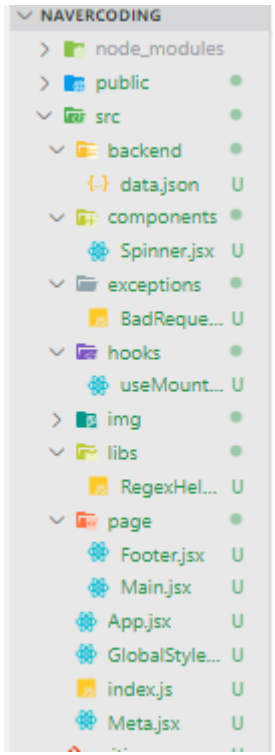
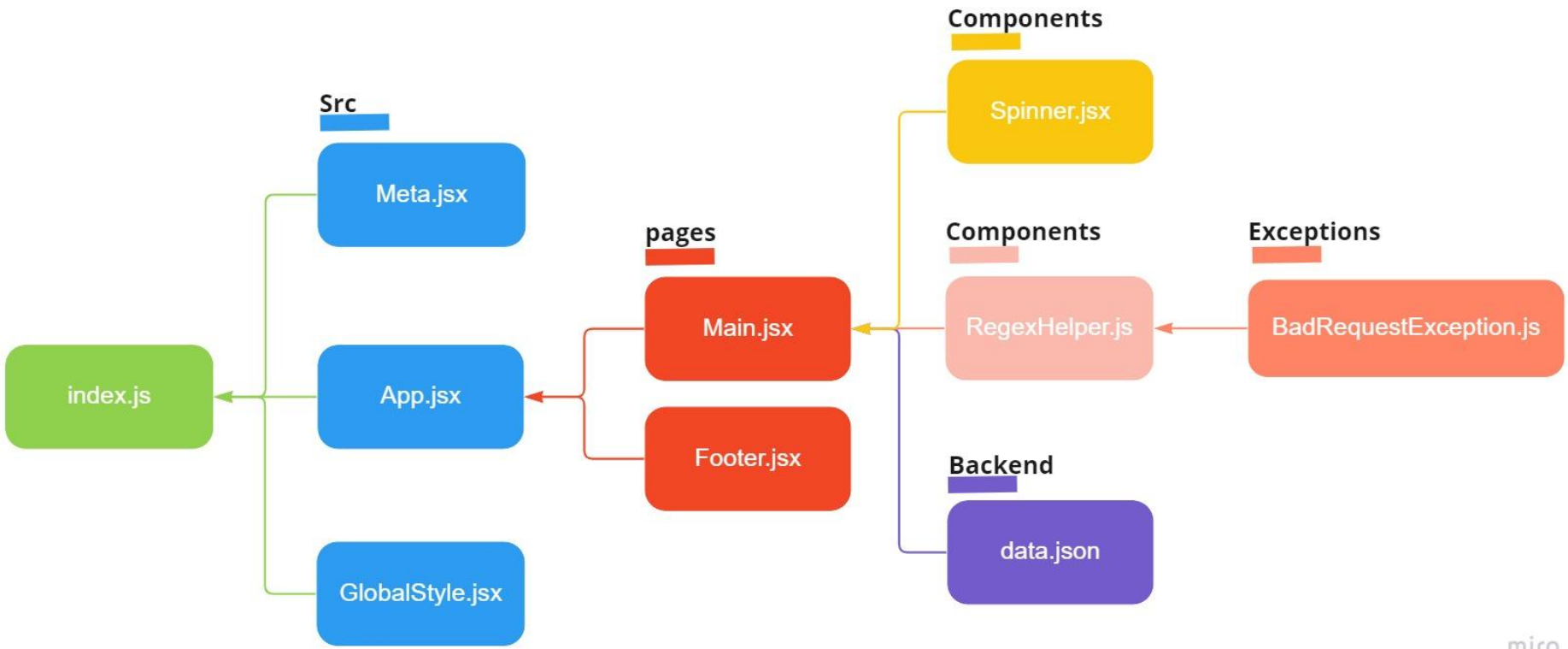
NAVER

