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

export const getAllAmbulance = createAsyncThunk(
  "ambulance/getAllAmbulance",
  async (data, { rejectWithValue }) => {
    try {
      const response = await AmbulanceService.searchGrid(data);
      return response.data;
    } catch (err) {
      return rejectWithValue(err.message || "Error");
    }
  }
);

export const getAmbulanceById = createAsyncThunk(
  "ambulance/getAmbulanceById",
  async (id, { rejectWithValue }) => {
    try {
      const response = await AmbulanceService.getAmbulanceById(id);
      return response.data;
    } catch (err) {
      return rejectWithValue(err.message || "Error");
    }
  }
);
const initialState = {
  loading: false,
  ambulanceList: [],
  ambulanceDetail: {},
};

export const ambulanceSlice = createSlice({
  name: "ambulance",
  initialState,
  reducers: {
    showLoading: (state) => {
      state.loading = true;
    },
    setHisInfo: (state, action) => {
      state.hisInfoList = action.payload;
    },
  },
  extraReducers: (builder) => {
    builder
      .addCase(getAllAmbulance.pending, (state) => {
        state.loading = true;
      })
      .addCase(getAllAmbulance.fulfilled, (state, action) => {
        state.loading = false;
        state.ambulanceList = action.payload;
      })
      .addCase(getAllAmbulance.rejected, (state, action) => {
        state.loading = false;
      })
      .addCase(getAmbulanceById.pending, (state) => {
        state.loading = true;
      })
      .addCase(getAmbulanceById.fulfilled, (state, action) => {
        state.loading = false;
        state.ambulanceDetail = action.payload;
      })
      .addCase(getAmbulanceById.rejected, (state, action) => {
        state.loading = false;
      });
  },
});

export const { showLoading, setHisInfo } = ambulanceSlice.actions;

export default ambulanceSlice.reducer;
```