1. The tutorial has two series
    - with react hooks
    - with redux
2. Use material ui, feature on this CRUD
    - have search input
    - pagination, and sorting from backend / api
    - have individual, selected, or batch(all) delete
    - have confirmation dialog before delete
    - have some success alert / notification, if create, update or delete is success
3. Code structure
    - for create and update, use one file form, so no duplication/repetition to write form ui and field on create and update
    - (question) for field / entities variable can store in some (maybe) like entities folder (post.js, post_category.js) ? so code like this `const initialTutorialState = { id: null, category_id: null, name: "", description: "", published: false };` can use on Table, View, Create or Update js file, no repetition to define twice
    - on some form, has a drop down, where it fetches from the database, I'll explain in the database section (post:category_id)
    - have photo for upload function
4. Database
    - post_category: id, name, description
    - post:id, category_id, name, slug, content, photo, published, created_at

GitHub Projects

- https://github.com/bezkoder/react-table-crud-example/
- https://github.com/bezkoder/redux-toolkit-example-crud
- https://github.com/bezkoder/react-redux-login-example
- https://github.com/bezkoder/react-pagination-material-ui
- https://github.com/bezkoder/react-table-pagination-server-side
- https://github.com/bezkoder/react-pagination-hooks
  (https://bezkoder.com/react-pagination-hooks/)
- https://github.com/bezkoder/redux-toolkit-example-crud-hooks
  (https://www.bezkoder.com/redux-toolkit-crud-react-hooks/)
- https://github.com/bezkoder/react-hooks-redux-crud/
  (https://www.bezkoder.com/react-hooks-redux-crud/)

MODULE:	**TOPICS**
DATA	MYSQL DB
API	REST API PHP
STATE	REACT-REDUX

Main Folder:	/app/topic/
Actions	/app/topic/actions
Action Types	/app/topic/actions/types
Reducers	/app/topic/reducers
TopicApp	/app/topic/index.js
Topics	/app/topic/Topics.js

ROUTES

Topics:	/topics	
Filter by Category	{category_name}/topics	/mobile/topics
Filter by Label	{category_name}/topics/{label}	/mobile/topics/business
Filter by Post_Status	{category_name}/topics/{label}/{post_status} eg: /mobile/topics/business/pending	
Pagination	/topics?page_no=1	

TYPE GET_TOPICS, FILTER_TOPICS

Actions /app/topic/actions

Questions/Clarifications:

- We use API to connect and get the data and return it via payload.
- How to get additional data as params?
- How do we pass the getTopics() return data to another TYPE — filterTopics?

```
export const getTopics = () => async (dispatch) => {
 try {
   const res = await TopicApiService.getAll();
   // console.log(res.data['items']) // need some work here.
   dispatch({
     type: GET_TOPICS,
     payload: res.data['topics'],
   });
 } catch (err) {
   console.log(err);
 }
};

// ** FILTER Topics
export const filterTopics = (state, action) => async (dispatch) => {
 try {
   const initialState = {
     topics: getTopics(),
     filteredTopics: [],
   };

   state.filteredTopics = state.topics.filter(
     (topic) => topic.category === action.payload
   );

   dispatch({
     type: FILTER_TOPICS,
     payload: action.payload,
   });

   return Promise.resolve(action.payload);
 } catch (err) {
   return Promise.reject(err);
 }
};
```

Reducers /app/topic/reducers

Questions:

- How to identify current topic and return additional params?
- How to return filter topics - should it be a separate TYPE or can it be done using the GET_TOPICS type.

```
const TopicReducer = (topics = initialState, action) => {

 const { type, payload } = action;

 switch (type) {
   case GET_TOPICS:
     // ** If currentTopic is not null / undefined then find and set currentTopic
     let currTopic = null
     if (topics.currentTopic !== null && topics.currentTopic !== undefined) {
       currTopic = action.data.topics.find(i => i.id === topics.currentTopic.id)
     }

     // return {
     //   // ...topics,
     //   // filterTopics: topicSlice.actions,
     //   // topicsMeta: action.data.topicsMeta,
     //   // params: action.params,
     //   // currentMail: currTopic
     // }
     return payload;

   case FILTER_TOPICS:
     // ** Find current topic & add hasNextTopic & hasPreviousTopic props to
current topic object based on index
     const data = action.data
     return { ...topics, currentTopic: data }
```

app/topic/index.js

Questions:

- How to get getTopics and getCategories from actions and pass it to other components?
- How to set initial data and also filter data?
- How to get currentCategory and currentTopic?

```
 import React, { useState, useEffect } from "react";
import { useDispatch } from "react-redux";

import Topics from "./Topics";
import Categories from "../../category/categories";
// import { filterTopics } from "./store/reducers";

import { getTopics, filterTopics } from "./store/actions";
import { getCategories } from "../../category/store/actions";

const TopicApp = () => {
 const [filter, getTopics, setFilter] = useState(false);
 const [currentCategory, setCurrentCategory] = useState("");

 const dispatch = useDispatch();
  const categoryChangeHandler = (category) => {
   setCurrentCategory(category);
   if (category === "All Categories") {
     setFilter(false);
   } else {
     dispatch(filterTopics(category));
     setFilter(true);
   }
 };
 // ** UseEffect: GET initial data on Mount
 useEffect(() => {
   dispatch(getTopics())
 }, [])

 return (
   <div className="container">
     <div className="row justify-content-center">
       <div className="category col-md-2">
         <Categories
             currentCategory={currentCategory}
             categoryChangeHandler={categoryChangeHandler}
           />
       </div>
```

```
      <div className="title col-md-6">
        <Topics
          getTopics={getTopics}
          filter={filter}
          />
      </div>
    </div>
  </div>
 );
};
export default TopicApp;
```

app/topic/Topics.js

Questions:

- This is the Topics component which is responsible for displaying the getTopics, filterTopics, categories etc.
- Is props the right method?
- How do I get the data from actions & reducers here?

```
import React from "react";
import { useSelector } from "react-redux";

const Topics = props => {

 // ** Props
 const { filter, store, getTopics } = props
 const { topics } = store
 // const { topics } = useSelector((state) => getTopics);
 const { filteredTopics } = useSelector((state) => state.topic);

return (
   <>
     <p>
       showing {filter ? filteredTopics.length : topics.length} topics of total{"
"}
       {topics.length} topics
     </p>
     <div>
       <ul>
       {!filter &&
         topics.map((topic) => (
           <li key={topic.id}>
             <span> {topic.title}</span>
           </li>
         ))}
       {filter &&
         filteredTopics.map((topic) => (
           <li key={topic.id}>
             <span> {topic.title}</span>
           </li>
         ))}
         </ul>
     </div>
   </>
 );
};
```

```
export default Topics;
```