state.album.songs.map( (song, index) =>
            <tr className='song'
            key={index}
            onClick={() => this.handleSongClick(song)}
            onMouseEnter={ ()=> this.handleSongHover(song, index)}
            onMouseLeave={ ()=> this.handleSongUnhover(index)}
            >
               <td>{
                 (this.state.isPlaying && (this.state.currentSong === song)) ?
                  <span className='ion-pause'></span> :
                  (this.state.hoveredSongIndex === index) ?
                  <span className='ion-play'></span> : index+1
                }
                </td>
                <td>{song.title}</td>
                <td>{song.duration}</td>
             </tr>
           )
           }
            </tbody>
          </table>
          <PlayerBar
          isPlaying={this.state.isPlaying}
          currentSong={this.state.currentSong}
          handleSongClick={this.handleSongClick(this.state.
currentSong)}
          handlePrevClick={this.handlePrevClick()}
          handleNextClick={this.handleNextClick()}
          currentTime={this.audioElement.currentTime}
          duration={this.audioElement.duration}
          />
       </section>
     );
  }
}
export default Album;
```

56. Now add code so that time control slider responds to user input. Create `handleTimeChange()` methodin `Album`, pass this method down to `PlayerBar` as props, and then add an

`onChange` event listender to the time control slider.

```
import React, {Component} from 'react';
import albumData from './../data/albums';
import PlayerBar from './PlayerBar';
class Album extends Component {
  constructor(props){
    super(props);
    const album = albumData.find( album => {
       return album.slug === this.props.match.params.slug
     });
     this.state = {
       album: album,
       currentSong: album.songs[0],
       hoveredSongIndex: null,
       hoveredSong: null,
       currentTime: 0,
       duration: album.songs[0].duration,
       isPlaying:false
     };

     this.audioElement = document.createElement('audio');
     this.audioElement.src = album.songs[0].audioSrc;
  }
  play(){
    this.audioElement.play();
    this.setState({isPlaying:true});
  }
  pause(){
    this.audioElement.pause();
    this.setState({isPlaying:false});
  }
  setSong(song){
    this.audioElement.src = song.audioSrc;
    this.setState({currentSong: song});
  }
  handleSongClick(song){
    const isSameSong = this.state.currentSong === song;
    if(this.state.isPlaying && isSameSong){
      this.pause();
    } else {
      if(!isSameSong) {
        this.setSong(song);
      }
      this.play();
    }
  }

  handlePrevClick(){
      const currentIndex = this.state.album.songs.findInde
x(song => this.state.currentSong === song);
      const newIndex = Math.max(0, currentIndex - 1);
      const newSong = this.state.album.songs[newIndex];
      this.setSong(newSong);
      this.play();
  }
```

```

    handleNextClick(){
       const currentIndex = this.state.album.songs.findInd
ex(song => this.state.currentSong === song);
        const newIndex = Math.min(this.state.album.songs.l
ength-1, currentIndex+1);
        const newSong = this.state.album.songs[newIndex];
        this.setSong(newSong);
        this.play();
    }

    componentDidMount(){
        this.eventListeners = {
       timeupdate: e => {
         this.setState({ currentTime: this.audioElement.cu
rrentTime });
       },
       durationchange: e => {
         this.setState({ duration: this.audioElement.durat
ion });
       }
     };
     this.audioElement.addEventListener('timeupdate', this
.eventListeners.timeupdate);
     this.audioElement.addEventListener('durationchange',
this.eventListeners.durationchange);
   }

```

```
    componentWillUnmount() {
      this.audioElement.src = null;
      this.audioElement.removeEventListener('timeupdate', t
his.eventListeners.timeupdate);
      this.audioElement.removeEventListener('durationchange
', this.eventListeners.durationchange);
    }

    handleTimeChange(e){
        const newTime = this.audioElement.duration * e.targ
et.value;
        this.audioElement.currentTime = newTime;
        this.setState({currentTime: newTime});
    }

   handleSongHover(song, index) {

     this.setState({hoveredSong: song})
     this.setState({hoveredSongIndex: index})
   };

   handleSongUnhover(song) {

     this.setState( {hoveredSongIndex: null} )