2022.05.23 이승아

INDEX

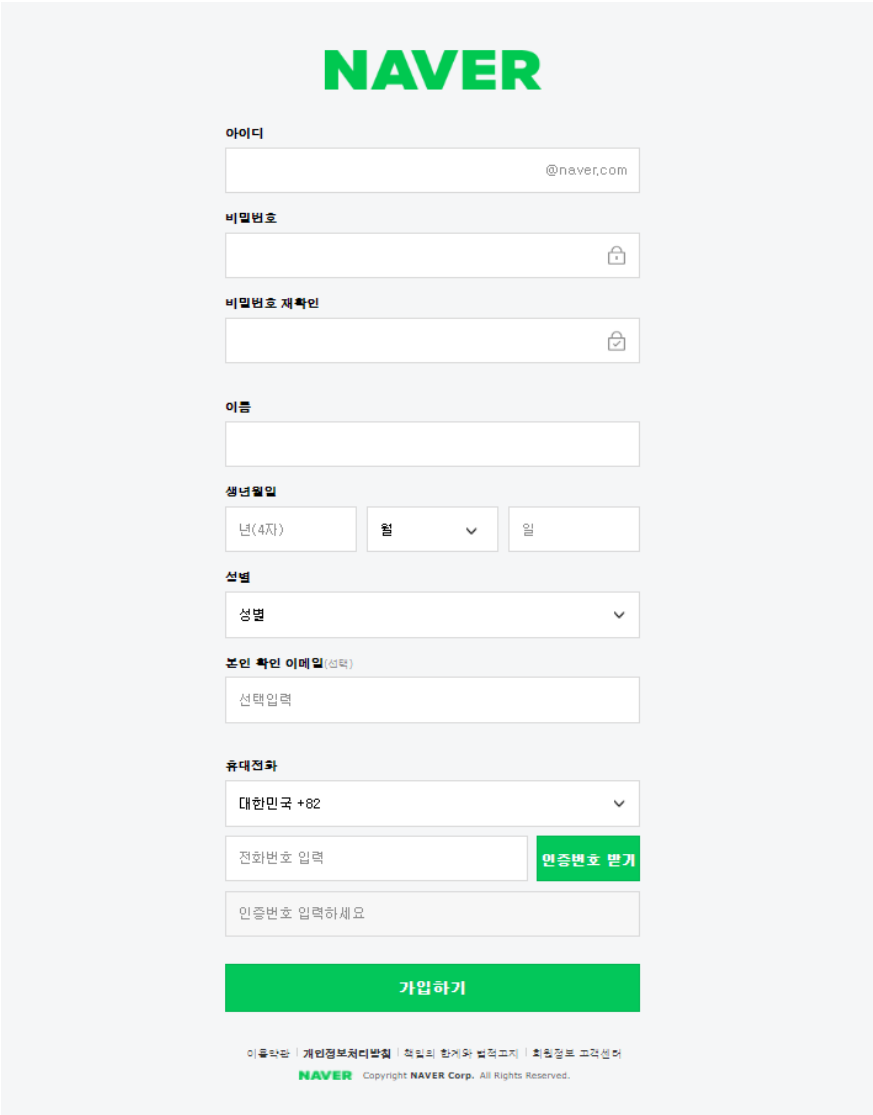
컴포넌트 구조


NAVERCODING
node_modules
public
src
backend
data.json U
components
Spinner.jsx U
exceptions
BadReque... U
hooks
useMount... U
img
libs
RegexHel... U
page
Footer.jsx U
Main.jsx U
App.jsx U
GlobalStyle... U
index.js U
Meta.jsx U

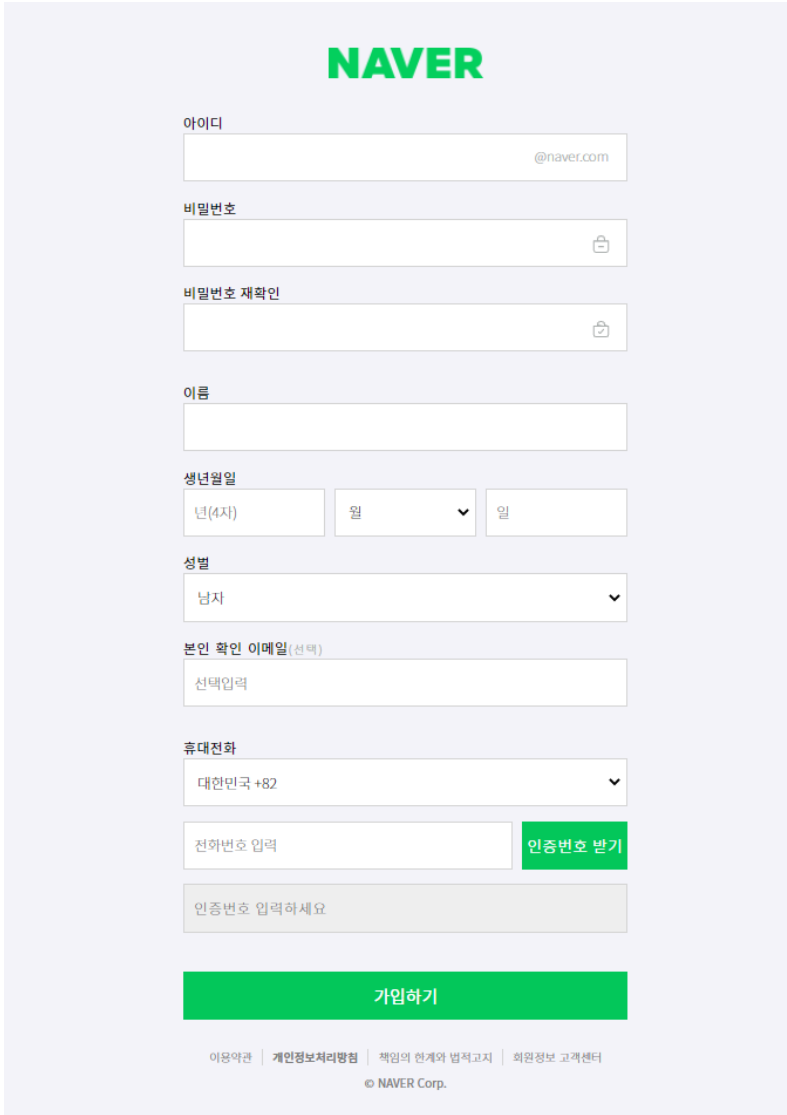

Components
Spinner.jsx
Src
Meta.jsx
pages
Components
Exceptions
Main.jsx
RegexHelper.js
BadRequestException.js
index.js
App.jsx
Footer.jsx
Backend
data.json
GlobalStyle.jsx
miro

