연습 문제 11 박세영

DOM 구조



components 파일들과 excptions, libs 는 실습 내용과 같습니다.

Index.js

App.js

```
import React from 'react';
import {Routes, Route} from 'react-router-dom';
import PropessorAdd from './pages/PropessorAdd';
import PropessorList from './pages/PropessorList';
import PropessorEdit from './pages/PropessorEdit';
function App() {
 return (
    <>
      <h1> Axios hooks CRUD 연습문제</h1>
      <br />
      <Routes>
        <Route path="/" export={true} element={<PropessorList/>}/>
        <Route path='/add' element={<PropessorAdd/>}/>
        <Route path='/edit/:id' element={<PropessorEdit/>}/>
      </Routes>
    </>>
 );
}
export default App;
```

PropessorList.js

```
import React from "react";
import useAxios from "axios-hooks";
import styled from "styled-components";
import { NavLink } from "react-router-dom";
import dayjs from "dayjs";
import Spinner from "../components/Spinner";
import Table from "../components/Table";
const LinkContainer = styled.div`
 position: sticky;
 top: 0;
 background-color: #fff;
 border-top: 1px solid #eee;
 border-bottom: 1px solid #eee;
 padding: 10px 0;
const TopLink = styled(NavLink)`
 margin-top: 15px;
 display: inline-block;
```

```
font-size: 16px;
 padding: 7px 10px 5px 10px;
 color: #fff;
 text-decoration: none;
 background-color: #168;
 &:hover {
   background-color: #124f60;
 }
`;
const PropessorList = () => {
 //화면에 표시할 교수 목록 데이터를 저장하기 위한 상태 변수
 const [professor, setProfessor] = React.useState([]);
 //백엔드로부터 데이터 불러오기 - 자체 캐시기능 방지
 const [{ data, loading, error }, refetch] = useAxios(
   "http://localhost:3001/professor",
     useCache: false,
 );
 React.useEffect(()=>{
     setProfessor(data);
 },[data])
 //백엔드로부터 데이터 삭제하기
  const [{ loading2 }, sendDelete] = useAxios(
   {
     method: "DELETE",
   },
     useCache: false,
     manual: true, //수동 전송
   }
 );
 //삭제 버튼 클릭시 호출될 이벤트 핸들러
 const onDeleteClick = (e) => {
   e.preventDefault();
   //이벤트가 발생한 대상을 가져옴
   const current = e.target;
   //클릭된 링크에 숨겨져잇는 dataset의 일련번호와 교수 이름 가져오기
   const id = parseInt(current.dataset.id);
   //알림창에 쓸 이름값
   const name = current.dataset.name;
   //삭제확인
   if (window.confirm(`정말 ${name}을 삭제하시겠습니까?`)) {
     (async () => {
       let json = null;
       try {
         const response = await sendDelete({
```

```
method: "Delete",
        url: `http://localhost:3001/professor/${id}`,
      });
      json = response.data;
     } catch (e) {
      console.error(e);
      window.alert(
        `[${e.response.status}] ${e.response.statusText}\n${e.message}`
      );
     }
     if (json != null) {
      setProfessor((grade) => grade.filter((v, i) => v.id !== id));
   })();
 }
};
//axios-hooks에의해 생성된 상태값 data가 변경되었을때 실행될 hook
React.useEffect(() => {
 setProfessor(data);
}, [data]);
return (
 <>
   <Spinner visible={loading || loading2} />
   <LinkContainer>
     <TopLink to="add">교수 등록하기</TopLink>
   </LinkContainer>
   {error ? (
     <div>
      <h1> {error.code} Error </h1>
      <hr />
      {error.message}
     </div>
   ) : (
     <Table>
      <thead>
        No.
          이름
          아이디
          직급
          급여
          입사일
          보직수당
          소속학과번호
          수정
          삭제
        </thead>
      {professor &&
          professor.map(
```

```
{ id, name, userid, position, sal, hiredate, comm, deptno },
             ) => {
              return (
                {id}
                  {name}
                  {userid}
                  {position}
                   {sal} {sal ? "만원" : ""}
                  {dayjs(`${hiredate}`).format("YYYY-MM-DD")}
                   {comm} {comm ? "만원" : ""}
                  {deptno}
                  <NavLink
                     to={`edit/${id}`}
                     style={{ textDecoration: "none", color: "green" }}
                     수정하기
                   </NavLink>
                  {/*클릭시 클릭된 항목의 id와 name값을 뽑아옴 */}
                   ≺a
                     href="#!"
                     data-id={id}
                     data-name={name}
                     style={{ textDecoration: "none", color: "red" }}
                     onClick={onDeleteClick}
                     삭제 하기
                   </a>
                  );
             }
           )}
        </Table>
    )}
   </>>
 );
};
export default PropessorList;
```

