

INFO

URL : https://boramiiii.github.io/boram_portfolio_first/

SPEC :   

RESPONSIVE : PC, Tablet, Mobile



Exhibition Of The Month



<

히토 슈라이얼 - 데이터의 바다

Lorem ipsum dolor sit amet consectetur, adipisicing elit. Laborum cupiditate, sit alias assumenda consequatur, natus modi similique eligendi repudiandae itaque et ea, soluta eius? Quis praesentium earum labore apriam qui temporibus error dicta officiis ad. Dignissimos molestiae atque nulla quidem?

2022.04.29 - 2022.09.19

[전시 관람 예약하기](#)

>

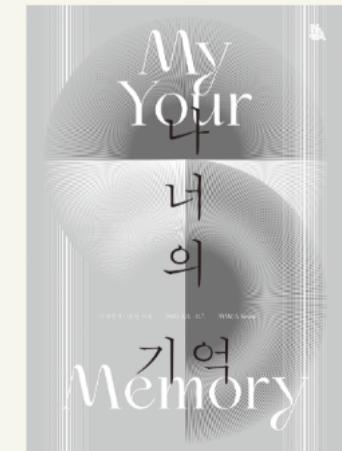
Current Exhibition

현대미술관에서 현재 진행중인 전시를 알려드립니다.

[More Exhibition](#)