전체 결과 비교

원본페이지



구현페이지



세부 페이지 결과

네이버 로고

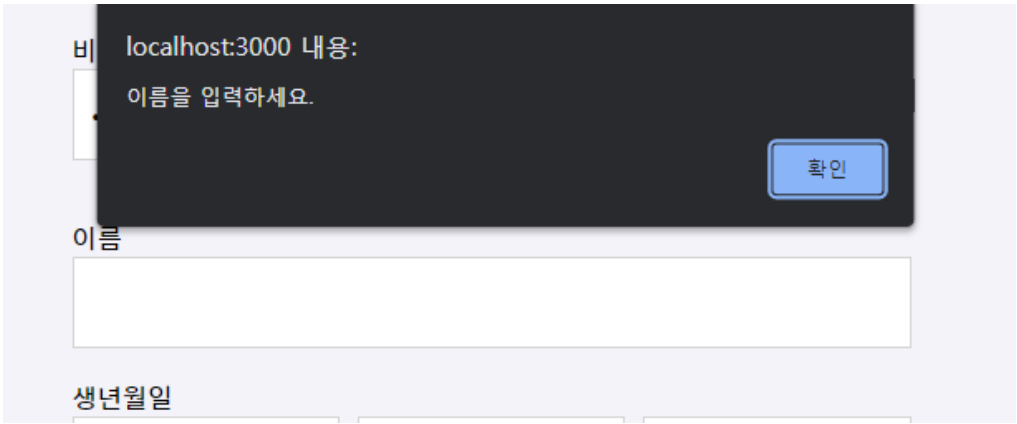
NAVER

아이디 구현

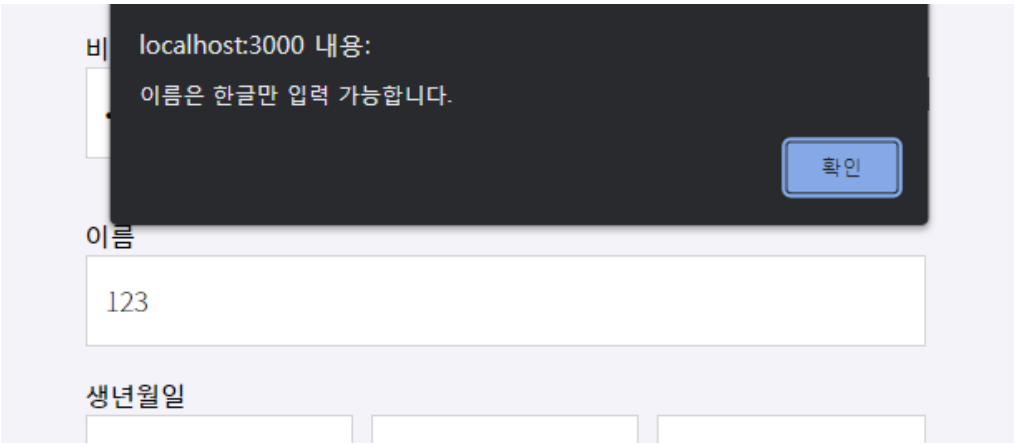
아이디

@naver.com

정규표현식 검사

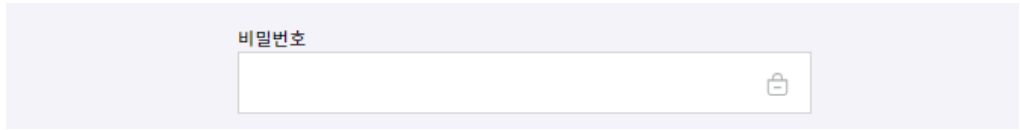


정규표현식 검사

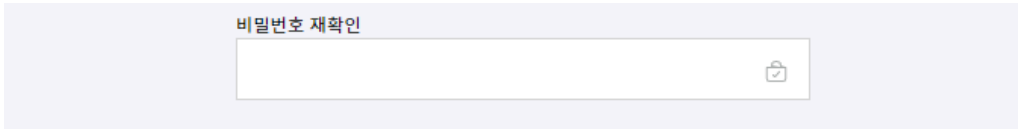


세부 페이지 결과

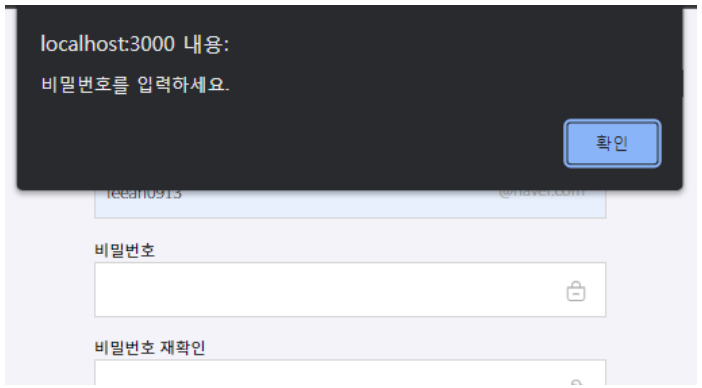
비밀번호 구현



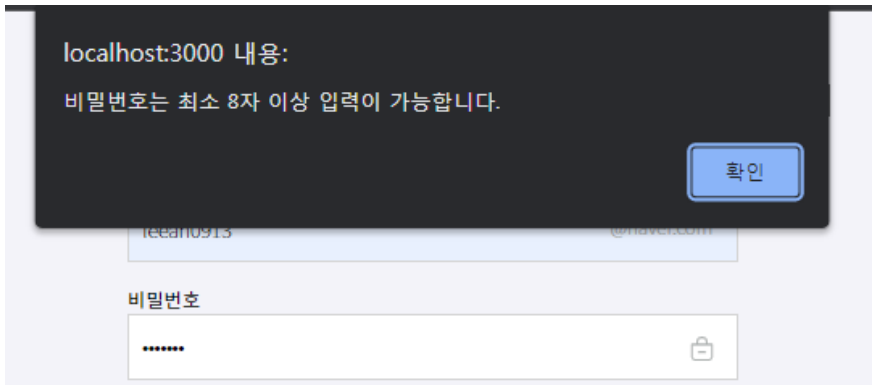
비밀번호 재확인 구현



정규표현식 검사



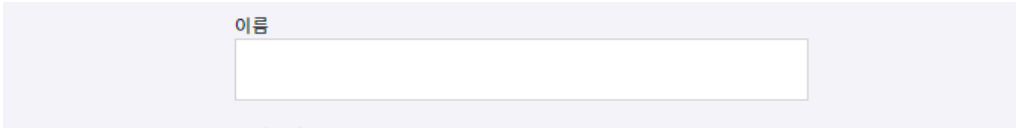
정규표현식 검사



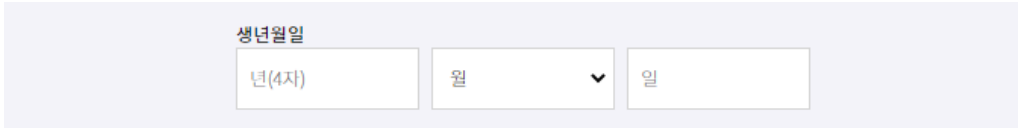
정규표현식 검사

세부 페이지 결과

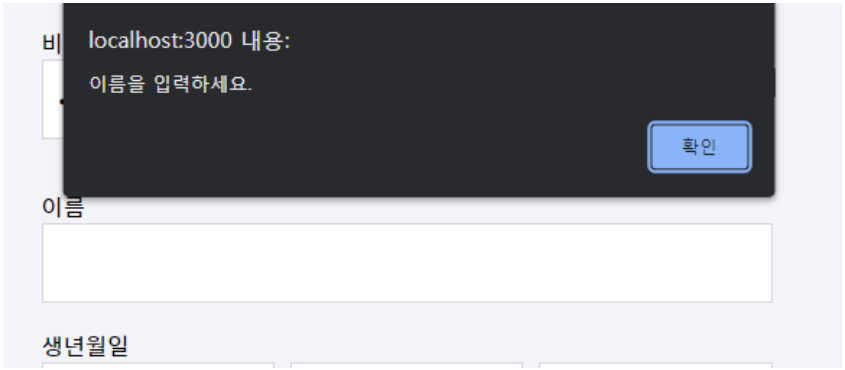
이름 구현

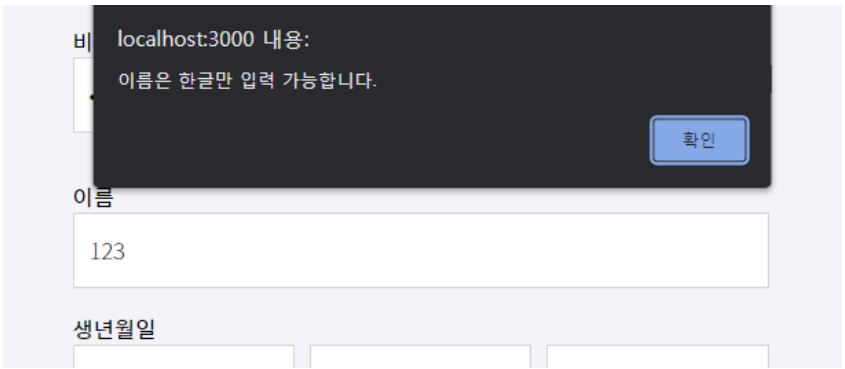


생년월일 구현

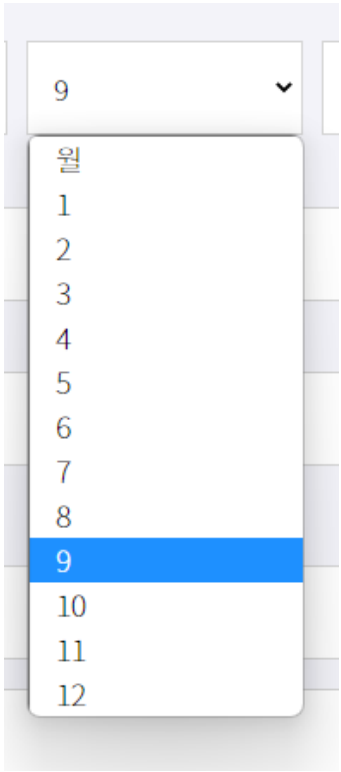


정규표현식 검사

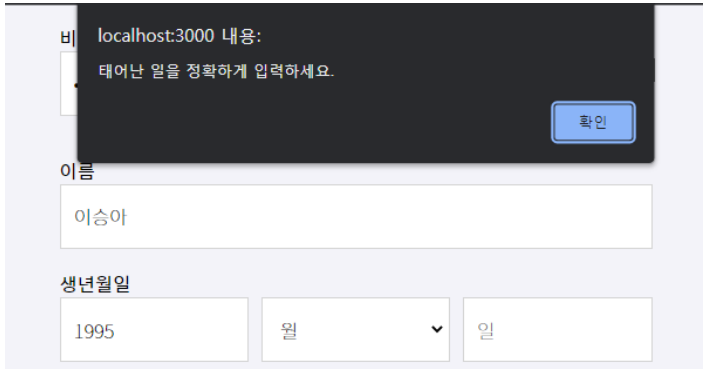


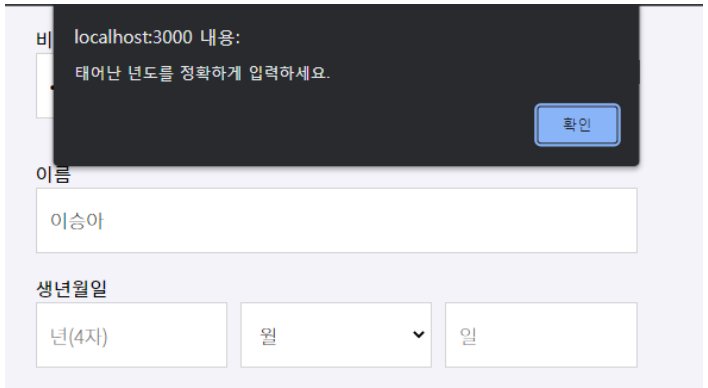


정규표현식 검사



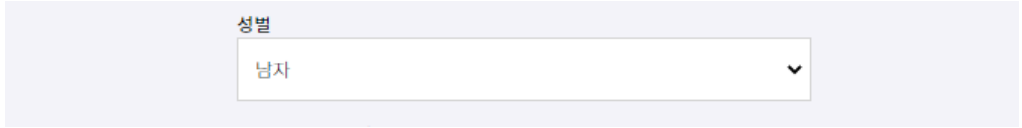
정규표현식 검사





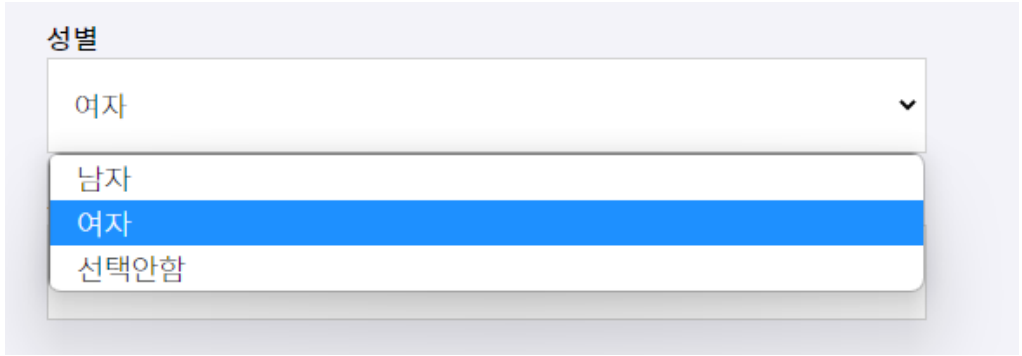
세부 페이지 결과

성별 구현

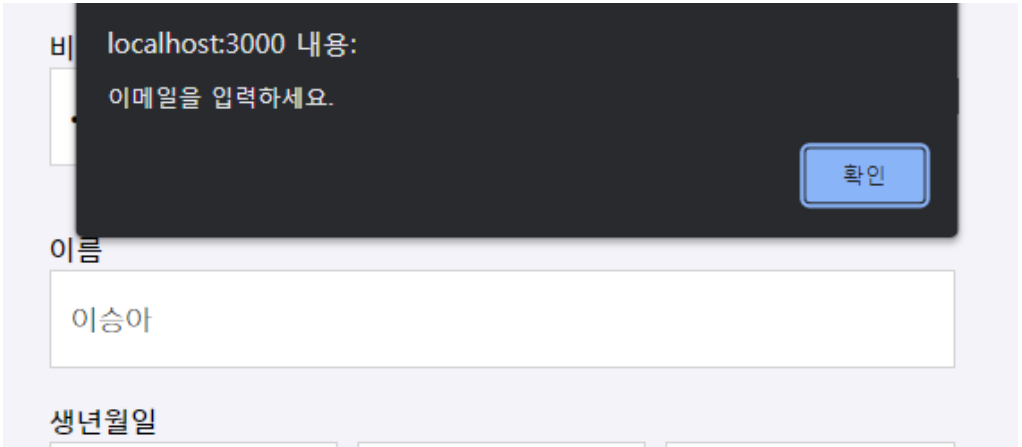


이메일 구현

정규표현식 검사

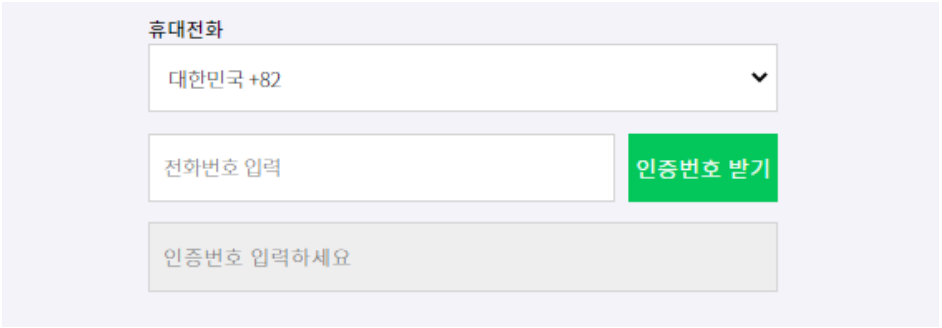


정규표현식 검사

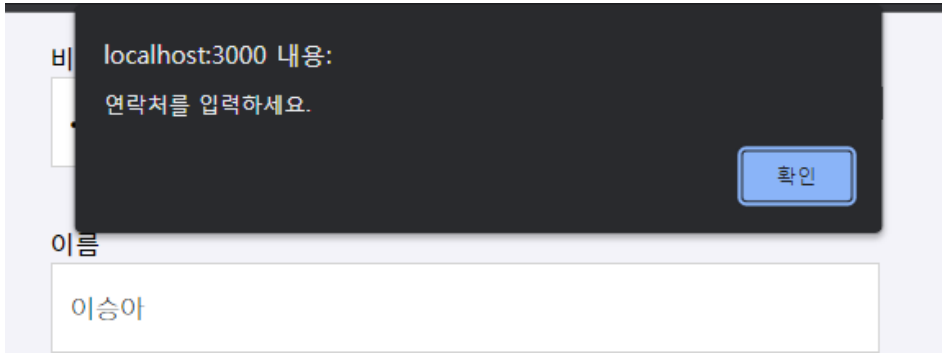


세부 페이지 결과

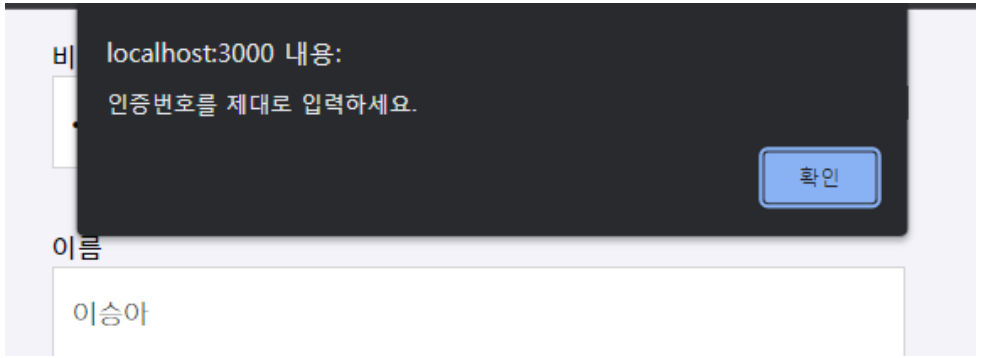
휴대전화 구현



정규표현식 검사

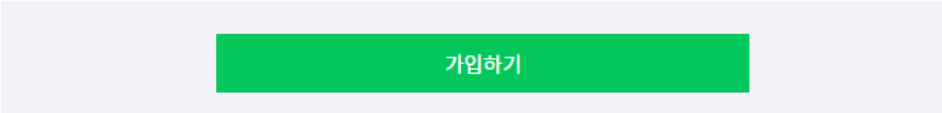


정규표현식 검사



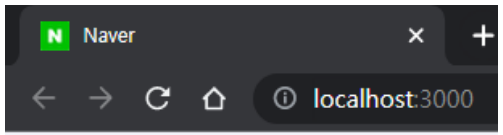
세부 페이지 결과

가입 버튼 구현



FOOTER 구현

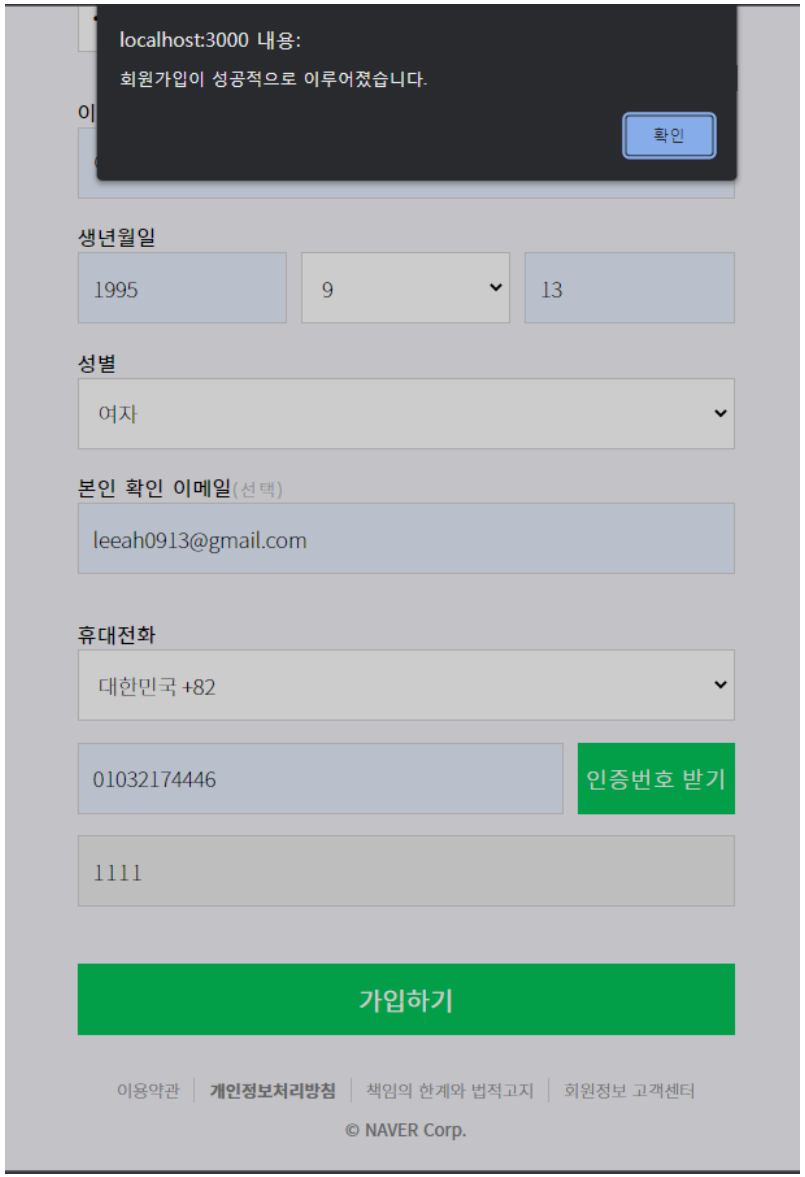
아이콘 변경



Member 데이터 파일로 전송

```
[
  {
    "userId": "leeah0913",
    "password": 12345678,
    "userRePw": 12345678,
    "name": "이승아",
    "birthYear": "1995",
    "birthMonth": "9",
    "birthDay": "13",
    "gender": "여자",
    "email": "leeah0913@gmail.com",