PropessorAdd.js

```
import React from "react";
import useAxios from "axios-hooks";
import styled from "styled-components";
import { useNavigate } from "react-router-dom";
import Spinner from "../components/Spinner";
import Table from "../components/Table";
import regexHelper from "../libs/RegexHelper";
//Table컴포넌트의 css를 확장한 컴포넌트
const TableEx = styled(Table)`
 margin-top: 50px;
 margin-bottom: 15px;
 .inputWrapper {
   padding: 0;
   position: relative;
   text-align: left;
   .field {
     box-sizing: border-box;
     display: block;
     left: 0;
     top: 0;
     width: 100%;
     height: 100%;
     border: 0;
     padding: 0 10px;
     outline: none;
     font-size: 14px;
   }
   label {
     margin-left: 7px;
     margin-right: 10px;
     input {
       margin-right: 10px;
     }
   }
 }
const PropessorAdd = () => {
 //저장 완료 후 목록으로 되돌아가기 위한 페이지 강제 이동 함수 생성
 const navigate = useNavigate();
 //백엔드에 데이터 저장을 위한 ajax요청 객체 생성- 메뉴얼 전송 모드
 const [{ loading }, refetch] = useAxios(
     url: "http://localhost:3001/professor",
     method: "POST",
```

```
{ manual: true }
);
const [{ data, loading: loading2, error }] = useAxios(
 "http://localhost:3001/department"
);
//form태그의 submit버튼이 눌러졌을때 호출될 이벤트 핸들러
const onSubmit = React.useCallback((e) => {
 e.preventDefault();
 //이벤트가 발생한 폼 객체
 const current = e.target;
 //입력값에 대한 유효성 검사
 try {
   regexHelper.value(current.name, "이름을 입력하세요");
   regexHelper.kor(current.name, "이름은 한글로 입력하세요");
   regexHelper.minLength(
     current.name,
     2,
     "이름은 최소 2글자 이상 입력해야 합니다"
   );
   regexHelper.maxLength(
     current.name,
     10,
     "이름은 최대 10글자까지 입력 가능합니다"
   );
   regexHelper.value(current.userid, "아이디를 입력하세요.");
   regexHelper.engNum(
     current.userid,
     "아이디는 영문과 숫자조합으로만 가능합니다"
   );
   regexHelper.minLength(
     current.userid,
     "아이디는 최소 2글자 이상 입력해야 합니다"
   regexHelper.maxLength(
     current.userid,
     "아이디는 최대 20글자까지 입력 가능합니다"
   regexHelper.check(current.position, "직급을 선택하세요");
   regexHelper.value(current.sal, "급여를 입력하세요");
   regexHelper.value(current.sal, "급여는 숫자만 입력 가능합니다");
   regexHelper.value(current.hiredate, "입사일을 입력하세요");
   regexHelper.value(current.comm, "보직 수당을 입력하세요");
   regexHelper.value(current.comm, "보직 수당은 숫자만 입력 가능합니다");
   regexHelper.value(current.deptno, "소속 학과를 선택하세요");
 } catch (e) {
   window.alert(e.message);
   e.field.focus();
   return;
```

```
//입력받은 값 취득
 const name = current.name.value;
 const userid = current.userid.value;
 const position = current.position.value;
 const sal = current.sal.value;
 const hiredate = current.hiredate.value;
 const comm = current.comm.value;
 const deptno = current.deptno.value;
 let json = null;
  (async () => {
   try {
     const response = await refetch({
       data: {
         name: name,
         userid: userid,
         position: position,
         sal: sal,
         hiredate: hiredate,
         comm: comm,
         deptno: deptno,
       },
     });
     json = response.data;
   } catch (e) {
     console.error(e);
     window.alert(
       `[${e.response.status}] ${e.response.statusText} \n${e.message}`
     );
   }
   if (json != null) {
     window.alert("저장되었습니다.");
     navigate("/");
   }
 })();
}, []);
return (
 <>
   <Spinner visible={loading || loading2} />
   <h1>교수 등록</h1>
   <form onSubmit={onSubmit}>
     <TableEx>
       <colgroup>
         <col width="120" />
         <col />
       </colgroup>
       이름
           <input type="text" className="field" name="name" />
```

```
아이디
 <input type="text" className="field" name="userid" />
 직급
 <label>
    <input type="radio" name="position" value="교수" />
  </label>
  <label>
    <input type="radio" name="position" value="부교수" />
    부교수
  </label>
  <label>
    <input type="radio" name="position" value="조교수" />
    조교수
  </label>
  <label>
    <input type="radio" name="position" value="전임강사" />
    전임강사
  </label>
 >
 급여(만원)
 <input</pre>
    type="number"
    className="field"
    name="sal"
    placeholder="숫자만 입력"
  />
 입사일
 <input type="date" className="field" name="hiredate" />
 보직수당(만원)
 <input</pre>
    type="number"
    className="field"
    name="comm"
    placeholder="숫자만 입력"
```

```
소속학과
           <select className="field" name="deptno">
               <option value=""> -- 선택하세요 -- </option>
               {data &&
                 data.map((v, i) \Rightarrow {
                  return (
                    <option key={i} value={v.id}>
                      {v.dname}
                    </option>
                  );
                })}
             </select>
           </TableEx>
      <div style={{ textAlign: "center" }}>
        <button type="submit">저장하기</button>
      </div>
     </form>
   </>>
 );
};
export default PropessorAdd;
```

