```
import { createSlice, createAsyncThunk } from "@reduxjs/toolkit";
import { cloneDeep } from "lodash";
import TongHopDeXuatMuaSamService from "services/sale/TongHopDeXuatMuaSamService";

export const GetAllTongHop = createAsyncThunk(
  "tonghopdexuatmuasam/GetAllTongHop",
  async (data, { rejectWithValue }) => {
    try {
      const { onSuccess } = data;
      const payload = cloneDeep(data);
      delete payload.onSuccess;
      const response = await TongHopDeXuatMuaSamService.GetAllTongHop(data);
      if (onSuccess) onSuccess(response.data.data);
      return response.data;
    } catch (err) {
      return rejectWithValue(err.message || "Error");
    }
  }
);
export const GetTongHop = createAsyncThunk(
  "tonghopdexuatmuasam/GetTongHop",
  async (data, { rejectWithValue }) => {
    try {
      const { onSuccess } = data;
      const payload = cloneDeep(data);
      delete payload.onSuccess;
      const response = await TongHopDeXuatMuaSamService.GetTongHop(data.id);
      if (onSuccess) onSuccess(response.data.data);
      return response.data;
    } catch (err) {
      return rejectWithValue(err.message || "Error");
    }
  }
);
export const GetTongHopChiTiet = createAsyncThunk(
  "tonghopdexuatmuasam/GetTongHopChiTiet",
  async (data, { rejectWithValue }) => {
    try {
      const { onSuccess } = data;
      const payload = cloneDeep(data);
      delete payload.onSuccess;
      const response = await TongHopDeXuatMuaSamService.GetTongHopChiTiet(data);
      if (onSuccess) onSuccess(response.data.data);
      return response.data;
    } catch (err) {
      return rejectWithValue(err.message || "Error");
    }
  }
);
export const CapNhatTongHop = createAsyncThunk(
  "tonghopdexuatmuasam/CapNhatTongHop",
  async (data, { rejectWithValue }) => {
    try {
      const { onSuccess } = data;
      const payload = cloneDeep(data);
      delete payload.onSuccess;
      const response = await TongHopDeXuatMuaSamService.CapNhatTongHop(data);
      if (onSuccess) onSuccess(response.data.data);
      return response.data;
    } catch (err) {
      return rejectWithValue(err.message || "Error");
    }
  }
);
export const CreateDeXuatChiTietDuyet = createAsyncThunk(
  "tonghopdexuatmuasam/CreateDeXuatChiTietDuyet",
  async (data, { rejectWithValue }) => {
    try {
      const { onSuccess } = data;
      const payload = cloneDeep(data);
      delete payload.onSuccess;
```

```
      const response =
        await TongHopDeXuatMuaSamService.CreateDeXuatChiTietDuyet(data);
      if (onSuccess) onSuccess(response.data.data);
      return response.data;
    } catch (err) {
      return rejectWithValue(err.message || "Error");
    }
  }
);
export const UpdateDeXuatChiTietDuyet = createAsyncThunk(
  "tonghopdexuatmuasam/CreateDeXuatChiTietDuyet",
  async (data, { rejectWithValue }) => {
    try {
      const { onSuccess } = data;
      const payload = cloneDeep(data);
      delete payload.onSuccess;
      const response =
        await TongHopDeXuatMuaSamService.UpdateDeXuatChiTietDuyet(data);
      if (onSuccess) onSuccess(response.data.data);
      return response.data;
    } catch (err) {
      return rejectWithValue(err.message || "Error");
    }
  }
);
export const GetDeXuatChiTietDuyet = createAsyncThunk(
  "tonghopdexuatmuasam/GetDeXuatChiTietDuyet",
  async (Id, { rejectWithValue }) => {
    try {
      const response = await TongHopDeXuatMuaSamService.GetDeXuatChiTietDuyet(
        Id
      );
      return response.data;
    } catch (err) {
      return rejectWithValue(err.message || "Error");
    }
  }
);
export const TaoBanSaoDeXuatChiTietDuyet = createAsyncThunk(
  "tonghopdexuatmuasam/TaoBanSaoDeXuatChiTietDuyet",
  async (data, { rejectWithValue }) => {
    try {
      const { onSuccess, id } = data;
      const response =
        await TongHopDeXuatMuaSamService.TaoBanSaoDeXuatChiTietDuyet(id);
      if (onSuccess) onSuccess(response.data);
      return response.data;
    } catch (err) {
      return rejectWithValue(err.message || "Error");
    }
  }
);
export const GetAllDepartmentCategory = createAsyncThunk(
  "tonghopdexuatmuasam/GetAllDepartmentCategory",
  async (_, { rejectWithValue }) => {
    try {
      const response =
        await TongHopDeXuatMuaSamService.GetAllDepartmentCategory();
      return response.data;
    } catch (err) {
      return rejectWithValue(err.message || "Error");
    }
  }
);

const initialState = {
  loading: false,
  TongHopList: [],
  TongHopDetail: {},
  TongHopChiTietList: [],
  DeXuatChiTietDuyetList: [],
  ListDetail: [],
  AllDepartmentList: [],
};
```

```
export const dexuatmuasamSlice = createSlice({
  name: "tongHopdeXuatMuaSam",
  initialState,
  reducers: {
    showLoading: (state) => {
      state.loading = true;
    },
  },
  extraReducers: (builder) => {
    builder
      //GetTongHop
      .addCase(GetTongHop.pending, (state) => {
        state.loading = true;
      })
      .addCase(GetTongHop.fulfilled, (state, action) => {
        state.loading = false;
        state.TongHopDetail = action.payload;
      })
      .addCase(GetTongHop.rejected, (state, action) => {
        state.loading = false;
      })
      //GetAllTongHop
      .addCase(GetAllTongHop.pending, (state) => {
        state.loading = true;
      })
      .addCase(GetAllTongHop.fulfilled, (state, action) => {
        state.loading = false;
        state.TongHopList = action.payload.data;
        state.ListDetail = action.payload.detail;
      })
      .addCase(GetAllTongHop.rejected, (state, action) => {
        state.loading = false;
      })
      //GetTongHopChiTiet
      .addCase(GetTongHopChiTiet.pending, (state) => {
        state.loading = true;
      })
      .addCase(GetTongHopChiTiet.fulfilled, (state, action) => {
        state.loading = false;
        state.TongHopChiTietList = action.payload;
      })
      .addCase(GetTongHopChiTiet.rejected, (state, action) => {
        state.loading = false;
      })
      .addCase(GetDeXuatChiTietDuyet.pending, (state) => {
        state.loading = true;
      })
      .addCase(GetDeXuatChiTietDuyet.fulfilled, (state, action) => {
        state.loading = false;
        state.DeXuatChiTietDuyetList = action.payload;
      })
      .addCase(GetDeXuatChiTietDuyet.rejected, (state, action) => {
        state.loading = false;
      })
      .addCase(GetAllDepartmentCategory.pending, (state) => {
        state.loading = true;
      })
      .addCase(GetAllDepartmentCategory.fulfilled, (state, action) => {
        state.loading = false;
        state.AllDepartmentList = action.payload;
      })
      .addCase(GetAllDepartmentCategory.rejected, (state, action) => {
        state.loading = false;
      });
  },
});

export const { showLoading, dexuatmuasamlist } = dexuatmuasamSlice.actions;

export default dexuatmuasamSlice.reducer;
```