인터넷과 웹 기초 텀 프로젝트 최종 보고서

201624589 조채우

최소 개발 기준

OPEN API 2개 이상 활용

1. 부산광역시 장애인 전동보장구 급속충전기 설치 현황:
   https://www.data.go.kr/data/15034013/openapi.do
2. 부산광역시 부산맛집 정보 서비스:
   https://www.data.go.kr/data/15063472/openapi.do
3. 부산광역시 부산명소 정보 서비스:
   https://www.data.go.kr/data/15063481/openapi.do
4. 네이버 클라우드 플랫폼 Maps – Web Dynamic Map:
   https://www.ncloud.com/product/applicationService/maps

웹 페이지에 표시할 정보를 1, 2, 3번 OPEN API를 이용해 가져왔고, 그 데이터를 이용해 위치정보를 얻어 Naver 지도에 표시했다.

서로 다른 HTML element 15개

```
<body>
    <div id="list">
        <div id="place_nav">
            <button class="category_button" onclick="changeMenu(0)">맛집</button>
            <button class="category_button" onclick="changeMenu(1)">관광명소</butto
n>
            <button class="category_button" onclick="setChargerMarker()">장애인<br>
편의시설</button>
            <form name="search" method="dialog">
                <input type="text" name="keyword">
                <button class="submit_button" onclick="searchListItem()">
                    <img src="./round_search_black.png">
                </button>
            </form>
        </div>
        <div id="place"  class="long_content">
            <ul class="food_list"></ul>
            <ul class="food_list"></ul>
```

```
            <ul class="food_list"></ul>
            <ul class="food_list"></ul>

            <ul class="place_list"></ul>
            <ul class="place_list"></ul>
            <ul class="place_list"></ul>
            <ul class="place_list"></ul>
        </div>
    </div>
    <div id="map">
    </div>
    <button id="extend_button"></button>
    <div id="describe">
        <div class="content" id="food_content">
            <h2></h2>
            <button class="lang_button" onclick="changeLanguage()">
                <img src="./korea.png" alt="flag">
                <p>Ko</p>
            </button>
            <img id="photo" alt="photo">
            <div id="available" class="impact"></div>
            <div id="menu" class="impact"></div>
            <div id="detail"></div>
            <div id="address" class="impact"></div>
            <div id="call" class="impact"></div>
        </div>
        <div class="content" id="place_content">
            <h2></h2>
            <button class="lang_button" onclick="changeLanguage()">
                <img src="./korea.png" alt="flag">
                <p>Ko</p>
            </button>
            <img id="view" alt="view">
            <div id="welfare" class="impact"></div>
            <div id="p_detail" ></div>
            <div id="p_call" class="impact"></div>
            <div id="p_address" class="impact"></div>
        </div>
    </div>
</body>
```

Code 1. project.html, body 부분

 메뉴 선택을 위한 네비게이션 바 장소를 검색할 수 있는 input 장소들을 나열할 리스트	 지도 API를 표시할 공간	 리스트에서 장소를 선택시 장소의 자세한 내용을 볼 수 있는 공간

HTML 구성은 위와 같이 크게 3개로 나누어 구성했다.
명소와 맛집 두 장소를 설명하기 위해 하나의 를 사용하기엔 설명할 내용이 서로 맞지 않아 두개의 division을 구성하고 javascript를 이용해서 display 속성을 변경해가며 둘 중 하나만 보이도록 구성했다. 왼쪽의 list 또한 장소와 명소, 그리고 각각 3개의 국가 언어로 총 6개의 unordered list로 구성했다. navigation bar는 button 3개를 병렬로 배치해 구성했고, 검색을 위해 form과 input text를 사용했다. 이외에도 언어 변경 버튼에 해단 언어에 맞는 국가의 국기 이미지를 넣기 위해 <img>를 사용하는 등 15개 이상의 HTML element를 활용했다.