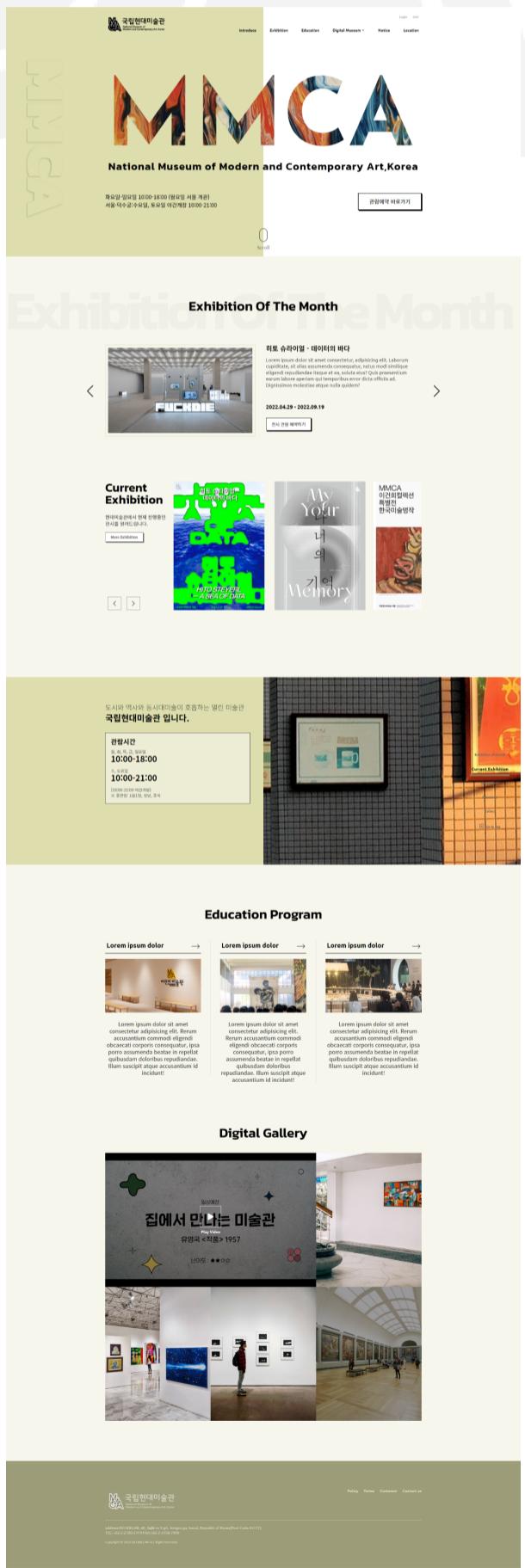
< >



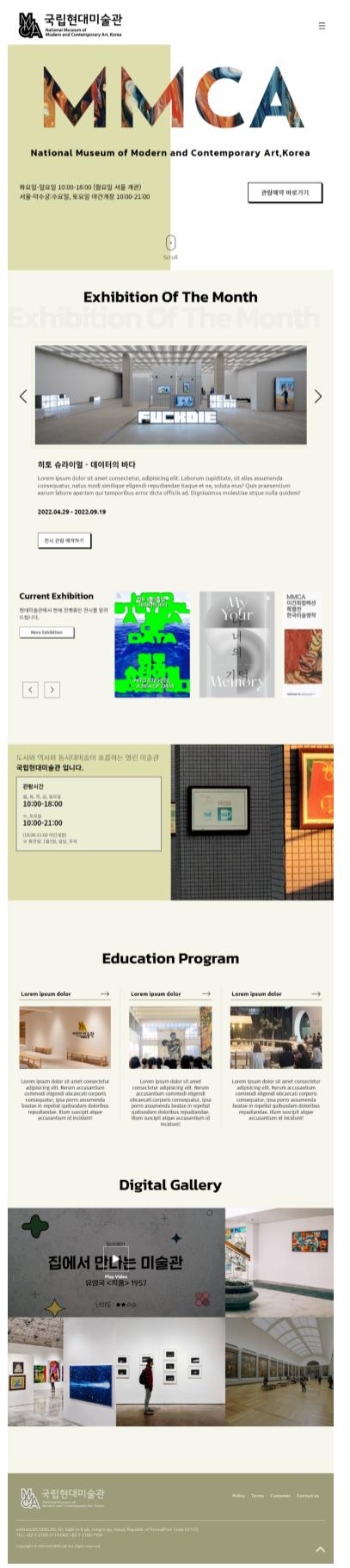
1. Responsive Layout

mediaQuery를 활용해 브라우저 폭에 따라 다양한 기기 환경에서도 동일하게 웹사이트를 이용할 수 있도록 처리

