권채림 react crud 연습문제

2022-11-16

문제1 Professor

App.js

```
import React, {memo} from 'react';
import {Routes, Route } from "react-router-dom";
import ProfessorList from './pages/ProfessorList';
import ProfessorAdd from './pages/ProfessorAdd';
import ProfessorView from './pages/ProfessorView';
import ProfessorEdit from './pages/ProfessorEdit';
const App = memo(() \Rightarrow \{
return (
   <div>
     <h1>professor</h1>
     <Routes>
       <Route path='/' exapt={true} element={<ProfessorList/>}/>
       <Route path='/professor_add' element={<ProfessorAdd/>}/>
       <Route path='/professor_view/:id' element={<ProfessorView/>}/>
       <Route path='/professor_edit/:id' element={<ProfessorEdit/>}/>
     </Routes>
   </div>
);
})
export default App;
```

index.js

```
import React from 'react';
import ReactDOM from 'react-dom/client';
// /*
import App from './App';
/*/
import App from './test';
/**/
import { BrowserRouter } from 'react-router-dom';
import { Provider } from 'react-redux';
import store from './store';

const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(
```

store.js

```
import { configureStore } from "@reduxjs/toolkit";
import ProfessorSlice from "./slices/ProfessorSlice";
const store = configureStore({
    reducer: {
        ProfessorSlice: ProfessorSlice,
      }
});
export default store;
```

ProfessorSlice.js

```
import { createSlice, createAsyncThunk } from '@reduxjs/toolkit'
import axios from 'axios';
import { pending, fulfilled, rejected } from '../helper/ReduxHelper';
import { cloneDeep } from 'lodash';
/** 다중행 데이터 조회를 위한 비동기 함수 */
export const getList = createAsyncThunk("ProfessorSlice/getList", async (payload,
{rejectWithValue }) => {
  let result = null;
  const URL = process.env.REACT APP API PROFESSOR LIST;
      const response = await axios.get(URL);
      result = response.data;
   } catch (err) {
      result = rejectWithValue(err.response);
  return result;
});
/** 단일행 데이터 조회를 위한 비동기 함수 */
export const getItem = createAsyncThunk("ProfessorSlice/getItem", async (payload,
{rejectWithValue }) => {
  let result = null;
   const URL = process.env.REACT_APP_API_PROFESSOR_ITEM.replace(':id',
```

```
payload.id);
   console.log(URL);
  try {
       const response = await axios.get(URL);
       result = response.data;
   } catch (err) {
       result = rejectWithValue(err.response);
   }
  return result;
});
/** 데이터 저장을 위한 비동기 함수 */
export const postItem = createAsyncThunk("ProfessorSlice/postItem", async
(payload, {rejectWithValue }) => {
   let result = null;
   const URL = process.env.REACT_APP_API_PROFESSOR_LIST;
  try {
       const response = await axios.post(URL, {
           name: payload.name,
           userid: payload.userid,
           position: payload.position,
           sal: payload.sal,
           hiredate: payload.hiredate,
           comm: payload.comm,
           deptno: payload.deptno,
       });
      result = response.data;
   } catch (err) {
       result = rejectWithValue(err.response);
   return result;
});
/** 데이터 수정을 위한 비동기 함수 */
export const putItem = createAsyncThunk("ProfessorSlice/putItem", async (payload,
{rejectWithValue }) => {
   let result = null;
   const URL = process.env.REACT APP API PROFESSOR ITEM.replace(':id',
payload.id);
   try {
       const response = await axios.put(URL, {
           name: payload.name,
           userid: payload.userid,
           position: payload.position,
           sal: payload.sal,
           hiredate: payload.hiredate,
           comm: payload.comm,
           deptno: payload.deptno,
       });
       result = response.data;
```

```
} catch (err) {
       result = rejectWithValue(err.response);
   }
  return result;
});
/** 데이터 삭제을 위한 비동기 함수 */
export const deleteItem = createAsyncThunk("ProfessorSlice/deleteItem", async
(payload, {rejectWithValue }) => {
  let result = null;
   const URL = process.env.REACT_APP_API_PROFESSOR_ITEM.replace(':id',
payload.id);
  try {
       const response = await axios.delete(URL);
       result = response.data;
   } catch (err) {
       result = rejectWithValue(err.response);
  return result;
});
const ProfessorSlice = createSlice({
   name: 'ProfessorSlice',
   initialState: {
       data: null,
       loading: false,
       error: null
   },
   reducers: {
       getCurrentData: (state, action) => {
           return state;
       }
   },
   extraReducers: {
       [getList.pending]: pending,
       [getList.fulfilled]: fulfilled,
       [getList.rejected]: rejected,
       [getItem.pending]: pending,
       [getItem.fulfilled]: (state, {meta, payload}) => {
           return {
               data: [payload],
               loading: false,
               error: null
           }
       },
       [getItem.rejected]: rejected,
```

```
[postItem.pending]: pending,
       [postItem.fulfilled]: (state, {meta, payload}) => {
           const data = cloneDeep(state.data);
           console.log("data", data);
           data.push(payload);
           return {
               data:data,
               loading: false,
               error: null
           }
       },
       [postItem.rejected]: rejected,
       [deleteItem.pending]: pending,
       [deleteItem.fulfilled]: (state, {meta, payload}) => {
           console.log("meta", meta)
           const data = cloneDeep(state.data);
           const targetId = data.findIndex((v, i) => v.id ===
Number(meta.arg.id));
           console.log("targetId", targetId);
           data.splice(targetId, 1);
           return {
               data: data,
               loading: false,
               error: null
       },
       [deleteItem.rejected]: rejected,
       [putItem.pending]: pending,
       [putItem.fulfilled]: (state, {meta, payload}) => {
           const data = cloneDeep(state.data);
           const targetId = data.findIndex((v, i) => v.id ===
Number(meta.arg.id));
           console.log("targetId", targetId);
           data.splice(targetId, 1, payload);
           return {
               data: data,
               loading: false,
               error: null
           }
       },
       [putItem.rejected]: rejected,
   },
});
export const {getCurrentData} = ProfessorSlice.actions;
```

```
export default ProfessorSlice.reducer;
```

