// Module Code: IT8906

// Module Name: Web and Mobile Development Using Frameworks

// Acad Year/Semester: 2022/2023 Sem 2

// Assignment Title: CA1 - React Core Concepts

// Student Name: Ong Wei Chuan

// Student No: P7461072

// Filename: MovieMgmt.jsx

// This file contains the root component App()

// which will render on the DOM node 'root'.

// It will call other children components.

// References: Information on movies, icons

// 1. https://www.rottentomatoes.com/

// 2. https://openclipart.org/

// 3. https://www.flaticon.com/

// --- Added MASTER CHECKBOX feature ----

import { movies } from "./data";

import DisplayMovies from "./DisplayMovies";

import PastBtn from "./PastBtn";

import GenreBtn from "./GenreBtn";

import CreateMovie from "./CreateMovie";

function App(props) {

  const [movieList, setMovieList] = React.useState(movies);

  // console.log("movieList:", movieList);

  const [hidePast, setHidePast] = React.useState(false);

  const [selectedGenre, setSelectedGenre] = React.useState("All");

  // console.log("hidePass:", hidePast);

  // Filtering of Movie List for display

  // ===================================

  // let hidePastList = movieList;

  // if (hidePast) {

  //   hidePastList = movieList.filter((movie) => movie.past === false);

  // }

  const hidePastList = hidePast

    ? movieList.filter((movie) => movie.past === false)

    : movieList;

  console.log("hidePastList:", hidePastList);

  const displayList =

    selectedGenre !== "All"

      ? hidePastList.filter((movie) => movie.genre === selectedGenre)

      : hidePastList;

  console.log("displayList:", displayList);

  return (

    <div className="movieMgmt">

      <h1>Movie Management System</h1>

      <h2>Create Movie</h2>

      <CreateMovie

        onAdd={(name, genre, past) => {

          // Get array of all the movie.id

          const ids = movieList.map((movie) => movie.id);

          console.log("movie.ids:", ids);

          // Find the max movie.id

          // Note: if array is empty, then maxId = -Infinity

          // So during setMovieList(), if maxId is -Infinity,

          // then set id=0, else id=maxId+1

          const maxId = Math.max(...ids);

          console.log("maxId:", maxId);

          setMovieList([

            ...movieList,

            {

              id: maxId == -Infinity ? 0 : maxId + 1,

              name: name,

              genre: genre,

              past: past,

              isChecked: false,

            },

          ]);

        }}

      />

      <hr />

      {/\* Display of Movies \*/}

      <h2 className="displayHeading">

        Movie List: {hidePast ? "Showing Now or Coming Soon" : "All Movies"} (

        {displayList.length})

      </h2>

      {/\* Hide/Show Past Movies Button \*/}

      {/\* ============================ \*/}

      {/\* Option 1: Call a child component (PastBtn) \*/}

      {/\* <PastBtn onClick={handlePastBtn} /> \*/}

      {/\* Option 2: Code button tag here \*/}

      {/\* <button onClick={handlePastBtn}>

        {(hidePast ? "Show" : "Hide") + " Past Movies"}

      </button> \*/}

      {/\* Option 3: Call a child component (PastBtn) and execute the handler directly \*/}

      <PastBtn

        onClick={() => setHidePast(!hidePast)}

        title={(hidePast ? "Show" : "Hide") + " Past Movies"}

      />

      {/\* Filter by Genre Button \*/}

      {/\* ====================== \*/}

      <GenreBtn onChange={(value) => setSelectedGenre(value)} />

      {console.log("SelectedGenre:", selectedGenre)}

      <br />

      <br />

      <DisplayMovies

        movies={displayList}

        // Delete 1 movie

        onDel={(selectedMovieId) => {

          console.log("movieList onDel:", movieList);

          console.log("selectedMovieId:", selectedMovieId);

          setMovieList(

            movieList.filter((movie) => movie.id !== selectedMovieId)

          );

        }}

        //

        // When checkbox is clicked (changed)

        // Id of checkbox is provided and the respective isChecked status will be toggled

        onChkBoxChange={(checkBoxId) => {

          setMovieList(

            movieList.map((movie) => {

              if (movie.id === checkBoxId) {

                return { ...movie, isChecked: !movie.isChecked };

              } else {

                return movie;

              }

            })

          );

        }}

        // Delete multiple movies

        onGrpDel={(delArray) => {

          console.log("delArray:", delArray);

          delArray.forEach(deleteFn);

        }}

        // --- MASTER CHECKBOX ---

        movieList={movieList}

        onMasterChkBox={(masterId) => {

          setMovieList(

            movieList.map((movie) => {

              if (masterId) {

                return { ...movie, isChecked: true };

              } else {

                return { ...movie, isChecked: false };

              }

            })

          );

        }}

        // --- MASTER CHECKBOX ---

      />

    </div>

  );

  // Note: Use updater function n=>n+1 for looping to work.

  // Else, only the last value (or id) in the delArray will

  // be filtered.

  function deleteFn(value) {

    setMovieList((movieList) =>

      movieList.filter((movie) => movie.id !== value)

    );

  }

} //end of App()

const root = ReactDOM.createRoot(document.getElementById("root"));

root.render(<App />);