    "selectTel": "대한민국 +82",
    "tel": 1032174446,
    "certification": 1111,
    "id": 1
  }
]
```

가입 성공 알람



App.jsx

```
import React from 'react'
import styled from 'styled-components';
import Main from './page/Main';
import logo from './img/logo.png';
import Footer from './page/Footer';

const NavContainer = styled.nav`
  text-align: center;
  margin: auto;
  height: 100px;
  .mainLogo{
    width: 160px;
    margin: 50px 0;
    cursor: pointer;
  }

`;

const App = () => {
  return(
    <>
    <NavContainer>
      <div className='text-blind'>NAVER</div>
      <img src={logo} alt="mainLogo" className="mainLogo" />
    </NavContainer>
    <Main/>
    <Footer/>
    </>
  )
}

export default React.memo(App);
```

Index.js

```
import React from 'react';
import ReactDOM from 'react-dom/client';
import App from './App';
import { BrowserRouter } from 'react-router-dom';
import Meta from './Meta';
import GlobalStyle from './GlobalStyle';

const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(
  <React.StrictMode>
    <Meta/>
```

```
    <GlobalStyle/>
    <BrowserRouter>
      <App />
    </BrowserRouter>
  </React.StrictMode>
);
```

Meta.jsx

```
/**
 * @filename: Meta.jsx
 * @description <head>태그 내의 SEO 처리 및 기본 참조 리소스 명시
 * @author: Seunga Lee(leeah0913@gmail.com)
 */
import React from 'react';
// SEO 처리기능 패키지
import { Helmet,HelmetProvider } from 'react-helmet-async';

/** SEO 처리 컴포넌트
 * @param props
 * @returns{JSX.Element}
 */

const Meta = (props) => {
  return (
      <div>
        <HelmetProvider>
            <Helmet>