서로 다른 CSS selector로 15개

```
@font-face {
    font-family: "nanumsq";
    src: url("./NanumSquareR.ttf");
}

body{
    margin: 2px 0px;
    font-family: 'nanumsq', initial;
}

div {
    float: left;
    height: 100%;
    box-sizing: border-box;
}

button{
    border-width: 0px;
    float: left;
}

#map {
    width: 50%;
    border-width: 2px;
    border-color: black;
}
#list{
```

```
        width: 25%;
    }
    #describe{
        width: 25%;
    }
    #place {
        overflow-y: scroll;
        overflow-x: hidden;
        width: 100%;
        height: 85%;
    }

    #place > ul {
        display: none;
        box-sizing: border-box;
        margin: 0px;
        padding: 0px;
        list-style: none;
    }
    #place > ul button {
        text-align: left;
        font-size: 15px;
        padding:0px 5px;
        margin: 10px 0xp;
        width: 100%;
        height: 90px;
        background-color: whitesmoke;
    }
    #place > ul button:hover{
        background-color: lightgray;
    }

    #place > ul  button > img {
        float: left;
        height: 90%;
        width: 30%;
    }
    .button_title_div{
        padding: 5px;
        height: 40%;
        width: 70%;
        font-size: 15px;
    }
```

```
    .button_addr_div {
        padding: 5px;
        height: 40%;
        width: 70%;
        font-size: 12px;
    }

    #place_nav {
        width: 100%;
        height: 15%;
    }
    .category_button {
        width: 33.3%;
        height: 50px;
        margin: 0%;
        text-align: center;
        background-color: black;
        color: white;
    }
    .category_button:hover {
        background-color: slateblue;
    }
    #place_nav input{
        float: left;
        box-sizing: border-box;
        border-radius: 3px;
        margin: 5px;
        height: 35%;
        width: 85%;
    }
    .submit_button {
        float: left;
        box-sizing: border-box;
        border-radius: 5px;
        border-width: 1px;
        border-color: silver;
        padding: 0%;
        margin: 5px;
        height: 35%;
        width: 9.5%;
    }
    .submit_button img {
        box-sizing: border-box;
```

```
        width: 100%;
        height: 100%;
    }

    #describe > div{
        display: none;
        overflow: auto;
        width: 100%;
        height: 100%;
        padding: 0px 3px;
        background-color: whitesmoke;
    }

    #describe h2 {
        padding-left: 5%;
        height: 32px;
    }

    .lang_button{
        position: fixed;
        bottom: 0%;
        right: 25%;
        box-sizing: border-box;
        border-radius: 4px;
        border: solid;
        border-color: rgba(192, 192, 192, 0.7);
        border-width: 2px;
        height: 30px;
        width: 100px;
    }

    .lang_button > img {
        height: 23px;
        width: 32px;
        float: left;
        border: solid;
        border-radius: 2px;
        border-width: 1px;
        border-color: rgb(239, 239, 239);
    }

    .lang_button > p{
        margin: 0px;
```

```
        padding: 5px;
    }

    #extend_button{
        position:absolute;
        height: 6%;
        top: 47%;
        right: 25%;
        border-radius: 3px;
        border-width: 1px;
        border-color: gray;
    }

    #extend_button:hover{
        background-color: lightgrey;
    }

    .content > div{
        padding-left: 8px;
        height: auto;
        width: 100%;
        margin: 3% 0%;
    }

    .impact::first-line {
        font-family: Impact;
    }

    #detail{
        height: 16% !important;
        font-size: 12px;
    }

    #place_content{
        overflow: auto;
    }

    #p_detail {
        font-size: 12px;
    }
    #welfare{
        font-size: 15px;
    }
```

```css
#photo, #view{
    display: none;
    padding: 2% 5%;
    width: 90%;
    height: 30%;
}
.long_content::-webkit-scrollbar {
    width: 7px;
}

.long_content::-webkit-scrollbar-thumb {
    background-color: lightgrey;
    border-radius: 15px;
}

.long_content::-webkit-scrollbar-track {
    background-color: white;
}
```

Code 2. project.css 코드

약 30개 가량의 CSS Selector를 사용했다.

Figure 1. 스크롤 스타일 변경

추가적으로 개발에 사용해본 subclass로 -webkit-scrollbar가 있다. 해당 subclass는 크롬에서 적용이 가능하다고 해서 직접 적용해 보았다. 위의 그림과 같이 스크롤 바의 두께, 배경색, 스크롤의 모양 또한 변경이 가능했다.

아직 CSS의 우선순위가 익숙하지 않아서 속성 중에 !important 또한 검색을 통해 알게 되었다.

강의에서 배운 것 중 사용한 것으로는 class와 id, '>', ::first-line, :hover 등이 있다.

JavaScript를 통해 3가지 이상의 동적 기능

JavaScript를 통해 구현한 기능은 총 4가지이다.

