```
import { createSlice, createAsyncThunk } from "@reduxjs/toolkit";
import { cloneDeep } from "lodash";
import GhiNhanYeuCauService from "services/ghiNhanYeuCau/ghiNhanYeuCauService";

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

export const GetCateGhiNhan = createAsyncThunk(
  "ghiNhanYeuCau/GetCateGhiNhan",
  async (_, { rejectWithValue }) => {
    try {
      const response = await GhiNhanYeuCauService.GetCategory();
      return response.data;
    } catch (err) {
      console.error("API call failed:", err);
      return rejectWithValue(err.message || "Error");
    }
  }
);

export const CreateGhiNhan = createAsyncThunk(
  "ghiNhanYeuCau/CreateGhiNhan",
  async (data, { rejectWithValue }) => {
    try {
      const { onSuccess } = data;
      const payload = cloneDeep(data);
      delete payload.onSuccess;
      const response = await GhiNhanYeuCauService.CreateGhiNhan(payload);
      if (onSuccess) onSuccess(response.data);
      return response.data;
    } catch (err) {
      console.error("API call failed:", err);
      return rejectWithValue(err.message || "Error");
    }
  }
);
export const UpdateGhiNhan = createAsyncThunk(
  "ghiNhanYeuCau/UpdateGhiNhan",
  async (data, { rejectWithValue }) => {
    try {
      const { onSuccess } = data;
      const payload = cloneDeep(data);
      delete payload.onSuccess;
      const response = await GhiNhanYeuCauService.UpdateGhiNhan(payload);
      if (onSuccess) onSuccess(response);
      return response.data;
    } catch (error) {
      console.error("API call failed:", error);
      return rejectWithValue(error.message || "Error");
    }
  }
);
export const GetGhiNhanById = createAsyncThunk(
  "ghiNhanYeuCau/GetGhiNhanById",
  async (data, { rejectWithValue }) => {
    const { onSuccess, id } = data;
    try {
      const response = await GhiNhanYeuCauService.GetById(id);
      if (onSuccess) onSuccess(response);
      return response.data;
    } catch (error) {
      console.error("API call failed:", error);
      return rejectWithValue(error.message || "Error");
    }
  }
);
export const GetHistoryGhiNhanById = createAsyncThunk(
  "ghiNhanYeuCau/GetHistoryGhiNhanById",
  async (data, { rejectWithValue }) => {
    const { onSuccess, id } = data;
    try {
      const response = await GhiNhanYeuCauService.GetHistoryById(id);
      if (onSuccess) onSuccess(response);
      return response.data;
    } catch (error) {
      console.error("API call failed:", error);
      return rejectWithValue(error.message || "Error");
    }
  }
);

const initialState = {
  loading: false,
  detailId: null,
  detail: {
    loading: false,
    data: null,
  },
  list: {
    loading: false,
    data: null,
  },
  category: {
    loading: false,
    data: null,
  },
  history: {
    loading: false,
    data: [],
  },
};

export const ghiNhanYeuCauSlice = createSlice({
  name: "ghiNhanYeuCau",
  initialState,
  reducers: {},
  extraReducers: (builder) => {
    builder
      .addCase(GetHistoryGhiNhanById.pending, (state) => {
        state.history = { ...state.history, loading: true };
      })
      .addCase(GetHistoryGhiNhanById.fulfilled, (state, action) => {
        state.history = { data: action.payload, loading: false };
      })
      .addCase(GetHistoryGhiNhanById.rejected, (state, _action) => {
        state.history = { ...state.history, loading: false };
      })
      .addCase(GetGhiNhanById.pending, (state) => {
        state.detail = { ...state.detail, loading: true };
      })
      .addCase(GetGhiNhanById.fulfilled, (state, action) => {
        state.detail = { data: action.payload, loading: false };
      })
      .addCase(GetGhiNhanById.rejected, (state, _action) => {
        state.detail = { ...state.detail, loading: false };
      })
      .addCase(GetAllGhiNhan.pending, (state) => {
        state.list = { ...state.list, loading: true };
      })
      .addCase(GetAllGhiNhan.fulfilled, (state, action) => {
        state.list = { data: action.payload, loading: false };
      })
      .addCase(GetAllGhiNhan.rejected, (state, _action) => {
        state.list = { ...state.list, loading: false };
      })
      .addCase(GetCateGhiNhan.pending, (state) => {
        state.category = { ...state.category, loading: true };
      })
      .addCase(GetCateGhiNhan.fulfilled, (state, action) => {
        state.category = { data: action.payload, loading: false };
      })
      .addCase(GetCateGhiNhan.rejected, (state, _action) => {
        state.category = { ...state.category, loading: false };
      })
      .addCase(CreateGhiNhan.pending, (state) => {
        state.loading = true;
      })
      .addCase(CreateGhiNhan.fulfilled, (state, action) => {
        state.loading = false;

        state.detailId = action.payload?.data;
      })
      .addCase(CreateGhiNhan.rejected, (state) => {
        state.loading = false;
      })
      .addCase(UpdateGhiNhan.pending, (state) => {
        state.loading = true;
      })
      .addCase(UpdateGhiNhan.fulfilled, (state) => {
        state.loading = false;
      })
      .addCase(UpdateGhiNhan.rejected, (state) => {
        state.loading = false;
      });
  },
});

export default ghiNhanYeuCauSlice.reducer;
```