                <meta charset='utf-8'/>
                <title>{props.title}</title>
                {/* SEO태그 */}
                <meta name='description' content={props.description} />
                <meta name='keywords' content={props.keywords} />
                <meta name='author' content={props.author} />
                <meta property='og:type' content='website' />
                <meta property='og:title' content={props.title} />
                <meta property='og:description' content={props.description} />
                <meta property='og:url' content={props.url} />

                <link rel="preconnect" href="https://fonts.googleapis.com"/>
               <link rel="preconnect" href="https://fonts.gstatic.com"
crossorigin/>
               <link href="https://fonts.googleapis.com/css2?
family=Noto+Sans+KR:wght@100;300;400;500;700;900&display=swap" rel="stylesheet"/>
            </Helmet>
        </HelmetProvider>
        </div>
  );
};
```

```
Meta.defaultProps={
    title:"Naver",
    description: 'Naver clone by ReactJs',
    keywords:'React',
    author:'Seunga Lee',
    url: window.location.href
};

export default Meta;
```

GlobalStyle.jsx

```
/**
 * @filename: GlobalStyles.js
 * @description: 전역으로 적용될 기본 스타일시트.
 *               이 파일에서 정의한 class는 ReaxtJSX에서 className 속성으로 참조한다.
 * @author:Seunga Lee(leeah0913@gmail.com)
 */

import { createGlobalStyle } from 'styled-components';

/**
 * 전역 스타일 시트를 정의한 객체
 * @type {GlobalStyleComponent<{}, DefaultTheme>}
 */
const GlobalStyle = createGlobalStyle`

   *{