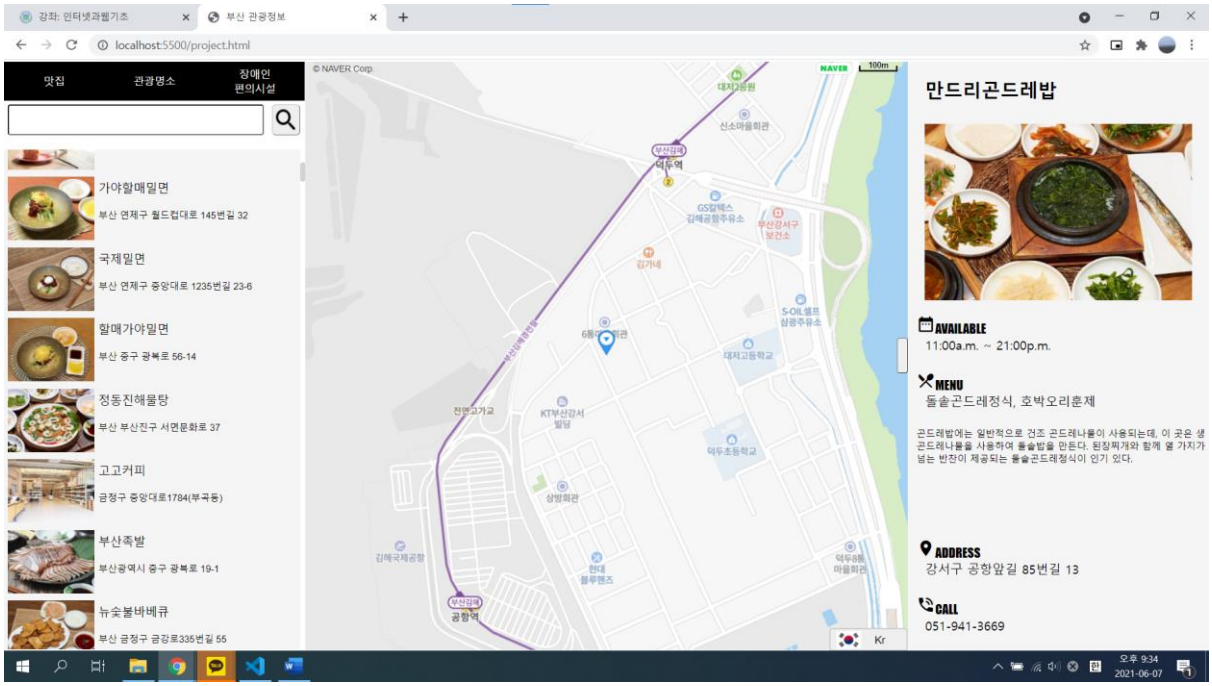

Figure 2.웹 최초 화면

1. 왼쪽의 맛집, 명소 리스트 중 하나를 누르면 오른쪽의 공간에 선택한 장소에 대한 세부 정보로 변경됨

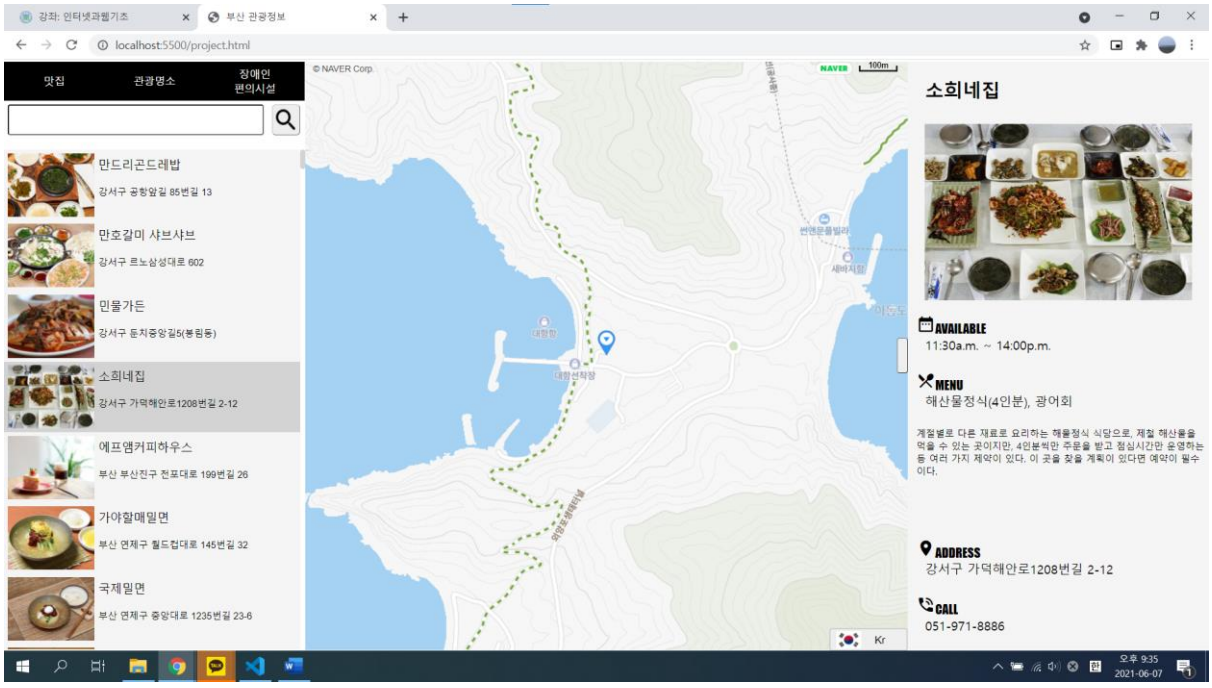

Figure 3.리스트 선택 후 변경된 화면