PropessorEdit.js

```
import React from "react";

import useAxios from "axios-hooks";
import styled from "styled-components";
import { useNavigate, useParams } from "react-router-dom";
import dayjs from "dayjs";

import regexHelper from "../libs/RegexHelper";

import Spinner from "../components/Spinner";
import Table from "../components/Table";

//Table컴포넌트의 css를 확장한 컴포넌트
const TableEx = styled(Table)`
margin-top: 50px;
margin-bottom: 15px;
.inputWrapper {
    padding: 0;
```

```
position: relative;
   text-align: left;
   .field {
     box-sizing: border-box;
     display: block;
     left: 0;
     top: 0;
     width: 100%;
     height: 100%;
     border: 0;
     padding: 0 10px;
     outline: none;
     font-size: 14px;
   }
   label {
     margin-left: 7px;
     margin-right: 10px;
     input {
      margin-right: 10px;
   }
 }
const PropessorEdit = () => {
 //path파라미터로 전달된 일련 번호
 const { id } = useParams();
 //데이터 수정후 목록 페이지로 강제 이동하기 위한 함수
 const navigate = useNavigate();
 const [{ data: department, loading: loading2, error: error2 }] = useAxios(
   "http://localhost:3001/department"
 );
 //수정할 대상을 백엔드로부터 로드- 자동 실행모드
 const [{ data, loading, error }, refresh] = useAxios(
   `http://localhost:3001/professor/${id}`
 );
 const onSubmit = React.useCallback((e) => {
   e.preventDefault();
   //이벤트가 발생한 폼 객체
   const current = e.target;
   //입력값에 대한 유효성 검사
   try {
     regexHelper.value(current.name, "이름을 입력하세요");
     regexHelper.kor(current.name, "이름은 한글로 입력하세요");
     regexHelper.minLength(
       current.name,
       2,
       "이름은 최소 2글자 이상 입력해야 합니다"
```

```
);
 regexHelper.maxLength(
   current.name,
   10,
   "이름은 최대 10글자까지 입력 가능합니다"
 regexHelper.value(current.userid, "아이디를 입력하세요.");
 regexHelper.engNum(
   current.userid,
   "아이디는 영문과 숫자조합으로만 가능합니다"
 );
 regexHelper.minLength(
   current.userid,
   "아이디는 최소 2글자 이상 입력해야 합니다"
 regexHelper.maxLength(
   current.userid,
   "아이디는 최대 20글자까지 입력 가능합니다"
 );
 regexHelper.check(current.position, "직급을 선택하세요");
 regexHelper.value(current.sal, "급여를 입력하세요");
 regexHelper.value(current.sal, "급여는 숫자만 입력 가능합니다");
 regexHelper.value(current.hiredate, "입사일을 입력하세요");
 regexHelper.value(current.comm, "보직 수당을 입력하세요");
 regexHelper.value(current.comm, "보직 수당은 숫자만 입력 가능합니다");
 regexHelper.value(current.deptno, "소속 학과를 선택하세요");
} catch (e) {
 window.alert(e.message);
 e.field.focus();
 return;
//입력받은 값 취득
const name = current.name.value;
const userid = current.userid.value;
const position = current.position.value;
const sal = current.sal.value;
const hiredate = current.hiredate.value;
const comm = current.comm.value;
const deptno = current.deptno.value;
let json = null;
//입력, 수정, 삭제 처리
(async () => {
 try {
   const response = await refresh({
     method: "PUT",
     data: {
       name: name,
       userid: userid,
       position: position,
       sal: sal,