        font-family: 'Noto Sans KR', sans-serif;
        margin: 0;
        padding: 0;
        width: 100%;
        box-sizing: border-box;

        body{
        height: 100%;
        margin: auto;
       width: 460px;
       display: block;
       background-color: #f3f3f9;
        }
        button{
            border: none;
            cursor: pointer;
            background-color: #03c75a;
               color: white;
        }
        ol, ul {
        list-style: none;
        }
        hr{
```

```
        border: none;
    }

    option{
        font-weight: 100;
        padding-left: 50px;

    }

.center{
    margin: 15px 0;
}
    .title{
    font-size: 14.5px;
    font-weight: 400;
    text-align: left;
    letter-spacing: 0.5px;

    }
    .inputBox{
    margin: auto;
    height: 50px;
    border: 1px solid #d5d5d5;
    padding-left: 10px;
    font-weight: 100;
    font-size: 15px;
    cursor: pointer;
    }

    p{
        display: inline;
        color:#ADB1B0;
        font-weight: 100;
        width: 460px;
        text-align: left;
    }
        .icon{
        position: absolute;
        margin: left;
        text-align: left;
        font-size: 25px;
        font-weight: 100;
        width: 460px;
        }



.text-blind{
    overflow: hidden;
    display: inline-block;
    position: relative;
    z-index: -1;
    border: 0;
    width: 1px;
```

```
            height: 1px;
            clip: rect(1px, 1px, 1px, 1px);
            clip-path: inset(50%);
        }

    }

`;

export default GlobalStyle;
```

