```
import { createSlice, createAsyncThunk } from "@reduxjs/toolkit";
import trainingCenterService from "services/Training/TrainingCenterService";
import { cloneDeep } from "lodash";
import ErpNotificationService from "services/common/ErpNotificationService";

export const getNoti = createAsyncThunk(
  "erpNotification/getNoti",
  async (data, { rejectWithValue }) => {
    try {
      const response = await ErpNotificationService.search(data);
      return response.data;
    } catch (err) {
      console.error("API call failed:", err);
      return rejectWithValue(err.message || "Error");
    }
  }
);
export const getSearchNoti = createAsyncThunk(
  "erpNotification/getSearchNoti",
  async (data, { rejectWithValue }) => {
    try {
      const response = await ErpNotificationService.search(data);
      return response.data;
    } catch (err) {
      console.error("API call failed:", err);
      return rejectWithValue(err.message || "Error");
    }
  }
);

export const readAllNoti = createAsyncThunk(
  "erpNotification/readAllNoti",
  async (data, { rejectWithValue }) => {
    const { onSuccess } = data;
    try {
      const response = await ErpNotificationService.ReadAll();
      if (onSuccess) onSuccess(response);
      return response.data;
    } catch (error) {
      console.error("API call failed:", error);
      return rejectWithValue(error.message || "Error");
    }
  }
);
export const deleteNoti = createAsyncThunk(
  "erpNotification/deleteNoti",
  async (data, { rejectWithValue }) => {
    const { onSuccess, id } = data;
    try {
      const response = await ErpNotificationService.Delete(id);
      if (onSuccess) onSuccess(response);
      return response.data;
    } catch (error) {
      console.error("API call failed:", error);
      return rejectWithValue(error.message || "Error");
    }
  }
);

export const readNoti = createAsyncThunk(
  "erpNotification/readNoti",
  async (data, { rejectWithValue }) => {
    const { onSuccess, id } = data;
    try {
      const response = await ErpNotificationService.Read(id);
      if (onSuccess) onSuccess(response);
      return response.data;
    } catch (error) {
      console.error("API call failed:", error);
      return rejectWithValue(error.message || "Error");
    }
```

```
  }
);

const initialState = {
  dataSearch: {
    loading: null,
    data: null,
  },
  dataNav: {
    loading: null,
    data: null,
  },
};

const erpNotificationSlice = createSlice({
  name: "erpNotification",
  initialState,
  reducers: {},
  extraReducers: (builder) => {
    builder
      .addCase(getNoti.pending, (state) => {
        state.dataNav = {
          ...state.dataNav,
          loading: true,
        };
      })
      .addCase(getNoti.fulfilled, (state, action) => {
        state.dataNav = {
          loading: false,
          data: action.payload,
        };
      })
      .addCase(getNoti.rejected, (state, action) => {
        state.dataNav = {
          ...state.dataNav,
          loading: false,
        };
      })
      .addCase(getSearchNoti.pending, (state) => {
        state.dataSearch = {
          ...state.dataSearch,
          loading: true,
        };
      })
      .addCase(getSearchNoti.fulfilled, (state, action) => {
        state.dataSearch = {
          loading: false,
          data: action.payload,
        };
      })
      .addCase(getSearchNoti.rejected, (state, action) => {
        state.dataSearch = {
          ...state.dataSearch,
          loading: false,
        };
      })
      .addCase(deleteNoti.pending, (state) => {
        state.dataSearch = {
          ...state.dataSearch,
          loading: true,
        };
      })
      .addCase(deleteNoti.fulfilled, (state, action) => {
        state.dataSearch = {
          ...state.dataSearch,
          loading: false,
        };
      })
      .addCase(deleteNoti.rejected, (state) => {
        state.dataSearch = {
          ...state.dataSearch,
          loading: false,
        };
      });
```

```
  },
});

export default erpNotificationSlice.reducer;
```