       hiredate: hiredate,
```

```
comm: comm,
        deptno: deptno,
      },
     });
     json = response.data;
   } catch (e) {
     console.error(e);
     window.alert(
       `[${e.response.status}] ${e.response.statusText} \n${e.message}`
     );
   }
   if (json != null) {
    window.alert("수정되었습니다.");
     navigate("/");
   }
 })();
}, []);
return (
 <>
   <Spinner visible={loading || loading2} />
   {error || error2 ? (
     <div>
       <h1>{error.code} Error!!</h1>
       <hr />
       {error.message}
     </div>
   ):(
     //ajax를 통해 조회한 결과가 존재할 때만 데이터를 표시함
     data && (
       <form onSubmit={onSubmit}>
        <TableEx>
          <colgroup>
            <col width="120" />
            <col />
          </colgroup>
          이름
              <input</pre>
                 type="text"
                 className="field"
                 name="name"
                 defaultValue={data.name}
               />
              아이디
              <input</pre>
                 type="text"
                 className="field"
                 name="userid"
```

```
defaultValue={data.userid}
   />
 직급
 <label>
     <input</pre>
      type="radio"
      name="position"
      value="교수"
      defaultChecked={data.position === "교수"}
     />
     교수
   </label>
   <label>
     <input</pre>
      type="radio"
      name="position"
      value="부교수"
      defaultChecked={data.position === "부교수"}
     />
     부교수
   </label>
   <label>
     <input</pre>
      type="radio"
      name="position"
      value="조교수"
      defaultChecked={data.position === "조교수"}
     />
     조교수
   </label>
   <label>
     <input</pre>
      type="radio"
      name="position"
      value="전임강사"
      defaultChecked={data.position === "전임강사"}
     />
     전임강사
   </label>
 급여(만원)
 <input</pre>
     type="number"
     className="field"
     name="sal"
     defaultValue={data.sal}
     placeholder="숫자만 입력"
```

```
/>
      입사일
      <input</pre>
         type="date"
         className="field"
         name="hiredate"
         defaultValue={dayjs(`${data.hiredate}`).format(
          "YYYY-MM-DD"
         )}
       />
      보직수당(만원)
      <input</pre>
         type="number"
         className="field"
         name="comm"
         defaultValue={data.comm}
         placeholder="숫자만 입력"
       />
      소속학과
      <select
         className="field"
         name="deptno"
         defaultValue={data.deptno}
         <option value=""> -- 선택하세요 -- </option>
         {department &&
          department.map((v, i) => {
            return (
             <option key={i} value={v.id}>
               {v.dname}
             </option>
            );
          })}
       </select>
      </TableEx>
 <div style={{ textAlign: "center" }}>
   <button type="submit">저장하기</button>
 </div>
</form>
```

```
)
))

);
);
);
export default PropessorEdit;
```