   };

   render() {
     return (
       <section className='album'>
       <section id="album-info">
         <img id="album-cover-art" src={this.state.album.al
 bumCover} alt={this.state.album.title}/>/>
          <div className="album-details">
          <h1 id="album-title">{this.state.album.title}</h1
 >
            <h2 className="artist">{this.state.album.artist
 }</h2>
            <div id="release-info">{this.state.album.releas
 eInfo}</div>
          </div>
        </section>
        <table align='center' id="song-list">
            <colgroup>
              <col id="song-number-column" />
              <col id="song-title-column" />
              <col id="song-duration-column" />
            </colgroup>
            <tbody>
            {
              this.state.album.songs.map( (song, index) =>
             <tr className='song'
             key={index}
             onClick={() => this.handleSongClick(song)}
             onMouseEnter={ ()=> this.handleSongHover(song,
 index)}
```

```
            onMouseLeave={ ()=> this.handleSongUnhover(ind
ex)}
            >
               <td>{
                 (this.state.isPlaying && (this.state.curr
entSong === song)) ?
                  <span className='ion-pause'></span> :
                  (this.state.hoveredSongIndex === index)
?
                  <span className='ion-play'></span> : ind
ex+1
                 }
               </td>
               <td>{song.title}</td>
               <td>{song.duration}</td>
            </tr>
          )
          }
           </tbody>
         </table>
         <PlayerBar
         isPlaying={this.state.isPlaying}
         currentSong={this.state.currentSong}
         handleSongClick={this.handleSongClick(this.state.
currentSong)}
         handlePrevClick={this.handlePrevClick()}
         handleNextClick={this.handleNextClick()}
         currentTime={this.audioElement.currentTime}
```

```
          duration={this.audioElement.duration}
          handleTimeChange={(e)=> this.handleTimeChange(e)}
          />
       </section>
     );
   }
}
export default Album;
```

```
import React, {Component} from 'react';

class PlayerBar extends Component{
    render(){
        return(
        <section className='player-bar'>
            <section id="buttons">
           <button id="previous" onClick={this.props.handl
ePrevClick}>
             <span className="ion-skip-backward"></span>
           </button>
           <button id="play-pause" onClick={this.props.han
dleSongClick}>
             <span
             className={this.props.isPlaying ? 'ion-pause'
 : 'ion-play'}>
             </span>
           </button>
           <button id="next" onClick={this.props.handleNextClick}>
             <span className="ion-skip-forward"></span>
           </button>
         </section>
         <section id="time-control">
           <div className="current-time">{this.props.currentTime}</div>
           <input
             type="range"
             className="seek-bar"
             value={(this.props.currentTime / this.props.duration) || 0}
             max="1"
             min="0"
             step="0.01"
             onChange={this.props.handleTimeChange}
           />
           <div className="total-time">{this.props.duration}</div>
         </section>
         <section id="volume-control">
           <div className="icon ion-volume-low"></div>
           <input type="range" className="seek-bar" value="80" />
           <div className="icon ion-volume-high"></div>
         </section>
        </section>
        );
    }
}

export default PlayerBar;
```

57. Now add volume slider functionality:

```
import React, {Component} from 'react';
import albumData from './../data/albums';
import PlayerBar from './PlayerBar';
class Album extends Component {
  constructor(props){
    super(props);
    const album = albumData.find( album => {
       return album.slug === this.props.match.params.slug
     });
     this.state = {
       album: album,
       currentSong: album.songs[0],
       hoveredSongIndex: null,
       hoveredSong: null,
       currentTime: 0,
       duration: album.songs[0].duration,
       isPlaying:false,
       currentVolume:0
     };

```

