

## 04장 활용 - React.js 외부 api 활용하기

### 096 서울시 유동 인구 데이터 사용하기 1 - 리스트 페이지 만들기

- 유동 인구 데이터 리스트를 표시할 컴포넌트를 추가한다. 실제 데이터를 불러와 사용하기 전에 데이터를 하드코딩으로 작성하고 미리 페이지 레이아웃을 확인한다.

- App.js 파일

[client/src/components/App.js]

```
01 import React, { Component } from 'react';
02 import { Route } from "react-router-dom";
03
04 // css
05 import '../css/new.css';
06
07 // header
08 import HeaderAdmin from './Header/Header admin';
09
10 // footer
11 import Footer from './Footer/Footer';
12
13 // login
14 import LoginForm from './LoginForm';
15
16 import reactThrottle from './R095_reactThrottle';
17 import floatingPopulationList from './Floating_population/floatingPopulationList';
18
19 class App extends Component {
20   render () {
21     return (
22       <div className="App">
23         <HeaderAdmin/>
24         <Route exact path="/" component={LoginForm} />
25         <Route exact path="/Throttle" component={reactThrottle} />
26         <Route path="/floatPopulationList" component={floatingPopulationList} />
27         <Footer/>
28       </div>
29     );
30   }
31 }
32
33 export default App;
```

- floatingPopulationList.js 파일

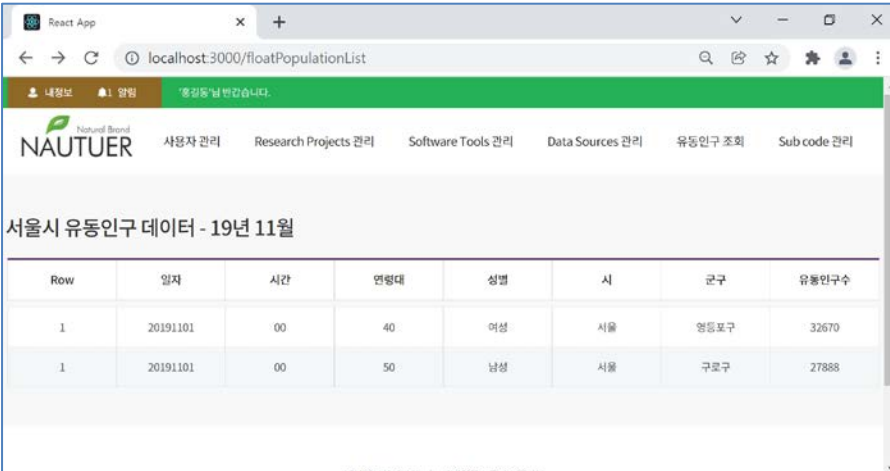
[client/src/components/Floating\_population/floatingPopulationList.js]

```
01 import React, { Component } from 'react';
02
03 class floatingPopulationList extends Component {
04   render () {
05     return (
06       <section class="sub_wrap" >
07         <article class="s_cnt mp_pro_li ct1 mp_pro_li_admin">
08           <div class="li_top">
09             <h2 class="s_tit1">서울시 유동인구 데이터 - 19년 11월</h2>
10           </div>
```

## 리액트 (프론트엔드 개발)

```
11      <div class="list_cont list_cont_admin">
12        <table class="table_ty1 fp_tlist">
13          <tr>
14            <th>Row</th>
15            <th>일자</th>
16            <th>시간</th>
17            <th>연령대</th>
18            <th>성별</th>
19            <th>시</th>
20            <th>군구</th>
21            <th>유동인구수</th>
22          </tr>
23        </table>
24        <table class="table_ty2 fp_tlist">
25          <tr class="hidden_type">
26            <td>1</td>
27            <td>20191101</td>
28            <td>00</td>
29            <td>40</td>
30            <td>여성</td>
31            <td>서울</td>
32            <td>영등포구</td>
33            <td>32670</td>
34          </tr>
35          <tr class="hidden_type">
36            <td>1</td>
37            <td>20191101</td>
38            <td>00</td>
39            <td>50</td>
40            <td>남성</td>
41            <td>서울</td>
42            <td>구로구</td>
43            <td>27888</td>
44          </tr>
45        </table>
46      </div>
47    </article>
48  </section>
49  );
50  }
51  }
52
53  export default floatingPopulationList;
```