출력 결과

유효성 검사시 등록 페이지와 수정 페이지가 같은 결과이기 때문에 등록 페이지에만 출력 결과를 첨부했습니 다

초기 화면

Axios hooks CRUD 연습문제

교수 등록하기

No.	이름	아이디	직급	급여	입사일	보직수당	소속학과번호	수정	삭제
9901	김도훈	capool	교수	500 만원	1982-06-12	20 만원	101	수정하기	삭제 하기
9902	이재우	sweat413	조교수	320 만원	1995-04-12		201	수정하기	삭제 하기
9903	성연희	pascal	조교수	360 만원	1993-03-17	15 만원	101	수정하기	삭제 하기
9904	염일웅	blue77	전임강사	240 만원	1998-10-11		102	수정하기	삭제 하기
9905	권혁일	refresh	교수	450 만원	1986-02-11	20 만원	102	수정하기	삭제 하기
9906	이만식	pocari	부교수	420 만원	1988-07-11		101	수정하기	삭제 하기
9907	전은지	totoro	전임강사	210 만원	2001-05-11		101	수정하기	삭제 하기
9908	남은혁	bird13	부교수	400 만원	1990-10-18	17 만원	202	수정하기	삭제 하기

Add

이름 입력을 하지 않았을시

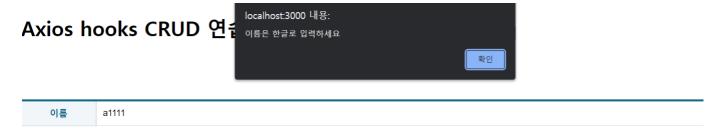


교수 등록



저장하기

이름 입력시 한글이 아닐 경우

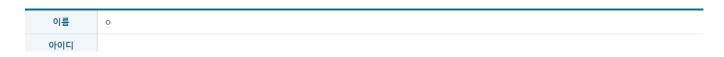


이름 입력시 최소 글자수나 최대 글자수가 맞지 않을 경우

최소 글자수



교수 등록



최대 글자수

2022. 5. 19.

