UNIVERZITET U BEOGRADU
FAKULTET ORGANIZACIONIH NAUKA
Laboratorija za elektronsko poslovanje

Dokumentacija- React JS:

Student:	Broj indeksa:
Uroš Dedović	**163/17**

Beograd - 2021.

Sadržaj

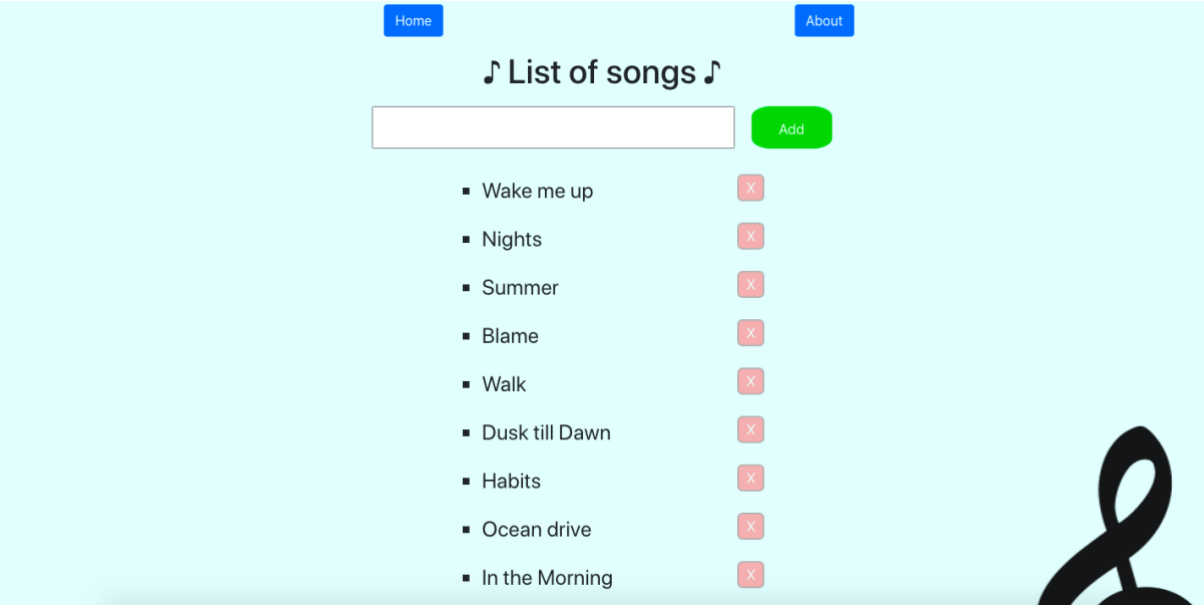
Uvod

Tema ovog projektnog zadatka je omogućavanje korisniku dodavanje i brisanje pesama na listu pesama. Lista pesama se čuva u local storage-u browser-a. Takođe, na stranici About se nalazi se kratak opis aplikacije i analogni sat.

Projekat je realizovan korišćenjem React Js-a.

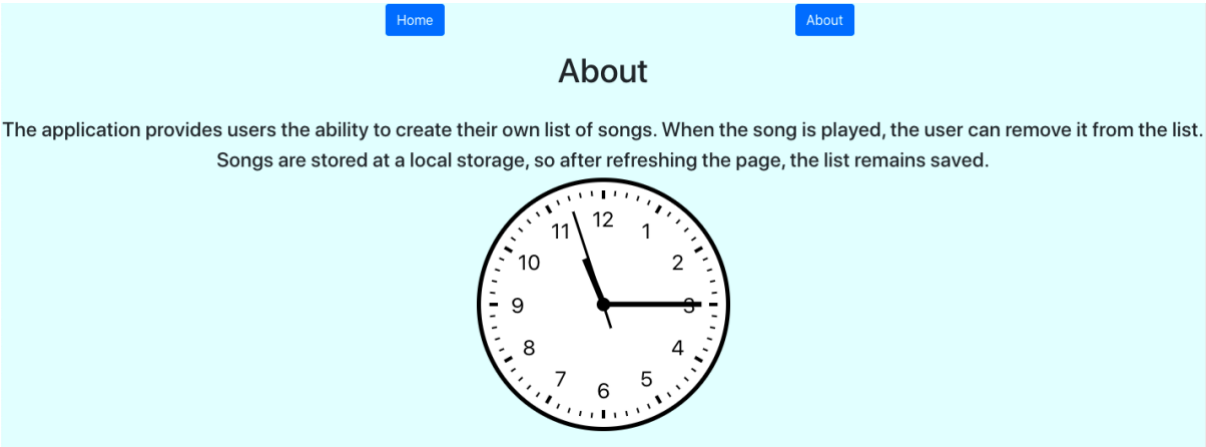
Opis funkcionalnosti sajta

Kada se pokrene aplikacija korišćenjem komande „npm start“ pokreće se glavna „Home“ stranica aplikacije. Na njoj se nalaze dva dugmeta za prebacivanje sa jedne na drugu stranicu. Takođe, nalazi se input polje u koje se unosi željena pesma. Na dugme „Add“ se pesma dodaj u listu pesama koja se čuva u local storage-u čime smo omogućili čuvanje liste i nakon gašenja sajta. Ispod dela forme nalazi se lista pesama i dugme kojim se označena pesma briše iz liste.