### ■ 실행 결과



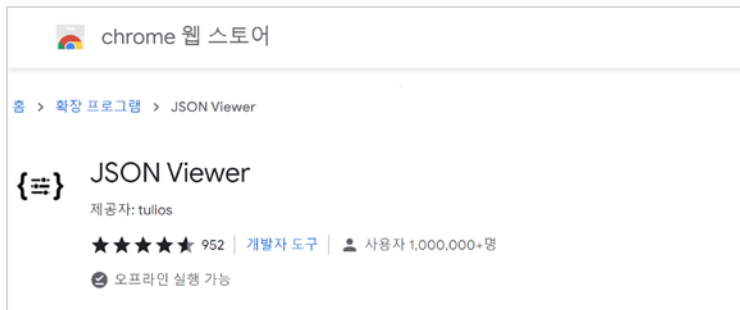
Row	일자	시간	연령대	성별	시	군구	유동인구수
1	20191101	00	40	여성	서울	영등포구	32670
1	20191101	00	50	남성	서울	구로구	27888

## 097 서울시 유동 인구 데이터 사용하기 2 - Open api Key 발급받기

- SK텔레콤 빅 데이터 허브(<https://www.bigdatahub.co.kr>)에서 제공하는 데이터를 사용한다.

## 098 서울시 유동 인구 데이터 사용하기 3 - JSON Viewer로 데이터 확인하기

- 크롬 웹 브라우저에서 가독성 있게 json 데이터를 확인하기 위해 Advanced REST client 혹은 JSON Viewer라는 확장 프로그램을 설치한다.



- JSON Viewer 설치 후 크롬 웹 브라우저에서 아래 url을 호출해보면, 다음과 같이 정렬된 json 데이터를 확인할 수 있다.
  - <http://openapi.seoul.go.kr:8088/746b4762786170703430676e6e4678/json/IotVdata018/1/5/>



**099 서울시 유동 인구 데이터 사용하기 4 - axios로 api 호출하기**

- axios 함수를 사용해 get 방식으로 데이터를 호출하고 반복문을 사용해 행 단위 리스트로 생성한다.
- App.js 파일

[client/src/components/App.js]

```

01 import React, { Component } from 'react';
02 import { Route } from "react-router-dom";
03
04 // css
05 import '../css/new.css';
06
07 // header
08 import HeaderAdmin from './Header/Header admin';
09
10 // footer
11 import Footer from './Footer/Footer';
12
13 // login
14 import LoginForm from './LoginForm';
15
16 import reactThrottle from './R095_reactThrottle';
17 import floatingPopulationList from './Floating_population/floatingPopulationList';
18
19 class App extends Component {
20   render () {
21     return (
22       <div className="App">
23         <HeaderAdmin/>
24         <Route exact path="/" component={LoginForm} />
25         <Route exact path="/Throttle" component={reactThrottle} />
26         <Route path="/floatPopulationList" component={floatingPopulationList} />
27         <Footer/>
28       </div>
29     );
30   }
31 }
32
33 export default App;

```

- floatingPopulationList.js 파일

[client/src/components/Floating\_population/floatingPopulationList.js]

```

01 import React, { Component } from 'react';
02 import axios from "axios";
03
04 class floatingPopulationList extends Component {
05   constructor(props) {
06     super(props);
07
08     this.state = {
09       responseFPList: '',
10       append_FPList: '',
11     }

```

```

12     }
13
14     componentDidMount() {
15         this.callFloatPopullListApi()
16     }
17
18     callFloatPopullListApi = async () => {
19
20     axios.get('http://openapi.seoul.go.kr:8088/746b4762786170703430676e6e4678/json/IotVdata018/1/5/', {
21         })
22         .then( response => {
23             try {
24                 this.setState({ responseFPList: response });
25                 this.setState({ append_FPList: this.FloatPopullListAppend() });
26             } catch (error) {
27                 alert(error)
28             }
29         })
30         .catch( error => {alert(error);return false;} );
31     }
32
33     FloatPopullListAppend = () => {
34         let result = []
35         var FPList = this.state.responseFPList.data
36         var jsonString = JSON.stringify(FPList)
37         //jsonString = jsonString.replace(/\(1시간단위\)/g, '')
38         //jsonString = jsonString.replace(/\(10세단위\)/g, '')
39         var json = JSON.parse(jsonString)
40
41         for(let i=0; i<json.IotVdata018.row.length; i++){
42             var data = json.IotVdata018.row[i]
43             var idx = i+1
44             result.push(
45                 <tr class="hidden_type">
46                     <td>{idx}</td>
47                     <td>{data.REGIST_DT}</td>
48                     <td>{data.ORGAN_NM}</td>
49                     <td>{data.TRNSMIT_SERVER_NO}</td>
50                     <td>{data.DATA_NO}</td>
51                     <td>{data.COLUMN0}</td>
52                     <td>{data.COLUMN1}</td>
53                     <td>{data.COLUMN6}</td>
54                 </tr>
55             )
56         }
57         return result
58     }
59
60     render () {
61         return (
62             <section class="sub_wrap" >
63                 <article class="s_cnt mp_pro_li ct1 mp_pro_li_admin">
64                     <div class="li_top">
65                         <h2 class="s_tit1">서울시 유동인구 데이터 - 19년 11월</h2>
66                     </div>
67                     <div class="list_cont list_cont_admin">
68                         <table class="table_ty1 fp_tlist">
69                             <tr>
70                                 <th>Row</th>
71                                 <th>일자</th>
72                                 <th>시간</th>
73                                 <th>연령대</th>
74                                 <th>성별</th>
75                                 <th>시</th>
76                                 <th>군구</th>