2. Navigation bar의 버튼으로 맛집의 리스트와 관광명소의 리스트를 교체할 수 있음

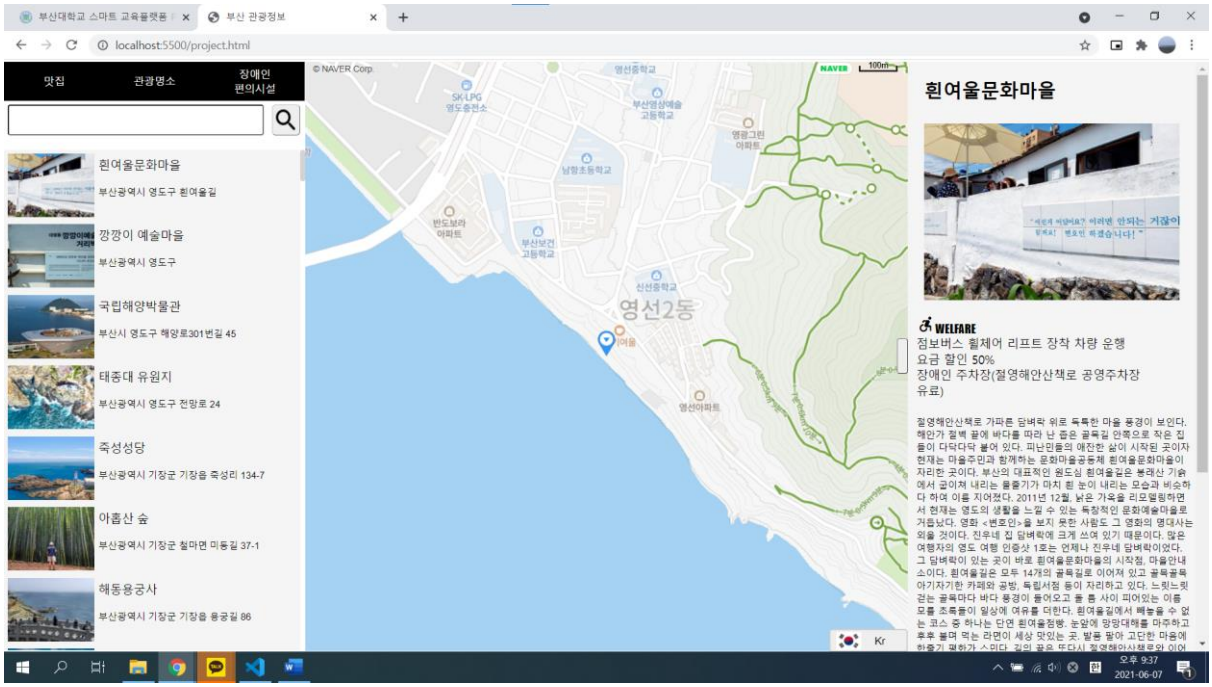

Figure 4. 관광명소 버튼 클릭 후 화면

3. 장애인 편의시설 버튼을 누르면 보조장구의 급속 충전기 위치가 지도에 표시됨

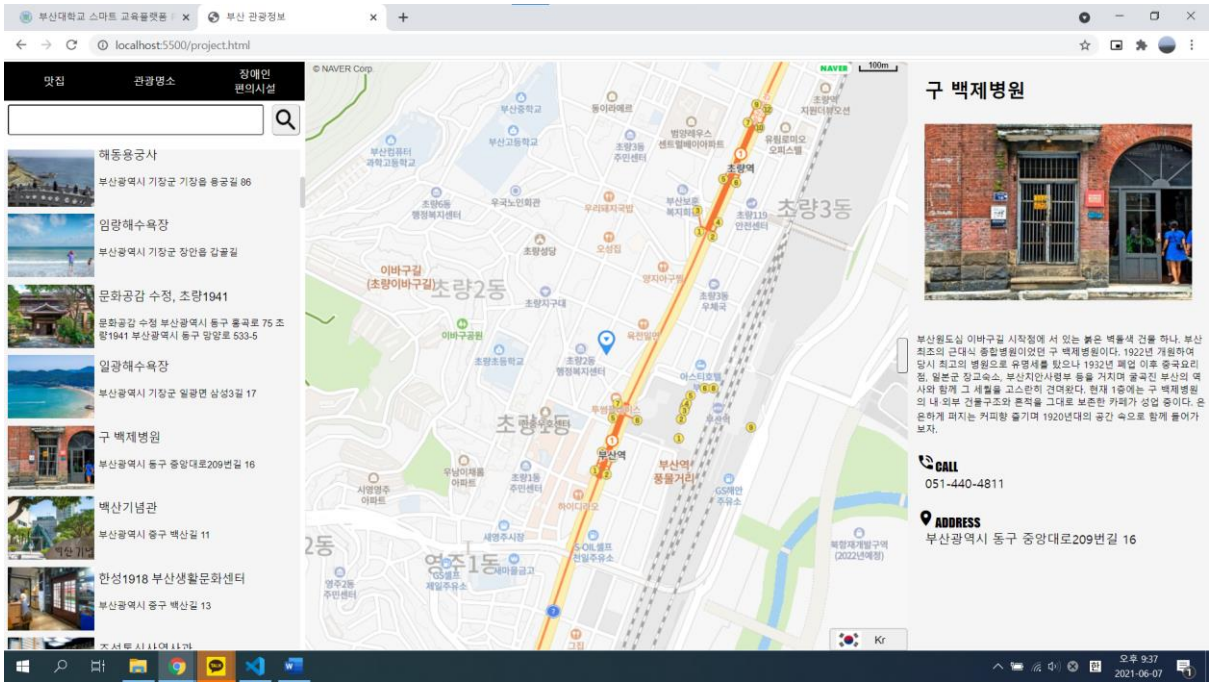