```
      this.audioElement = document.createElement('audio');
      this.audioElement.src = album.songs[0].audioSrc;
   }
   play(){
     this.audioElement.play();
     this.setState({isPlaying:true});
   }
   pause(){
     this.audioElement.pause();
     this.setState({isPlaying:false});
   }
   setSong(song){
     this.audioElement.src = song.audioSrc;
     this.setState({currentSong: song});
   }
   handleSongClick(song){
     const isSameSong = this.state.currentSong === song;
     if(this.state.isPlaying && isSameSong){
       this.pause();
     } else {
       if(!isSameSong) {
         this.setSong(song);
       }
       this.play();
     }
   }

   handlePrevClick(){
       const currentIndex = this.state.album.songs.findInde
 x(song => this.state.currentSong === song);
       const newIndex = Math.max(0, currentIndex - 1);
       const newSong = this.state.album.songs[newIndex];
       this.setSong(newSong);
       this.play();
   }

     handleNextClick(){
        const currentIndex = this.state.album.songs.findInd
 ex(song => this.state.currentSong === song);
         const newIndex = Math.min(this.state.album.songs.l
 ength-1, currentIndex+1);
         const newSong = this.state.album.songs[newIndex];
         this.setSong(newSong);
         this.play();
     }

     componentDidMount(){
         this.eventListeners = {
        timeupdate: e => {
          this.setState({ currentTime: this.audioElement.cu
 rrentTime });
        },
        durationchange: e => {
          this.setState({ duration: this.audioElement.durat
 ion });
        }
      };
      this.audioElement.addEventListener('timeupdate', this
.eventListeners.timeupdate);
      this.audioElement.addEventListener('durationchange',
this.eventListeners.durationchange);
    }

    componentWillUnmount() {
      this.audioElement.src = null;
      this.audioElement.removeEventListener('timeupdate', t
his.eventListeners.timeupdate);
      this.audioElement.removeEventListener('durationchange
', this.eventListeners.durationchange);
    }

    handleTimeChange(e){
        const newTime = this.audioElement.duration * e.targ
et.value;
        this.audioElement.currentTime = newTime;
        this.setState({currentTime: newTime});
    }

    handleVolumeChange(e) {
     const newVolume = e.target.value;
     this.audioElement.volume = newVolume;
    this.setState({currentVolume: newVolume});
   }

```

```
  handleSongHover(song, index) {

    this.setState({hoveredSong: song})
    this.setState({hoveredSongIndex: index})
  };

  handleSongUnhover(song) {

    this.setState( {hoveredSongIndex: null} )

  };

  render() {
    return (
      <section className='album'>
      <section id="album-info">
        <img id="album-cover-art" src={this.state.album.al
bumCover} alt={this.state.album.title}/>/>
         <div className="album-details">
         <h1 id="album-title">{this.state.album.title}</h1
>
           <h2 className="artist">{this.state.album.artist
}</h2>
           <div id="release-info">{this.state.album.releas
eInfo}</div>
         </div>
        </section>
        <table align='center' id="song-list">
```

```
           <colgroup>
             <col id="song-number-column" />
             <col id="song-title-column" />
             <col id="song-duration-column" />
           </colgroup>
           <tbody>
           {
             this.state.album.songs.map( (song, index) =>
            <tr className='song'
            key={index}
            onClick={() => this.handleSongClick(song)}
            onMouseEnter={ ()=> this.handleSongHover(song, index)}
            onMouseLeave={ ()=> this.handleSongUnhover(index)}
            >
               <td>{
                 (this.state.isPlaying && (this.state.currentSong === song)) ?
                  <span className='ion-pause'></span> :
                  (this.state.hoveredSongIndex === index) ?
                  <span className='ion-play'></span> : index+1
                }
               </td>
               <td>{song.title}</td>
               <td>{song.duration}</td>
             </tr>
           )
           }
            </tbody>
          </table>
          <PlayerBar
          isPlaying={this.state.isPlaying}
          currentSong={this.state.currentSong}
          handleSongClick={this.handleSongClick(this.state.
currentSong)}
          handlePrevClick={this.handlePrevClick()}
          handleNextClick={this.handleNextClick()}
          currentTime={this.audioElement.currentTime}
          duration={this.audioElement.duration}
          handleTimeChange={(e)=> this.handleTimeChange(e)}
          currentVolume={this.audioElement.currentVolume}
          handleVolumeChange={(e)=> this.handleVolumeChange
(e)}