```

## 리액트 (프론트엔드 개발)

```
77         <th>유동인구수</th>
78     </tr>
79 </table>
80 <table class="table_ty2 fp_tlist">
81     {this.state.append_FPList}
82 </table>
83 </div>
84 </article>
85 </section>
86 );
87 }
88 }
89
90 export default floatingPopulationList;
```

### ■ 실행 결과

React App

localhost:3000/floatPopulationList

내정보 1 알림 '홍길동'님 반갑습니다.

NAUTUER

사용자 관리

Research Projects 관리

Software Tools 관리

Data Sources 관리

유동인구 조회

Sub code 관리

서울시 유동인구 데이터 - 19년 11월

Row	일자	시간	연령대	성별	시	군구	유동인구수
1	2021-12-18 23:57:52.0	서울시	32	1	SDOT001	4025	2
2	2021-12-18 23:57:48.0	서울시	32	1	SDOT001	4042	0
3	2021-12-18 23:57:42.0	서울시	32	1	SDOT001	4040	1
4	2021-12-18 23:57:28.0	서울시	32	1	SDOT001	4007	3

### [꿀팁] axios.get() 사용할 때 CORS 에러 해결

#### ■ 참고

- <https://developer-talk.tistory.com/91>

#### ■ 해결

```
# package.json 파일의 일부
...(생략)...
},
"proxy": "http://127.0.0.1:8000/"
}

# axios.get() 함수의 예
...(생략)...
callFloatPopullistApi = async () => {
    axios.get('/api/user/', {})
    .then( response => {
        //console.log(response.data)
    })
}
```

```

        try {
            this.setState({ responseFPList: response });
            this.setState({ append_FPList: this.FloatPopullListAppend() });
        } catch (error) {
            alert(error)
        }
    })
    ...(<생략>)...

```

## 100 recharts로 LineChart 구현하기 1

- 유동 인구 데이터 리스트를 표시할 컴포넌트를 추가한다. 실제 데이터를 불러와 사용하기 전에 데이터를 하드코딩으로 작성하고 미리 페이지 레이아웃을 확인한다.

```

// recharts를 설치한다.
D:\dev\workspace\react\client>yarn add recharts

```

- App.js 파일

```

[client/src/components/App.js]

01 import React, { Component } from 'react';
02 import { Route } from "react-router-dom";
03
04 // css
05 import '../css/new.css';
06
07 // header
08 import HeaderAdmin from './Header/Header admin';
09
10 // footer
11 import Footer from './Footer/Footer';
12
13 // login
14 import LoginForm from './LoginForm';
15
16 import reactThrottle from './R095_reactThrottle';
17 import floatingPopulationList from './Floating_population/floatingPopulationList';
18 import rechartsSimpleLineChart from './Floating_population/rechartsSimpleLineChart';
19
20 class App extends Component {
21   render () {
22     return (
23       <div className="App">
24         <HeaderAdmin/>
25         <Route exact path="/" component={LoginForm} />
26         <Route exact path="/Throttle" component={reactThrottle} />
27         <Route path="/floatPopulationList" component={floatingPopulationList} />
28         <Route path="/rechartsSimpleLineChart" component={rechartsSimpleLineChart} />
29         <Footer/>
30       </div>
31     );
32   }
33 }
34
35 export default App;

```

#### ■ rechartsSimpleLineChart.js 파일

- 02~04: recharts 패키지를 임포트해 관련 태그를 사용할 수 있도록 한다.
- 06~12: 군구별, 유동 인구 수와 비유동 인구 수를 비교하기 위해 json 형태의 데이터를 세팅한다.
- 19~24: <LineChart> 태그에 화면에 표시할 차트 영역의 가로 길이(width), 세로 길이(height), 데이터(data), margin 값을 할당한다.
- 25: 차트 내부에 표시되는 격자선 간격을 조정할 수 있다.
- 26: data 변수에 할당한 데이터 중 X축에 사용할 데이터의 key 값을 지정한다.
- 27: <Tooltip> 태그는 마우스가 차트로 이동했을 때 이동한 좌표의 데이터를 화면에 나타낸다.
- 28: <Legend> 태그는 차트 하단 범례를 영역에 표시한다.
- 29~30: Y축에 표현될 데이터 key 값과 라인색을 지정한다. activeDot의 r 값은 마우스 커서가 차트로 이동했을 때 나타나는 색이 채워지는 동그라미의 크기다.