ProfessorList.js

```
import React, {memo, useCallback} from 'react';
import { useDispatch, useSelector } from 'react-redux';
import { NavLink, useNavigate } from 'react-router-dom';
import Spinner from '../components/Spinner';
import ErrorView from '../components/ErrorView';
import Table from '../components/Table';
import { deleteItem, getList } from '../slices/ProfessorSlice';
import styled from 'styled-components';
import dayjs from 'dayjs';
const ControlContainer = styled.form`
   position: sticky;
  top:0;
   background-color: #fff;
   border-top: 1px solid #eee;
   border-bottom: 1px solid #eee;
   padding: 10px 0;
   .controll {
       margin-right: 5px;
       display: inline-block;
       font-size: 16px;
       padding: 7px 10px 5px 10px;
       border: 1px solid #ccc;
   }
   .clickable {
       background-color: #fff;
       color: #000;
       text-decoration: none;
       cursor: pointer;
       &:hover {
           background-color: #06fe06;
       }
       &:active {
          transform: scale(0.9, 0.9);
       }
   }
const ProfessorList = memo(() => {
   const dispatch = useDispatch();
   const {data, loading, error} = useSelector((state) => state.ProfessorSlice);
```

```
React.useEffect(()=>{
      dispatch(getList())
  },[]);
  const navigate = useNavigate();
  const onProfessorItemDelete = useCallback((e) => {
      e.preventDefault();
      const current = e.currentTarget;
      const { id, name} = current.dataset;
      if(window.confirm(`정말 ${name}(을)를 삭제하시겠습니까?`)) {
          dispatch(deleteItem({
             id: id
         }))
      }
  }, []);
  const onProfessorEditClick = useCallback((e) => {
      e.preventDefault();
      const current = e.currentTarget;
      const {id} = current.dataset;
      navigate(`/professor_edit/${id}`);
  })
  return(
      <div>
          {/* 로딩바 */}
          <Spinner loading={loading}/>
          {/* 추가탭 */}
          <ControlContainer>
             <NavLink to="professor_add" className="controll clickable">학과정보
추가하기</NavLink>
          </ControlContainer>
          {/* 조회결과 */}
          {error ? (
             <ErrorView error={error}/>
          ) : (
             data && (
                 <Table>
                     <thead>
                        >
                            No.
                            이름
                            아이디
                            직급
                            급여
                            입사일
                            보직수당
```

```
소속학과번호
                        수정
                        삭제
                     </thead>
                  {
                        data.length > 0 ? (
                           data.map((v, i) \Rightarrow {
                              return (
                                 {v.id}
                                    <NavLink to=
{\rangle / \professor_view/\$\{v.id}\rangle \rangle \rangle \name\} </NavLink>
                                    {v.userid}
                                    {v.position}
                                    {v.sal}
                                    {dayjs(v.hiredate).format('YYYY-MM-DD')}
                                    {v.comm}
                                    {v.deptno}
                                       <button type='button' data-id =
{v.id} data-name={v.name} onClick={onProfessorEditClick} >
                                          수정하기
                                       </button>
                                    <button type='button' data-id =</pre>
{v.id} data-name={v.name} onClick={onProfessorItemDelete}>
                                          삭제하기
                                       </button>
                                    )
                           })
                        ):(
                           검색결과가 없
습니다.
                           )
                     }
                  </Table>
           )
        )}
     </div>
  )
})
```

```
export default ProfessorList;
```