Figure 5. 위치 표시를 껐을 때

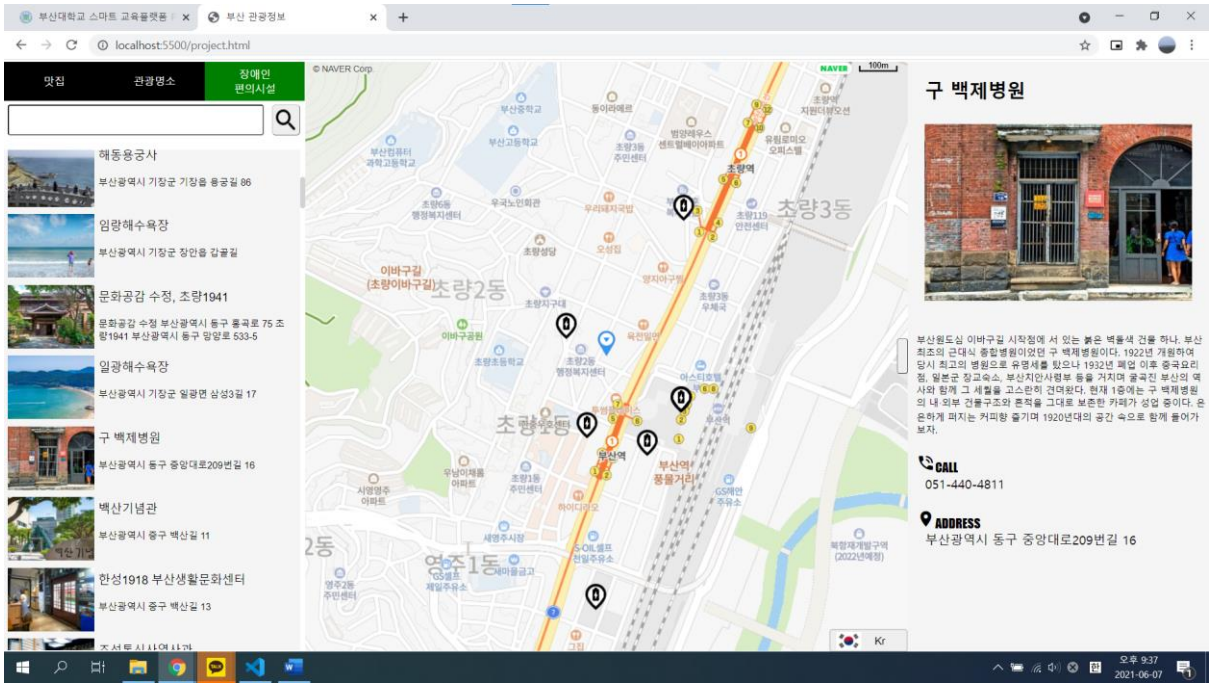

Figure 6. 위치 표시를 켰을 때

4. (추가구현)우측 하단의 국기 버튼을 누르면 순서대로 한국어-영어-일본어 순서로 언어가 바뀜

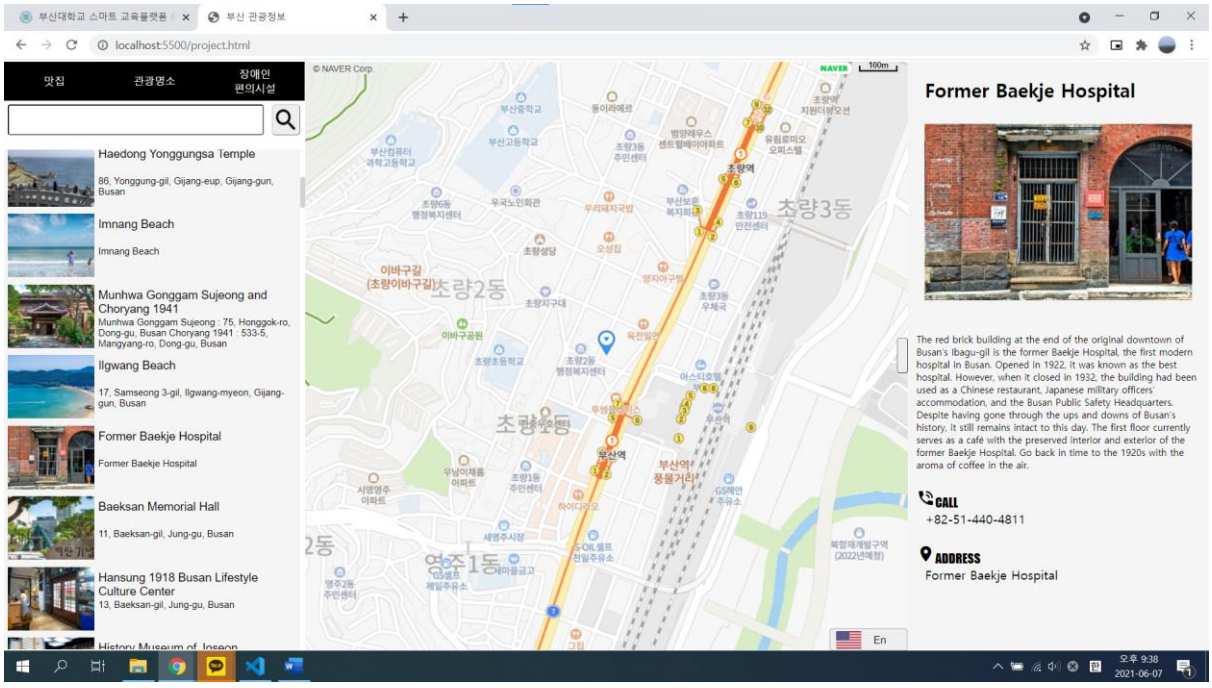

Figure 7.언어 변경버튼 클릭 후 (영어)

