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

export const getAllComment = createAsyncThunk(
  "comment/getAllComment",
  async (data, { rejectWithValue }) => {
    try {
      const response = await CommentService.searchGrid(data);
      return response.data;
    } catch (err) {
      return rejectWithValue(err.message || "Error");
    }
  }
);
export const getCommentById = createAsyncThunk(
  "appointment/getAppointmentById",
  async (id, { rejectWithValue }) => {
    try {
      const response = await CommentService.getCommentById(id);
      return response.data;
    } catch (err) {
      return rejectWithValue(err.message || "Error");
    }
  }
);

export const createComment = createAsyncThunk(
  "appointment/createComment",
  async (data, { rejectWithValue }) => {
    try {
      const { onSuccess } = data;
      const payload = cloneDeep(data);
      delete payload.onSuccess;
      const response = await CommentService.create(payload);
      if (onSuccess) onSuccess(response);
      return response.data;
    } catch (err) {
      return rejectWithValue(err.message || "Error");
    }
  }
);

export const updateComment = createAsyncThunk(
  "appointment/updateComment",
  async (data, { rejectWithValue }) => {
    try {
      const { onSuccess } = data;
      const payload = cloneDeep(data);
      delete payload.onSuccess;
      const response = await CommentService.update(payload);
      if (onSuccess) onSuccess(response);
      return response.data;
    } catch (err) {
      return rejectWithValue(err.message || "Error");
    }
  }
);
export const deleteComment = createAsyncThunk(
  "comment/deleteComment",
  async (data, { rejectWithValue }) => {
    try {
      const { onSuccess, id } = data;
      const response = await CommentService.delete(id);
      if (onSuccess) onSuccess(response);
      return response.data;
    } catch (err) {
      return rejectWithValue(err.message || "Error");
    }
  }
);
```

```
const initialState = {
  loading: false,
  listComment: [],
  commentDetail: {},
};

export const commentSlice = createSlice({
  name: "comment",
  initialState,
  reducers: {
    showLoading: (state) => {
      state.loading = true;
    },
  },
  extraReducers: (builder) => {
    builder
      .addCase(getAllComment.pending, (state) => {
        state.loading = true;
      })
      .addCase(getAllComment.fulfilled, (state, action) => {
        state.loading = false;
        state.listComment = action.payload;
      })
      .addCase(getAllComment.rejected, (state, action) => {
        state.loading = false;
      })
      .addCase(getCommentById.pending, (state) => {
        state.loading = true;
      })
      .addCase(getCommentById.fulfilled, (state, action) => {
        state.loading = false;
        state.commentDetail = action.payload;
      })
      .addCase(getCommentById.rejected, (state, action) => {
        state.loading = false;
      });
  },
});

export const { showLoading } = commentSlice.actions;

export default commentSlice.reducer;
```