ProfessorAdd.js

```
import React, {memo, useCallback} from 'react';
import TableEx from '../components/TableEx';
import { useNavigate } from 'react-router-dom';
import { useSelector, useDispatch } from 'react-redux';
import { postItem } from '../slices/ProfessorSlice';
import Spinner from '../components/Spinner';
import ErrorView from '../components/ErrorView';
const ProfessorAdd = memo(() => {
  const navigate = useNavigate();
  const dispatch = useDispatch();
  const { loading, error} = useSelector((state) => state.ProfessorSlice);
  const onProfessorSubmit = useCallback((e)=>{
       e.preventDefault();
       const current = e.currentTarget;
       dispatch(postItem({
           name: current.name.value,
           userid: current.userid.value,
           position: current.position.value,
           sal: current.sal.value,
           hiredate: current.hiredate.value,
           comm: current.comm.value,
           deptno: current.deptno.value,
       })).then((result) => {
           console.log("result:",result);
           navigate(`/professor_view/${result.payload.id}`);
       })
  }, [])
  return(
       <div>
           <Spinner loading={loading}/>
           {error ? (
               <ErrorView error={error}/>
           ): (
           <form onSubmit={onProfessorSubmit}>
               <TableEx>
                   <colgroup>
                       <col width="120"/>
                       <col/>
                   </colgroup>
```

```
이름
               <input className="field" type="text" name="name"/>
               아이디
                  <input className="field" type="text"</pre>
name="userid"/>
               직급
                  <input className="field" type="text"</pre>
name="position"/>
               급여
                  <input className="field" type="text" name="sal"/>
               입사일
                  <input className="field" type="text"</pre>
name="hiredate"/>
               보직수당
                  <input className="field" type="text" name="comm"/>
               소속학과번호
                  <input className="field" type="text"</pre>
name="deptno"/>
               </TableEx>
        <div style={{textAlign: 'center'}}>
           <button type='submit'>저장하기</button>
        </div>
      </form>
```

ProfessorEdit.js

```
import React, {memo, useCallback, useEffect, useMemo} from 'react';
import TableEx from '../components/TableEx';
import { useNavigate, useParams } from 'react-router-dom';
import { useSelector, useDispatch } from 'react-redux';
import { getCurrentData, getItem, putItem } from '../slices/ProfessorSlice';
import Spinner from '../components/Spinner';
import ErrorView from '../components/ErrorView';
import dayjs from 'dayjs';
const ProfessorEdit = memo(() => {
   const { id } = useParams();
   const dispatch = useDispatch();
   const { data, loading, error} = useSelector((state) => state.ProfessorSlice);
   useEffect(()=> {
       dispatch(getCurrentData());
   },[]);
   const item = useMemo(() => {
       if(data) {
           return data.find((v, i) => v.id == id);
       } else {
           dispatch(getItem({id:id}));
   }, [data])
   const navigate = useNavigate();
   const onProfessorSubmit = useCallback((e)=>{
       e.preventDefault();
       const current = e.currentTarget;
       dispatch(putItem({
           id: current.id.value,
           name: current.name.value,
           userid: current.userid.value,
           position: current.position.value,
           sal: current.sal.value,
```

```
hiredate: current.hiredate.value,
        comm: current.comm.value,
        deptno: current.deptno.value,
     })).then((result) => {
        console.log("result:",result);
        navigate(`/professor_view/${result.payload.id}`);
     })
  },[])
  return(
     <div>
        <Spinner loading={loading}/>
        {error ? (
           <ErrorView error={error}/>
        ) : (
        <form onSubmit={onProfessorSubmit}>
           <input type="hidden" name="id" defaultValue={item?.id} />
           <TableEx>
              <colgroup>
                  <col width="120"/>
                 <col/>
              </colgroup>
              이름
                     <input className="field" type="text" name="name"</pre>
defaultValue={item?.name}/>
                     아이디
                        <input className="field" type="text" name="userid"</pre>
defaultValue={item?.userid}/>
                     직급
                        <input className="field" type="text"</pre>
name="position" defaultValue={item?.position}/>
                     급여
                        <input className="field" type="text" name="sal"</pre>
defaultValue={item?.sal}/>
                     입사일
```

```
<input className="field" type="text"</pre>
name="hiredate" defaultValue={dayjs(item?.hiredate).format('YYYY-MM-DD')}/>
                     보직수당
                        <input className="field" type="text" name="comm"</pre>
defaultValue={item?.comm}/>
                     소속학과번호
                        <input className="field" type="text" name="deptno"</pre>
defaultValue={item?.deptno}/>
                     </TableEx>
            <div style={{textAlign: 'center'}}>
               <button type='submit'>저장하기</button>
            </div>
        </form>
        )}
     </div>
  )
})
export default ProfessorEdit;
```

