import { createSlice, createAsyncThunk } from "@reduxjs/toolkit";

import axios from "axios";

export const createReusableSlice = ({ name, apiUrl }) => {

  const asyncActions = {

    fetchItems: createAsyncThunk(`${name}/fetchItems`, () => axios.get(apiUrl).then((res) => res.data)),

    addItem: createAsyncThunk(`${name}/addItem`, (newItem) => axios.post(apiUrl, newItem).then((res) => res.data)),

    updateItem: createAsyncThunk(`${name}/updateItem`, (updatedItem) =>

      axios.put(`${apiUrl}/${updatedItem.id}`, updatedItem).then((res) => res.data)

    ),

    deleteItem: createAsyncThunk(`${name}/deleteItem`, (id) => axios.delete(`${apiUrl}/${id}`).then(() => id)),

  };

  const slice = createSlice({

    name,

    initialState: { items: [], loading: true, error: null },

    reducers: {},

    extraReducers: (builder) => {

      builder

        .addCase(asyncActions.fetchItems.pending, (state) => { state.loading = true; state.error = null; })

        .addCase(asyncActions.fetchItems.fulfilled, (state, action) => { state.loading = false; state.items = action.payload; })

        .addCase(asyncActions.fetchItems.rejected, (state, action) => { state.loading = false; state.error = action.error.message; })

        .addCase(asyncActions.addItem.fulfilled, (state, action) => { state.items.push(action.payload); })

        .addCase(asyncActions.updateItem.fulfilled, (state, action) => {

          const index = state.items.findIndex((item) => item.id === action.payload.id);

          if (index !== -1) state.items[index] = action.payload;

        })

        .addCase(asyncActions.deleteItem.fulfilled, (state, action) => {

          state.items = state.items.filter((item) => item.id !== action.payload);

        });

    },

  });

  return { slice: slice.reducer, actions: asyncActions };

};

import { configureStore } from "@reduxjs/toolkit";

import { createReusableSlice } from "./slice.js";

const postsSlice = createReusableSlice({

  name: "test",

  apiUrl: "https://671891927fc4c5ff8f49fcac.mockapi.io/test",

});

export const store = configureStore({

  reducer: {

    test: postsSlice.slice,

  },

});

export const { fetchItems, addItem, updateItem, deleteItem } = postsSlice.actions;