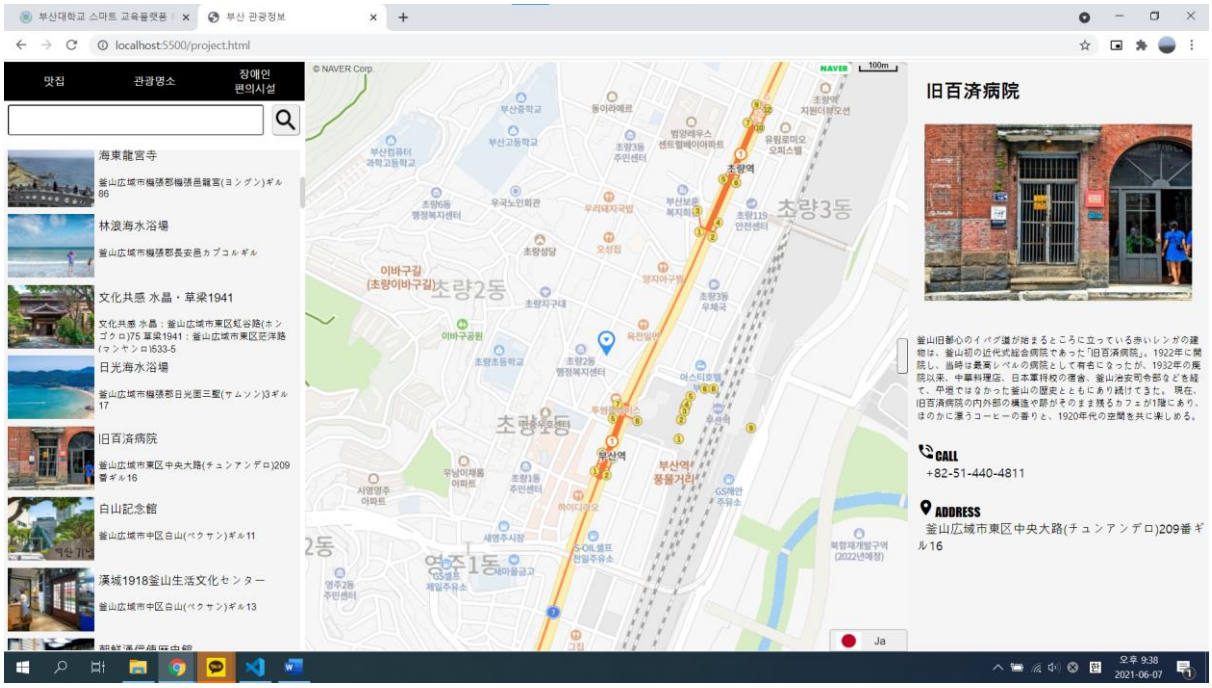

Figure 8. 언어 변경버튼 클릭 후 (일어)