Axios hooks CRUD 연 f 이름은 최대 10글자까지 입력 가능합니다

localhost:3000 내용: 이름은 최대 10글자까지 입력 가능합니다 확인

교수 등록



ID를 입력하지 않았을 경우



교수 등록



ID 입력시 최소 글자수나 최대 글자수가 맞지 않을 경우

최소 글자수

Axios hooks CRUD 연 on only 10 calhost:3000 내용:
아이디는 최소 2글자 이상 입력해야 합니다

교수 등록



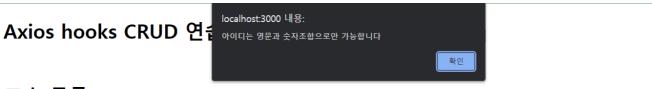
최대 글자수



교수 등록

이름	000
아이디	ааааааааааааааааааааа
직급	○ 교수 ○ 부교수 ○ 조교수 ○ 전임강사

ID입력시 영문과 숫자가 아닐 경우



교수 등록

이름	000
아이디	0000

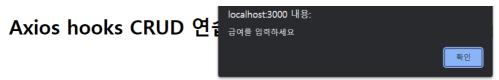
직급을 선택하지 않았을 경우



교수 등록



급여를 입력하지 않았을 경우



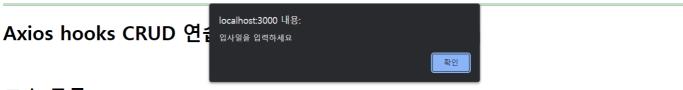
교수 등록



급여 입력시 숫자가 아닐 경우



입사일을 입력하지 않았을 경우



교수 등록



보직수당을 입력하지 않았을 경우



교수 등록



보직수당 입력시 숫자가 아닐 경우



소속학과를 선택하지 않았을 경우

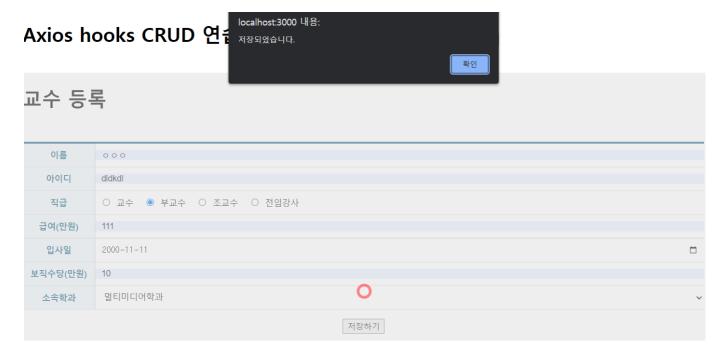


교수 등록



저장하기

교수 등록시



저장된 목록

교수 등록하기

No.	이름	아이디	직급	급여	입사일	보직수당	소속학과번호	수정	삭제
9901	김도훈	capool	교수	500 만원	1982-06-12	20 만원	101	수정하기	삭제 하기
9902	이재우	sweat413	조교수	320 만원	1995-04-12		201	수정하기	삭제 하기
9903	성연희	pascal	조교수	360 만원	1993-03-17	15 만원	101	수정하기	삭제 하기
9904	염일웅	blue77	전임강사	240 만원	1998-10-11		102	수정하기	삭제 하기
9905	권혁일	refresh	교수	450 만원	1986-02-11	20 만원	102	수정하기	삭제 하기
9906	이만식	pocari	부교수	420 만원	1988-07-11		101	수정하기	삭제 하기
9907	전은지	totoro	전임강사	210 만원	2001-05-11		101	수정하기	삭제 하기
9908	남은혁	bird13	부교수	400 만원	1990-10-18	17 만원	202	수정하기	삭제 하기
9909	000	dldkdl	부교수	111 만원	2000-11-11	10 만원	102	수정하기	삭제 하기

저장된 데이터 파일

```
{} data.json > ...
              "comm": null,
              "deptno": 101
              "id": 9908,
              "name": "남은혁",
              "userid": "bird13",
              "position": "부교수",
              "sal": 400,
              "hiredate": "1990-10-17T15:00:00.000Z",
              "comm": 17,
              "deptno": 202
              "userid": "dldkdl",
              "position": "조교수",
              "sal": "111",
              "hiredate": "2000-11-11",
              "comm": "100",
              "deptno": "101",
              "id": 9909
```