[client/src/components/Floating\_population/rechartsSimpleLineChart.js]

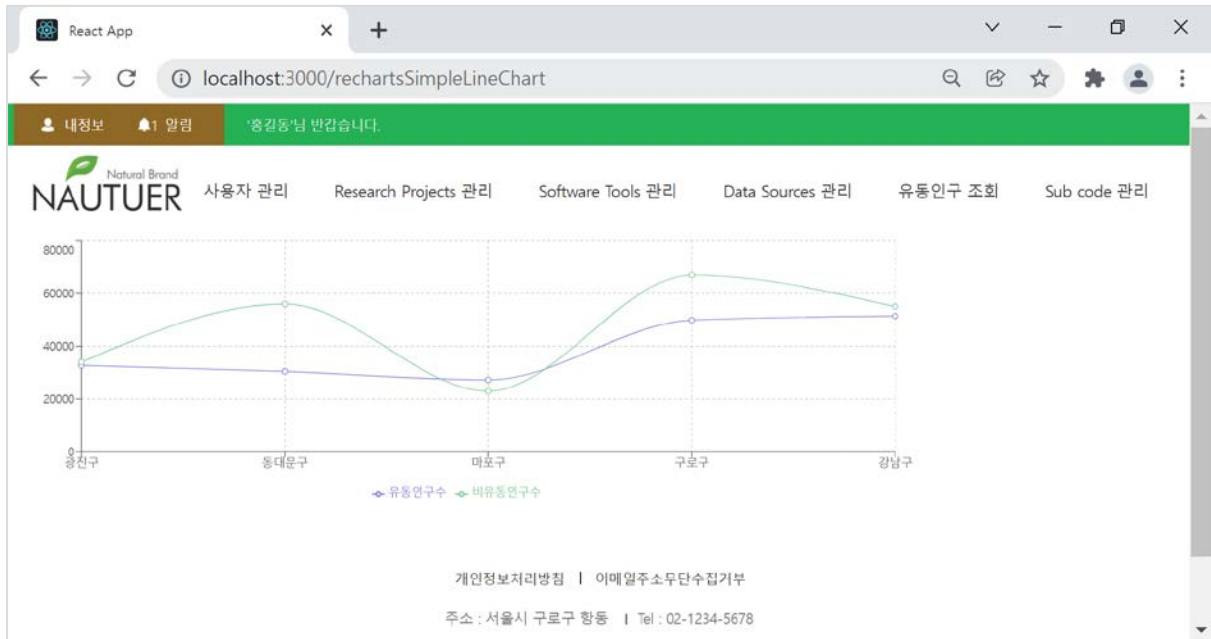
```

01 import React, { PureComponent } from 'react';
02 import {
03   LineChart, Line, XAxis, YAxis, CartesianGrid, Tooltip, Legend,
04 } from 'recharts';
05
06 const data = [
07   { 군구: '광진구', 유동인구수: 32760, 비유동인구수: 34000 },
08   { 군구: '동대문구', 유동인구수: 30480, 비유동인구수: 56000 },
09   { 군구: '마포구', 유동인구수: 27250, 비유동인구수: 23000 },
10   { 군구: '구로구', 유동인구수: 49870, 비유동인구수: 67000 },
11   { 군구: '강남구', 유동인구수: 51420, 비유동인구수: 55000 },
12 ];
13
14 export default class rechartsSimpleLineChart extends PureComponent {
15   static jsfiddleUrl = 'https://jsfiddle.net/alidingling/xqjtetw0/';
16
17   render() {
18     return (
19       <LineChart
20         width={1000}
21         height={300}
22         data={data}
23         margin={{ top: 5, right: 30, left: 20, bottom: 5 }}
24       >
25         <CartesianGrid strokeDasharray="3 3" />
26         <XAxis dataKey="군구" /><YAxis />
27         <Tooltip />
28         <Legend />
29         <Line type="monotone" dataKey="유동인구수" stroke="#8884d8" activeDot={{ r: 8 }} />
30         <Line type="monotone" dataKey="비유동인구수" stroke="#82ca9d" />
31       </LineChart>
32     );
33   }
34 }

```

#### ■ 실행 결과





- 101 recharts로 LineChart 구현하기 2
- 102 recharts로 AreaChart 구현하기
- 103 recharts로 BarChart 구현하기
- 104 recharts로 ComposedChart 구현하기
- 105 recharts로 ScatterChart 구현하기