Main.jsx

```
import React from 'react'
import useAxios from 'axios-hooks';
import styled from 'styled-components';
import regexHelper from '../libs/RegexHelper';
import Spinner from '../components/Spinner';
import { IoBagCheckOutline,IoBagRemoveOutline } from "react-icons/io5";

/* 회원가입 전체 폼 */
const MainContainer=styled.div`
 width: 100%;
 height: 100%;
 cursor: pointer;

`;

/* 아이디 */
const IdSection=styled.div`
  .id_form{
    position: relative;
    p{
      position: absolute;
      top: 27%;
      left: 79%;
      font-size: 14px;
    }
  }
`;

/* 비밀번호 */
const PwSection=styled.div`
.pw_form{
  position: relative;
    p{
      position: absolute;
      top: 27%;
      left: 44%;
      font-size: 20px;
```

```
    }
  }
`;

/* 비밀번호 재확인 */
const RepwSection=styled.div`
.repw_form{
  position: relative;
    p{
      position: absolute;
      top: 27%;
      left: 44%;
      font-size: 20px;

    }
  }
`;


/* 생년월일 */
const BirthSection=styled.div`
width: 100%;
  .birthBoxs{
    display: flex;
    justify-content:space-between;
    .birthBox{
      width: 32%;
      .birthItem{
      height: 50px;
      border: 1px solid #d5d5d5;
      font-weight: 100;
      font-size: 15px;
      padding: 0 10px;
      cursor: pointer;
      }
    }
  }


`;



/* 이메일 */
const EmailSection=styled.div`
word-spacing: 2px;
  label>p{
    font-size: 12px;
    letter-spacing: 2px;
  }
`;

/* 국가번호 */
const CountryTelSection=styled.div`
```

```
`;

/* 휴대폰 번호 */
const TelSection=styled.div`
display: flex;
justify-content: space-between;

.telBox{
  width: 80%;
  padding-right: 10px;

  input{
    border: 1px solid #d5d5d5;
    height: 50px;
    padding-left: 10px;
    align-items: center;
    font-weight: 100;
    font-size: 15px;
  }
}
.buttonBox{
  width: 25%;
  button{
    height: 50px;
    font-size:16px;
    padding: 5px;
    font-weight: 300;
    letter-spacing: 1px;

  }
}
`;

/* 인증번호 */
const CertificationSection=styled.div`
height: 50px;
input{
  background-color: #eee;
  letter-spacing: 1px;
}
`;

/* 가입하기 버튼 */
const JoinBox = styled.div`
width: 460px;
margin: 25px 0;
button{
  height: 50px;
  font-size: 18px;
}
`;
const Main = () => {
  /* 월 option 반복문을 위한 배열 생성 */
```

```
    const month=[1,2,3,4,5,6,7,8,9,10,11,12];

    const [{loading}, refetch] = useAxios({
      url: "http://localhost:3001/memeber",
      method: 'POST'
      }, {manual: true});

      /** <form>의 submit 버튼이 눌러졌을 때 호출될 이벤트 핸들러 */
      const onSubmit = React.useCallback((e)=>{
      e.preventDefault();

     //이벤트가 발생한 폼 객체
      const current = e.target;

      //입력값 유효성 검사
        try{
          /* 아이디 검사 */
          regexHelper.value(current.userId, '아이디를 입력하세요.')
          regexHelper.minLength(current.userId, 5, '아이디는 최소 5자 이상 입력 가능
합니다.');
          regexHelper.maxLength(current.userId, 20, '아이디는 최대 20자 까지만 입력
가능합니다.');
          regexHelper.engNum(current.userId, '아이디는 영어와 숫자만 입력 가능합니
다.');

          /* 비밀번호 검사 */
          regexHelper.value(current.password, '비밀번호를 입력하세요.');
          regexHelper.minLength(current.password, 8, '비밀번호는 최소 8자 이상 입력
이 가능합니다.');
          regexHelper.maxLength(current.password, 16, '비밀번호는 회대 16자 까지만
입력 가능합니다.');
          regexHelper.engNum(current.password, '비밀번호는는 영어와 숫자 특수기호를
사용하세요.');
          regexHelper.compareTo(current.userRePw, current.password, '비밀번호가 일
치하지 않습니다.');

          /* 이름 검사 */
          regexHelper.value(current.name, '이름을 입력하세요.');
          regexHelper.minLength(current.name, 2, '이름은 최소 2자 이상 입력 가능합니
다.');
          regexHelper.maxLength(current.name,20, '이름은 최대 20자 까지만 입력 가능
합니다.');
          regexHelper.korEng(current.name, '이름은 한글만 입력 가능합니다.')

          /* 생년월일 검사 */
          regexHelper.value(current.birthYear, '태어난 년도를 정확하게 입력하세
요.');
          regexHelper.maxLength(current.birthYear, 4, '태어난 년도 4자리를 정확하게
입력하세요.');
          regexHelper.value(current.birthMonth,'태어난 월을 선택해주세요.')
          regexHelper.value(current.birthDay,'태어난 일을 정확하게 입력하세요.');
          regexHelper.maxLength(current.birthDay, 2, '태어난 일 2자리를 정확하게 입
력하세요.');
```

```
            /* 이메일 검사 */
            regexHelper.value(current.email, '이메일을 입력하세요.');
            regexHelper.email(current.email, '이메일 주소가 잘못되었습니다.');

            /* 전화번호 검사 */
            regexHelper.value(current.selectTel, '국가번호를 선택하세요.');
            regexHelper.value(current.tel, '연락처를 입력하세요.');
            regexHelper.phone(current.tel, '연락처가 잘못 되었습니다.');

            /* 인증번호 검사 */
            regexHelper.maxLength(current.certification, 4, '인증번호를 제대로 입력하
세요.');

          }catch (e) {
            window.alert(e.message);
            e.field.focus();
            return;
          }

          let json = null;

          (async () => {
            try{
              const response = await refetch({
                data:{
                userId: current.userId.value,
                password: parseInt(current.password.value),
                userRePw: parseInt(current.userRePw.value),
                name: current.name.value,
                birthYear: current.birthYear.value,
                birthMonth: current.birthMonth.value,
                birthDay: current.birthDay.value,
                gender: current.gender.value,
                email: current.email.value,
                selectTel:current.selectTel.value,
                tel:parseInt(current.tel.value),
                certification:parseInt(current.certification.value)
                }
              });
            json = response.data;
            }catch(e){
            console.error(e);
            window.alert(`[${e.response.status}]
            ${e.response.statusText}\n${e.message}`);
              };

            /* async()(); 후 한 줄 띄우기 */
              if(json !== null){
              window.alert('회원가입이 성공적으로 이루어졌습니다.');
              }
          })();
          },[refetch]);

        return (
```

```
        <>
        <Spinner visible={loading}/>
        <form onSubmit={onSubmit}>
          {/* 메인 전체 폼 */}
          <MainContainer className='mainContainer'>

            {/* 아이디 */}
            <IdSection className='center'>
              <label className="title" htmlFor="userId">아이디</label>
              <div className='id_form'>
                <input type="text" name="userId" className="field inputBox"/>
                <p>@naver.com</p>
              </div>

            </IdSection>

          {/* 비밀번호 */}
            <PwSection className='center'>
            <label className="title" htmlFor="password">비밀번호</label>
              <div className='pw_form'>
                <input type="password" name="password" className="field
inputBox"/>
                <p><IoBagRemoveOutline /></p>
              </div>


            </PwSection>

          {/* 비밀번호 재확인 */}
            <RepwSection className='center'>
            <label className="title" htmlFor="userRePw">비밀번호 재확인</label>
              <div className='repw_form'>
                <input type="password" name="userRePw" className="field
inputBox"/>
                <p><IoBagCheckOutline /></p>
              </div>
            </RepwSection>
            <hr/>

          {/* 이름 */}
            <div className='center'>
            <label className="title" htmlFor="name">이름</label>
              <div className='name_form'>
                <input type="text" name="name" className="field inputBox"/>
              </div>
            </div>

          {/* 생년월일 */}
            <BirthSection className='center'>
              <label htmlFor="birthYear" className='title'>생년월일</label>
              <div className='birthBoxs'>
              <div className='birthBox'>
                <input type='text' name='birthYear' className='field birthItem'
placeholder='년(4자)'/>
```

```
              </div>
              {/* 월 선택을 위한 반복문 */}
              <div className='birthBox'>
              <select name='birthMonth' className='field birthItem'>
                <option placeholder='월'>월</option>
                {month.map((v,i)=>{
                  return(<option value={v} key={i}>{v}</option>)
                })}
              </select>
              </div>
              <div className='birthBox'>
                <input type='text' name='birthDay' className='field birthItem'
placeholder='일'/>
              </div>
              </div>
            </BirthSection>

            {/* 성별 */}
            <div className='center'>
              <span className='title'>성별</span>
              <select name="gender" className='field inputBox' placeholder='성별'
>
                <option>남자</option>
                <option>여자</option>
                <option>선택안함</option>
              </select>
            </div>

            {/* 이메일 */}
            <EmailSection className='center'>
            <label className="title" htmlFor="email">본인 확인 이메일<p>(선택)</p>
</label>
              <div className='pw_form'>
                <input type="text" name="email" className="field inputBox"
placeholder='선택입력'/>
              </div>
            </EmailSection>
            <hr/>

            {/* 국가번호 */}
            <CountryTelSection className='center'>
            <span className='title'>휴대전화</span>
            <select name="selectTel" className='field inputBox' >
                <option>대한민국 +82</option>
                <option>일본 +81</option>
                <option>중국 +86</option>
              </select>
            </CountryTelSection>

            {/* 휴대폰 번호 */}
            <TelSection className='center'>
            <div className='telBox'>
                <input type='text' name='tel' className='field birthBox'
placeholder='전화번호 입력'/>
```

```
              </div>
              <div className='buttonBox'>
                <button>인증번호 받기</button>
              </div>
            </TelSection>

            {/* 인증번호 */}
            <CertificationSection className='center'>
              <div className='pw_form'>
                <label htmlFor='certification'></label>
                <input type="text" name="certification" className="field inputBox"
placeholder='인증번호 입력하세요'/>
              </div>
            </CertificationSection>
            <hr/>

            <JoinBox>
              <button>가입하기</button>
            </JoinBox>
          </MainContainer>
        </form>

        </>
  )
}

export default React.memo(Main);
```