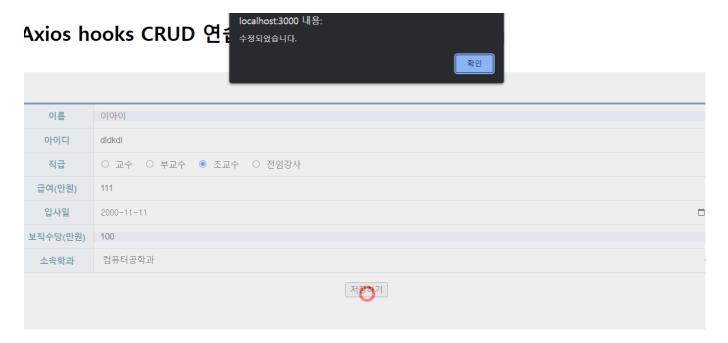
Edit

수정 페이지

Axios hooks CRUD 연습문제



교수 수정시



수정된 목록

교수 등록하기

No.	이름	아이디	직급	급여	입사일	보직수당	소속학과번호	수정	삭제
9901	김도훈	capool	교수	500 만원	1982-06-12	20 만원	101	수정하기	삭제 하기
9902	이재우	sweat413	조교수	320 만원	1995-04-12		201	수정하기	삭제 하기
9903	성연희	pascal	조교수	360 만원	1993-03-17	15 만원	101	수정하기	삭제 하기
9904	염일웅	blue77	전임강사	240 만원	1998-10-11		102	수정하기	삭제 하기
9905	권혁일	refresh	교수	450 만원	1986-02-11	20 만원	102	수정하기	삭제 하기
9906	이만식	pocari	부교수	420 만원	1988-07-11		101	수정하기	삭제 하기
9907	전은지	totoro	전임강사	210 만원	2001-05-11		101	수정하기	삭제 하기
9908	남은혁	bird13	부교수	400 만원	1990-10-18	17 만원	202	수정하기	삭제 하기
9909	0 0 0	dldkdl	조교수	111 만원	2000-11-11	100 만원	101	수정하기	삭제 하기

Delete

삭제 버튼 클릭시



교수 등록하기

No.	이름	아이디	직급	급여	입사일	보직수당	소속학과번호	수정	삭제
9901	김도훈	capool	교수	500 만원	1982-06-12	20 만원	101	수정하기	삭제 하기
9902	이재우	sweat413	조교수	320 만원	1995-04-12		201	수정하기	삭제 하기
9903	성연희	pascal	조교수	360 만원	1993-03-17	15 만원	101	수정하기	삭제 하기
9904	염일웅	blue77	전임강사	240 만원	1998-10-11		102	수정하기	삭제 하기
9905	권혁일	refresh	교수	450 만원	1986-02-11	20 만원	102	수정하기	삭제 하기
9906	이만식	pocari	부교수	420 만원	1988-07-11		101	수정하기	삭제 하기
9907	전은지	totoro	전임강사	210 만원	2001-05-11		101	수정하기	삭제 하기
9908	남은혁	bird13	부교수	400 만원	1990-10-18	17 만원	202	수정하기	삭제 하기
9909	이아이	dldkdl	조교수	111 만원	2000-11-11	100 만원	101	수정하기	삭제 하기

삭제 후 목록

Axios hooks CRUD 연습문제

교수 등록하기

No.	이름	아이디	직급	급여	입사일	보직수당	소속학과번호	수정	삭제
9901	김도훈	capool	교수	500 만원	1982-06-12	20 만원	101	수정하기	삭제 하기
9902	이재우	sweat413	조교수	320 만원	1995-04-12		201	수정하기	삭제 하기
9903	성연희	pascal	조교수	360 만원	1993-03-17	15 만원	101	수정하기	삭제 하기
9904	염일웅	blue77	전임강사	240 만원	1998-10-11		102	수정하기	삭제 하기
9905	권혁일	refresh	교수	450 만원	1986-02-11	20 만원	102	수정하기	삭제 하기
9906	이만식	pocari	부교수	420 만원	1988-07-11		101	수정하기	삭제 하기
9907	전은지	totoro	전임강사	210 만원	2001-05-11		101	수정하기	삭제 하기
9908	남은혁	bird13	부교수	400 만원	1990-10-18	17 만원	202	수정하기	삭제 하기

삭제 후 데이터 파일

```
{} data.json > [ ] student
              "id": 9907,
              "name": "전은지",
              "userid": "totoro",
              "position": "전임강사",
              "sal": 210,
              "hiredate": "2001-05-10T15:00:00.000Z",
              "comm": null,
              "deptno": 101
              "id": 9908,
              "name": "남은혁",
              "userid": "bird13",
              "position": "부교수",
              "sal": 400,
              "hiredate": "1990-10-17T15:00:00.000Z",
              "comm": 17,
              "deptno": 202
```