          />
       </section>
     );
  }
}
export default Album;
```

```
import React, {Component} from 'react';

```

```
class PlayerBar extends Component{
    render(){
        return(
        <section className='player-bar'>
            <section id="buttons">
           <button id="previous" onClick={this.props.handl
ePrevClick}>
             <span className="ion-skip-backward"></span>
           </button>
           <button id="play-pause" onClick={this.props.han
dleSongClick}>
             <span
             className={this.props.isPlaying ? 'ion-pause'
 : 'ion-play'}>
             </span>
           </button>
           <button id="next" onClick={this.props.handleNex
tClick}>
             <span className="ion-skip-forward"></span>
           </button>
         </section>
         <section id="time-control">
           <div className="current-time">{this.props.curre
ntTime}</div>
           <input
             type="range"
             className="seek-bar"
             value={(this.props.currentTime / this.props.d
uration) || 0}
             max="1"
             min="0"
             step="0.01"
             onChange={this.props.handleTimeChange}
           />
           <div className="total-time">{this.props.duratio
n}</div>
         </section>
         <section id="volume-control">
           <div className="icon ion-volume-low"></div>
           <input
           type="range"
           className="seek-bar"
           value={this.props.currentVolume}
           max="1"
           min="0"
           step=".01"
           onChange={this.props.handleVolumeChange}
           />
           <div className="icon ion-volume-high"></div>
         </section>
        </section>
        );
    }
}
```

```
export default PlayerBar;
```

58. Change the time format so that, rather than displayed in seconds, time is displayed in a `M:SS` format wherever time is displayed in the application. Give a fallback value of `"-:--"`

```
import React, {Component} from 'react';
import albumData from './../data/albums';
import PlayerBar from './PlayerBar';
class Album extends Component {
  constructor(props){
    super(props);
    const album = albumData.find( album => {
       return album.slug === this.props.match.params.slug
     });
     this.state = {
       album: album,
       currentSong: album.songs[0],
       hoveredSongIndex: null,
       hoveredSong: null,
       currentTime: 0,
       duration: album.songs[0].duration,
       isPlaying:false,
       currentVolume:0
     };


     this.audioElement = document.createElement('audio');
     this.audioElement.src = album.songs[0].audioSrc;
  }
  play(){
    this.audioElement.play();
    this.setState({isPlaying:true});
  }
  pause(){
    this.audioElement.pause();
    this.setState({isPlaying:false});
  }
  setSong(song){
    this.audioElement.src = song.audioSrc;
    this.setState({currentSong: song});
  }
  handleSongClick(song){
    const isSameSong = this.state.currentSong === song;
    if(this.state.isPlaying && isSameSong){
      this.pause();
    } else {
      if(!isSameSong) {
        this.setSong(song);
      }
      this.play();
    }
  }

  handlePrevClick(){
      const currentIndex = this.state.album.songs.findInde
x(song => this.state.currentSong === song);
```

```
      const newIndex = Math.max(0, currentIndex - 1);
      const newSong = this.state.album.songs[newIndex];
      this.setSong(newSong);
      this.play();
  }

    handleNextClick(){
       const currentIndex = this.state.album.songs.findInd
ex(song => this.state.currentSong === song);
        const newIndex = Math.min(this.state.album.songs.l
ength-1, currentIndex+1);
        const newSong = this.state.album.songs[newIndex];
        this.setSong(newSong);
        this.play();
    }

    componentDidMount(){
        this.eventListeners = {
       timeupdate: e => {
         this.setState({ currentTime: this.audioElement.cu
rrentTime });
       },
       durationchange: e => {
         this.setState({ duration: this.audioElement.durat
ion });
       }
      };
      this.audioElement.addEventListener('timeupdate', this
.eventListeners.timeupdate);
     this.audioElement.addEventListener('durationchange',
this.eventListeners.durationchange);
   }

   componentWillUnmount() {
     this.audioElement.src = null;
     this.audioElement.removeEventListener('timeupdate', t
his.eventListeners.timeupdate);
     this.audioElement.removeEventListener('durationchange
', this.eventListeners.durationchange);
   }

   handleTimeChange(e){
       const newTime = this.audioElement.duration * e.targ
et.value;
       this.audioElement.currentTime = newTime;
       this.setState({currentTime: newTime});
   }

   handleVolumeChange(e) {
    const newVolume = e.target.value;
    this.audioElement.volume = newVolume;
   this.setState({currentVolume: newVolume});
  }

  formatTime=(timeInSeconds) => {
      //accept time in seconds as parameter and convert it
```