Footer.jsx

```
import React from 'react'
import styled from 'styled-components';

const FooterSection = styled.div`
margin-top: 30px;
font-size: 12px;
font-weight: 100;
text-align: center;
color: gray;
word-spacing: 1px;

.underLine{
    display: inline;
    justify-content: space-evenly;
    span{
    border-right: 0.5px solid #d5d5d5;
    padding: 0 10px;
        &:last-child{
            border-right: none;
        }
        &:hover{
```

```
            text-decoration: underline;
            color: #03c75a;
            cursor: pointer;
        }
    }
}

.copyRight{
    margin-top: 10px;
    padding-bottom: 20px;
    font-weight: 400;
    &:hover{
            text-decoration: underline;
            color: #03c75a;
            cursor: pointer;
        }
}
`;
const Footer = () => {
  return (
    <FooterSection>
        <div className="underLine">
            <span>이용약관</span>
            <span><b>개인정보처리방침</b></span>
            <span>책임의 한계와 법적고지</span>
            <span>회원정보 고객센터</span>
        </div>
        <div className="copyRight">
            <span>&copy; NAVER Corp.</span>
        </div>
    </FooterSection>

  )
}

export default React.memo(Footer);
```

문제 및 소감

정규 표현식을 조금 더 업그레이드를 시켜보고 싶어서
찾아 보는데 아직 그 방법을 온전히 제가 구현하기에는
어려움이 있어 시간 내에 구현을 하기에는 무리가 있어
하다가 완성을 하는 못했지만, 할 수 있는 방법들을
찾아보면서 혼자라도 해보겠다는 생각이 들었습니다.

코딩을 하면서 에러는 많이 마주 하다보니 문제를
해결하는데 시간을 너무 많이 소요를 한 점이 아쉬웠습니다.
에러노트를 조금 더 열심히 적어보고 혼자 해결하는 능력을
길러야 한다는 생각을 더 격하게 했습니다.

이번 코딩을 하면서 많은 자료들을 보고 영상도 찾아보고
수업 자료들도 보다 보니 그때는 완전히 이해가 되지 않았던
부분들을 다시 보고 공부를 할 수 있어서 좋았습니다.