ProfessorView.js

```
import React, {memo, useCallback, useEffect, useMemo} from 'react';
import { NavLink, useParams, useNavigate } from 'react-router-dom';
import { useSelector, useDispatch } from 'react-redux';
import { getCurrentData, deleteItem, getItem } from '../slices/ProfessorSlice';

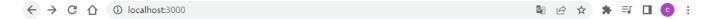
import Spinner from '../components/Spinner';
import ErrorView from '../components/ErrorView';
import Table from '../components/Table';
import dayjs from 'dayjs';

const ProfessorView = memo(() => {
   const {id} = useParams();

   const dispatch = useDispatch();
   const { data, loading, error} = useSelector((state) => state.ProfessorSlice);
```

```
useEffect(()=>{
   dispatch(getCurrentData());
}, [])
const item = useMemo(() => {
   if(data) {
       return data.find((v,i)=> v.id == id);
   } else {
       dispatch(getItem({id: id}));
}, [data]);
const navigate = useNavigate();
const onProfessorItemDelete = useCallback((e) => {
   e.preventDefault();
   const current = e.currentTarget;
   const { id, name} = current.dataset;
   if(window.confirm(`정말 ${name}(을)를 삭제하시겠습니까?`)) {
       dispatch(deleteItem({
           id: id
       })).then(({meta, payload}) => {
           navigate('/');
       })
   }
}, []);
return(
   <div>
       <Spinner loading={loading}/>
       {error ? (
           <ErrorView error={error}/>
       ) : (
           item && (
               <div>
                  <Table>
                      <colgroup>
                          <col width="120"/>
                          <col/>
                      </colgroup>
                      >
                              No.
                              {item.id}
                          이름
                              {item.name}
```

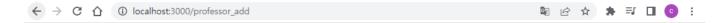
```
아이디
                         {item.userid}
                      직급
                         {item.position}
                      급여
                         {item.sal}
                      입사일
                         {dayjs(item.hiredate).format('YYYY-MM-DD')}
보직수당
                         {item.comm}
                      소속학과번호
                         {item.deptno}
                      </Table>
                <div style={{textAlign: 'center'}}>
                   <NavLink to="/">목록</NavLink>
                     | 
                   <NavLink to="/professor_add">등록</NavLink>
                     |  
                   <NavLink to={`/professor_edit/${item.id}`}>수정
</NavLink>
                     | 
                   <NavLink to="#!" data-id = {item.id} data-name=</pre>
{item.name} onClick={onProfessorItemDelete}>삭제</NavLink>
                </div>
             </div>
          )
       )}
    </div>
  )
})
export default ProfessorView;
```



professor

학과정보 추가하기

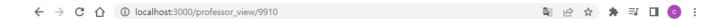
No.	이름	아이디	직급	급여	입사일	보직수당	소속학과번호	수정	삭제
9901	<u>김도훈</u>	capool	교수	500	1982-06-12	20	101	수정하기	삭제하기
9902	<u>이재우</u>	sweat413	조교수	320	1995-04-12		201	수정하기	삭제하기
9903	<u>성연희</u>	pascal	조교수	360	1993-03-17	15	101	수정하기	삭제하기
9904	<u>염일웅</u>	blue77	전임강사	240	1998-10-11		102	수정하기	삭제하기
9905	<u>권혁일</u>	refresh	교수	450	1986-02-11	25	102	수정하기	삭제하기
9906	<u>이만식</u>	pocari	부교수	420	1988-07-11		101	수정하기	삭제하기
9907	<u>전은지</u>	totoro	전임강사	210	2001-05-11		101	수정하기	삭제하기
9909	<u>권채림2</u>	test2	test	test	2014-12-23	test	test	수정하기	삭제하기



professor

이름	
아이디	
직급	
급여	
입사일	
보직수당	
소속학과번호	

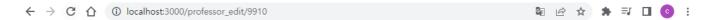
저장하기



professor

No.	9910
이름	권채림33
아이디	test
직급	test
급여	44
입사일	2014-12-23
보직수당	222
소속학과번호	3

<u>목록 | 등록 | 수정 | 삭제</u>



professor

이름	권채림 수정
아이디	test
직급	test
급여	44
입사일	2014-12-23
보직수당	222
소속학과번호	3

저장하기