5. (추가구현) 리스트 위의 검색창을 이용해 키워드가 포함된 장소만 리스트에 표시함

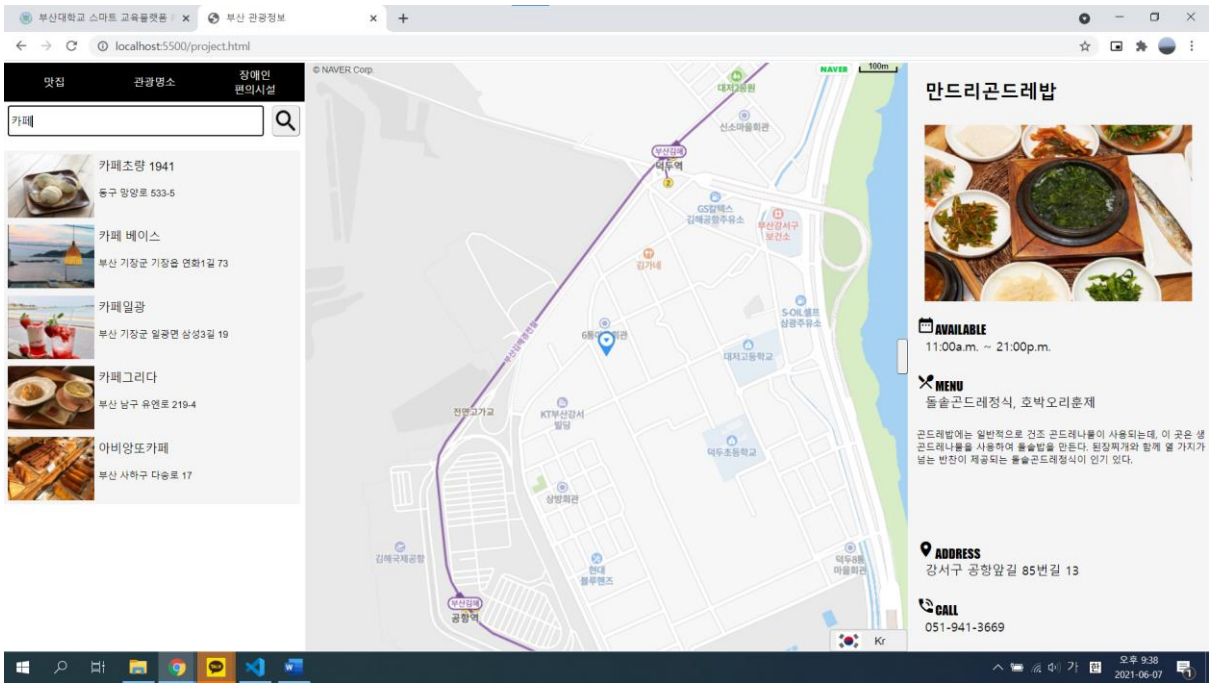

Figure 9. "카페" 검색 후 결과화면

JavaScript를 활용 부적절한 입력 및 OPEN API의 response중 에러 처리

```
function parseJSON(json_var, menu, lang) {
    try {
        var list = JSON.parse(json_var);
    } catch (e){
        xmlParser = new DOMParser();
        const xmlObject = xmlParser.parseFromString(json_var, "text/xml");
        var msg = '';
        for( i in xmlObject.firstChild.childNodes){
            if(xmlObject.firstChild.childNodes[i].textContent == undefined) { continue;}
            msg += xmlObject.firstChild.childNodes[i].textContent;
        }
        msg = msg.replaceAll('\t', '');
        alert("데이터를 읽어오는 중 문제가 생겼습니다.\n" + msg);
        return;
    }
```

Figure 10.project.js 내부의 에러처리 코드 부분

Node js를 이용해서 'type=json'으로 받아온 정보를 JSON object로 파싱하는 함수에서 try-catch문을 이용해 에러를 처리했다. 테스트 결과 Error가 발생한 경우 JSON object에 에러코드가 전송되는 것이 아닌 xml 형식의 데이터가 전송되는 것을 확인했고, 따라서 response중 에러가 발생한 경우 parseJSON이 정상적으로 작동하지 않아 catch문에 의해서 웹에서 alert 알림을 전송하도록 했다. 전송된 error에 관련한 xml 데이터는 DOMparser의 객체에 의해서 parse되어서 alert 알림에 추가된다. 오류관련 알림 데이터에 정해진 형식이 없어 제대로 동작하지 않을 수 있다.

```
<OpenAPI_ServiceResponse>
        <cmmMsgHeader>
                <errMsg>SERVICE ERROR</errMsg>
                <returnAuthMsg>SERVICE_KEY_IS_NOT_REGISTERED_ERROR</returnAuthMsg>
                <returnReasonCode>30</returnReasonCode>
        </cmmMsgHeader>
</OpenAPI_ServiceResponse>
```

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<result>
    <code>ERROR-601</code>
</result>
```

Figure 11. 서로 다른 형식의 오류 메시지들

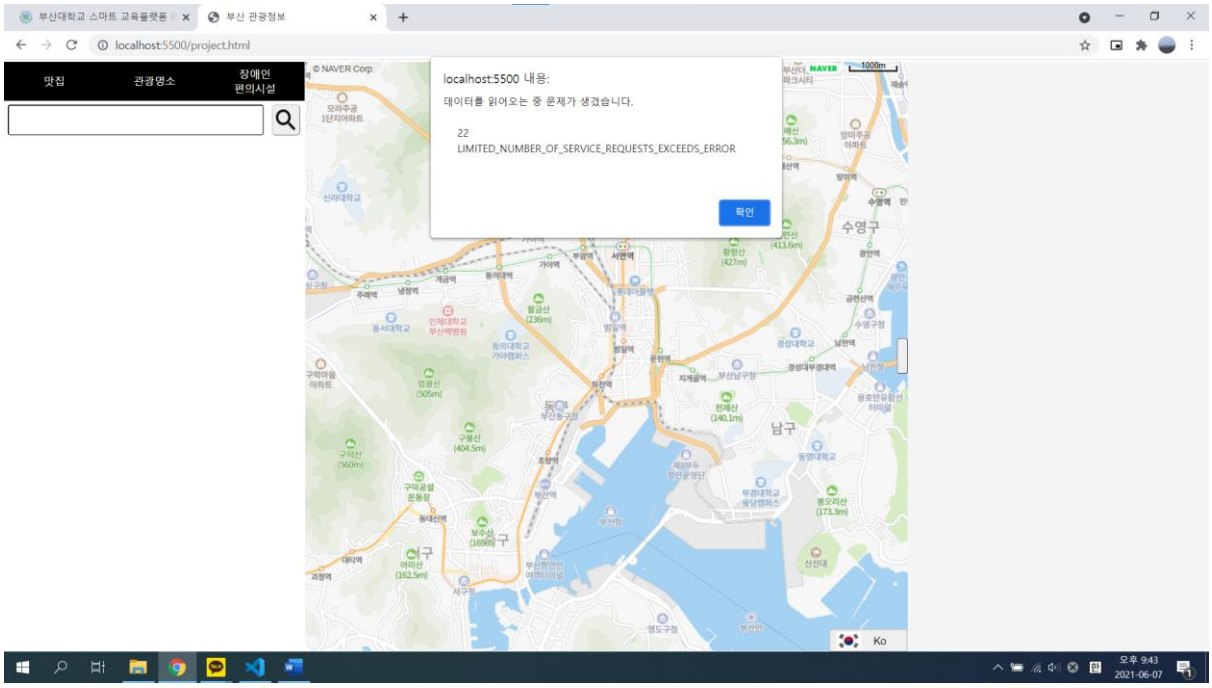

Figure 12.alert 메시지

jQuery/AJAX/JSON 기능 활용

가장 먼저 Open API의 data는 모두 JSON item 전달 받아서 웹 페이지의 구성에 사용했다.

그리고 jQuery의 animate를 이용해서 오른쪽의 메뉴를 축소시켜 지도를 더 크게 볼 수 있는 기능을 구현했다.