```
into string M:SS, and with default value of -:--
   var minutes = Math.floor(timeInSeconds/60);
   var seconds = Math.round(timeInSeconds - minutes*60);
   if (seconds < 10) {
      return minutes + ":0" + seconds;
    }
    else {
      return minutes + ":" + seconds
    }
 }

 handleSongHover(song, index) {

   this.setState({hoveredSong: song})
   this.setState({hoveredSongIndex: index})
 };

 handleSongUnhover(song) {

   this.setState( {hoveredSongIndex: null} )

 };

 render() {
   return (
     <section className='album'>
     <section id="album-info">
       <img id="album-cover-art" src={this.state.album.al
```

```
bumCover} alt={this.state.album.title}/>/>
         <div className="album-details">
         <h1 id="album-title">{this.state.album.title}</h1
>
           <h2 className="artist">{this.state.album.artist
}</h2>
           <div id="release-info">{this.state.album.releas
eInfo}</div>
         </div>
       </section>
       <table align='center' id="song-list">
           <colgroup>
             <col id="song-number-column" />
             <col id="song-title-column" />
             <col id="song-duration-column" />
           </colgroup>
           <tbody>
           {
             this.state.album.songs.map( (song, index) =>
            <tr className='song'
            key={index}
            onClick={() => this.handleSongClick(song)}
            onMouseEnter={ ()=> this.handleSongHover(song,
index)}
            onMouseLeave={ ()=> this.handleSongUnhover(ind
ex)}
            >
               <td>{
                 (this.state.isPlaying && (this.state.curr
entSong === song)) ?
                  <span className='ion-pause'></span> :
                  (this.state.hoveredSongIndex === index)
?
                  <span className='ion-play'></span> : ind
ex+1
                }
               </td>
               <td>{song.title}</td>
               <td>{song.duration}</td>
            </tr>
          )
          }
           </tbody>
         </table>
         <PlayerBar
         isPlaying={this.state.isPlaying}
         currentSong={this.state.currentSong}
         handleSongClick={this.handleSongClick(this.state.
currentSong)}
         handlePrevClick={this.handlePrevClick()}
         handleNextClick={this.handleNextClick()}
         currentTime={this.audioElement.currentTime}
         duration={this.audioElement.duration}
         handleTimeChange={(e)=> this.handleTimeChange(e)}
         currentVolume={this.audioElement.currentVolume}
         handleVolumeChange={(e)=> this.handleVolumeChange
```

```
(e)}
         formatTime={ (timeInSeconds)=> this.formatTime(ti
meInSeconds) }

         />
      </section>
    );
  }
}
export default Album;
```

```
import React, {Component} from 'react';

class PlayerBar extends Component{
    render(){
        return(
        <section className='player-bar'>
            <section id="buttons">
           <button id="previous" onClick={this.props.handl
ePrevClick}>
             <span className="ion-skip-backward"></span>
           </button>
           <button id="play-pause" onClick={this.props.han
dleSongClick}>
             <span
             className={this.props.isPlaying ? 'ion-pause'
 : 'ion-play'}>

```

```
             </span>
           </button>
           <button id="next" onClick={this.props.handleNextClick}>
             <span className="ion-skip-forward"></span>
           </button>
         </section>
         <section id="time-control">
           <div className="current-time">{this.props..formatTime(this.props.currentTime)}</div>
           <input
             type="range"
             className="seek-bar"
             value={(this.props.currentTime / this.props.duration) || 0}
             max="1"
             min="0"
             step="0.01"
             onChange={this.props.handleTimeChange}
           />
           <div className="total-time">{this.props.formatTime(this.props.duration)}</div>
         </section>
         <section id="volume-control">
           <div className="icon ion-volume-low"></div>
           <input
           type="range"
           className="seek-bar"
```

```
            value={this.props.currentVolume}
            max="1"
            min="0"
            step=".01"
            onChange={this.props.handleVolumeChange}
            />
            <div className="icon ion-volume-high"></div>
          </section>
        </section>
        );
    }
}

export default PlayerBar;
```

59. Add styling to application as desired.