Slika 1 – izgled početne stranice

Pritiskom na dugme „About“ korisnik se odvodi na stranicu „About“ na kojoj se nalazi kratak opis aplikacije i analogni sat. Analogni sat je dodat korišćenjem javascript widget-a.



Slika 2 – izgled About stranice

Opisivanje funkcionalnosti kroz JS

Home page

Početna (Home) stranica se sastoji od div polja koji sadrži sledeće komponente:

```
return(
    <div style={{backgroundColor:'#e6ffff', height:'100vh'}}>
        <h1>♪ List of songs ♪</h1>
        <SongsForm addSong={addSong} />
        <SongList songs={songs} removeSong={removeSong}/>
    </div>
);
```

Vidimo da postoje dva naša tag-a, to su SongsForm i SongsList koji su zaduženi za deo forme(unos nove pesme i dugme za dodavanja) kao i za prikaz liste svih pesama.

SongsForm

U SongsForm komponenti nam se nalazi sledeće:

```
function SongsForm({addSong}){

    const [song, setSong] = useState({
        id: "",
        name: ""
    });
```

```
  function handleSongInputChange(e){
      setSong({...song, name: e.target.value})
  }

  function handleSubmit(e) {
      e.preventDefault();
      if(song.name.trim()){
          addSong({...song, id:uuidv4()});
          setSong({...song, name:""});
      }
  }

  return(
    <form class="main-form" onSubmit={handleSubmit} >
        <input autoComplete="off" onChange={handleSongInputChange} value={song.name} name="name"
type="text"/>
        <button class="addbtn" type="submit">Add</button>
    </form>
  );
}

export default SongsForm;
```

Vidimo da se sastoji iz input polja za unos pesme i dugmeta za dodavanje pesme u listu. Takođe, imamo funkciju handleSubmit koja nam ostvaruje mogućnost dodavanja u listu.

SongList

Što se tiče SongList komponente, ona se sastoji od:

```
function SongList({ songs, removeSong }) {
 let data = Array.from(songs);
 return (
  <div class ="pozadina" style={{backgroundColor:'#e6ffff', height:'100vh'}}>
    <ul class="list">
      {data.map((song) => (
        <Song key={song.id} song={song} removeSong={removeSong} />
      ))}
    </ul>
  </div>
```

```
  );
}


export default SongList;
```

Ona prima listu pesama koju treba da ispiše i onda preko lambda izraza prolazi kroz listu pesama i za svaku pojedinačnu pesmu kreira komponentu Song i dodeljuje joj potrebne atribute.

Song

Song komponenta predstavlja jedan red u listi pesama i sadrži naziv pesme i dugme „X“ kojim se ta pesma izbacuje iz lista:

```
function Song({song, removeSong}) {

   function handleRemoveClick(){
      removeSong(song.id);
   }

   return(
      <div class = "container">
         <li class="item">{song.name}</li>
         <button class="Xbtn" onClick={handleRemoveClick}>X</button>
      </div>
   );
}

export default Song;
```

Meni

Što se menija tiče, imamo kompomentu Meni koja obrađuje dva dugmeta za dve stranice:

```
function Meni() {
  return (
    <div>
      <ul
        style={{
          display: "flex",
          justifyContent: "space-evenly",
          paddingTop: "5px",
          backgroundColor:'#e6ffff'
        }}
      >
        <Button variant="primary">
          <Link to="/" style={{ textDecoration: "none", color: "white" }}>
            Home
          </Link>
```

```
      </Button>{" "}
      <Button variant="primary">
        <Link to="/about" style={{ textDecoration: "none", color: "white" }}>
          About
        </Link>
      </Button>{" "}
    </ul>
  </div>
 );
}

export default Meni;
```

Vidimo da korišćenjem Link komponente označavamo stranicu koja će se pokrenuti pritiskom na dugme.

About page

Na about stranici se nalazi kratak opis aplikacije i analogni sat:

```
return (
  <div style={{backgroundColor:'#e6ffff', height:'100vh'}}>
    <h1>About</h1>
    <br></br>
    <h4>
      The application provides users the ability to create their own list of
      songs. When the song is played, the user can remove it from the list.
    </h4>
    <h4>
     {" "}
      Songs are stored at a local storage, so after refreshing the page, the
      list remains saved.
    </h4>
    <div style={{ display: "flex", justifyContent: "center" }}>
      <AnalogueClock
        {...clockOptions}
        style={{ display: "flex", justifyContent: "center" }}
      />
    </div>
  </div>
 );
```

```
}

export default About;
```

Analogni sat je dodat preko komponente AnalogueClock.

App

Na kraju, komponenta koja pokreće sve jeste App.js komponenta u kojoj se nalazi Meni komponenta i Route komponenta koja će prikazivati sadržaj u zavisnosti rute koja je ukucana:

```
return (
    <div className="App" style={{backgroundColor:'#e6ffff'}}>
      <Meni />
      <Route exact path="/" component={Home} />
      <Route exact path="/about" component={About} />

    </div>
  );
}
```

Za svaku od komponenti je kreiran poseban .css fajl u kome je izvršeno stilizovanje.