```
$(document).ready(function(){
    var isOdd = 0;
    $("#extend_button").click(function(){
        if(isOdd % 2 == 0){
            $('#describe > div').css('padding','0px'),
            $('#extend_button').animate({
                right: '0%'
            }),$('.lang_button').animate({
                right: '0%'
            }),$('#describe').animate({
                width: '0%'
            }),$('#map').animate({
                width: '75%'
            }, function(){
                mapObject.setSize({width: $('#map').css('width'), height: $('#map').css('heig
ht')});
            });
        }
        else{
```

```
            $('#describe > div').css('padding','0px 3px'),
            $('#extend_button').animate({
                right: '25%'
            }),$('.lang_button').animate({
                right: '25%'
            }),$('#describe').animate({
                width: '25%'
            }),$('#map').animate({
                width: '50%'
            }, function(){
                mapObject.setSize({width: $('#map').css('width'), height: $('#map').css('heig
ht')});
            });
        }
        isOdd = (isOdd + 1) % 2;
    });

});
```

Code 3. jQuery 코드 전문

click()을 사용해서 버튼을 누를 때 animate()가 동작한다.

animate는 division의 너비와 버튼의 위치를 조정하고 그 과정을 천천히 움직이는 애니메이션으로 표시한다. 이후 지도 API를 표시하는 map의 너비가 바뀐 뒤 콜백으로 지도의 크기를 지정하는 메소드를 사용하게 했다.

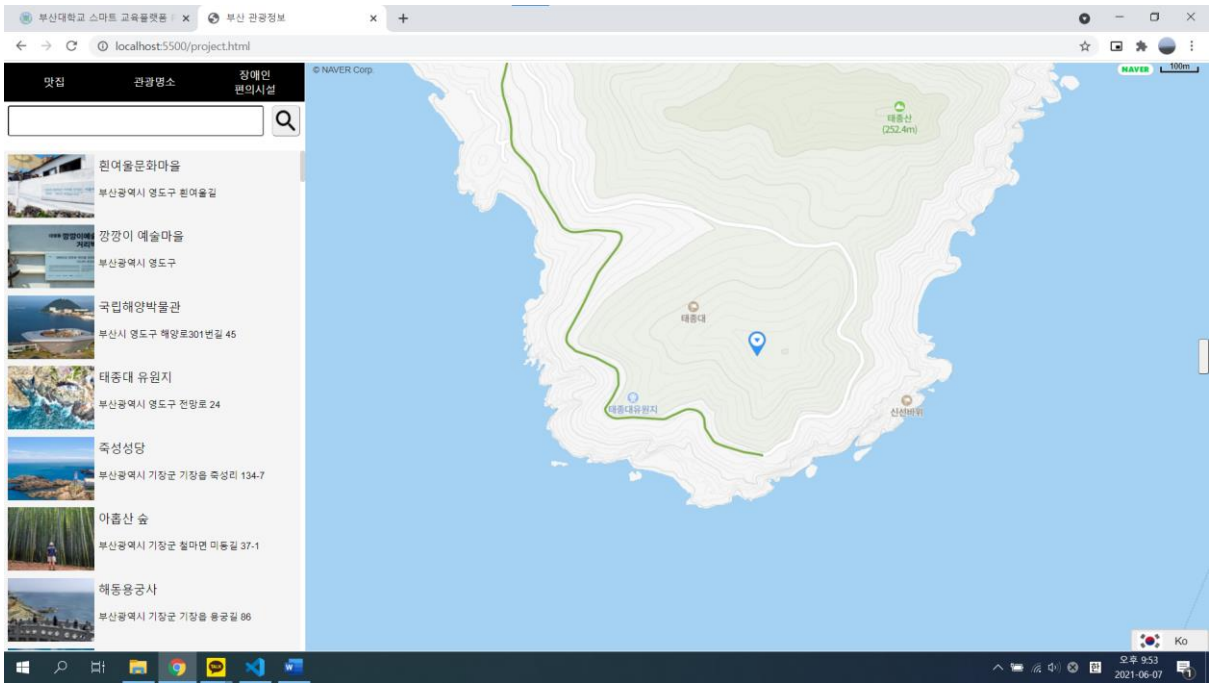

Figure 13.버튼을 눌러 왼쪽 창을 최소화한 모습