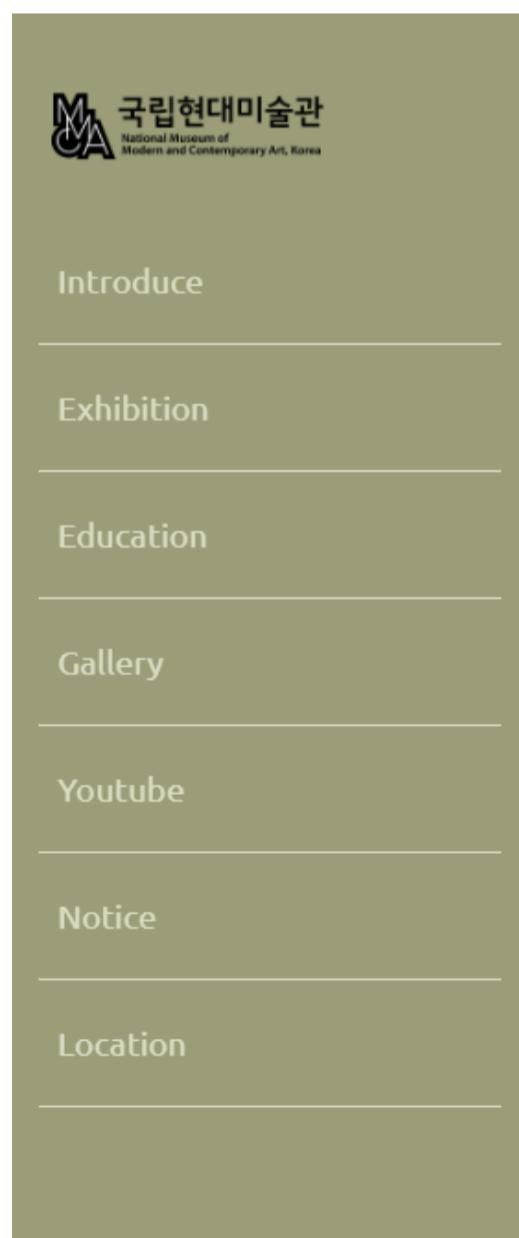
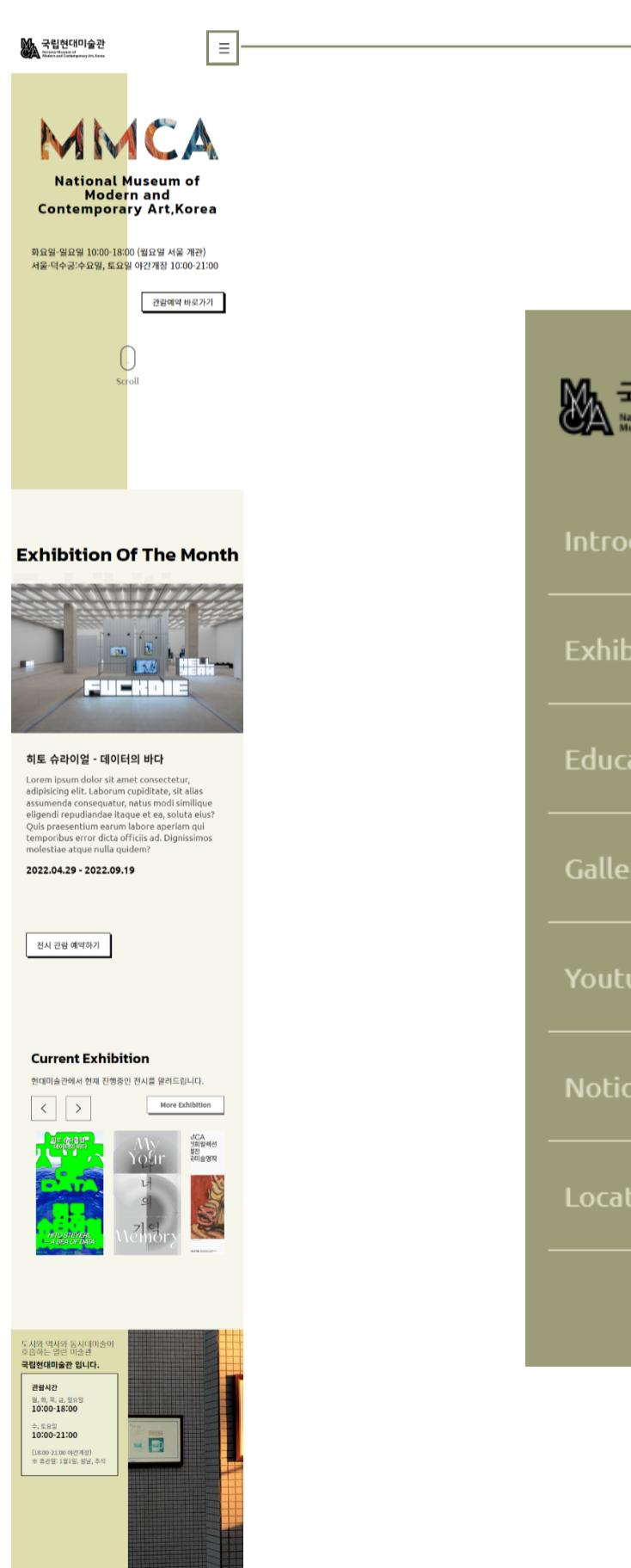
Desktop



Tablet



Mobile



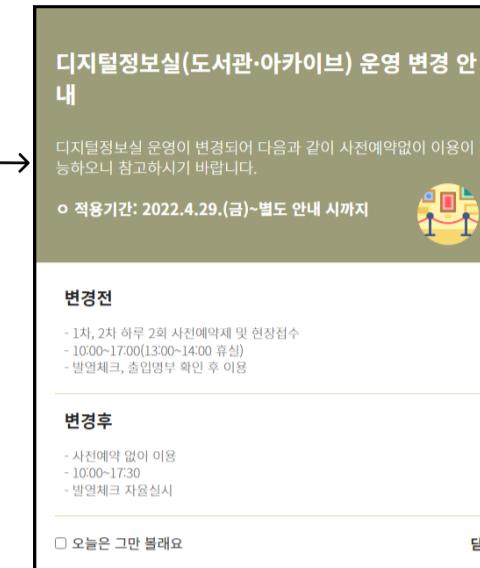
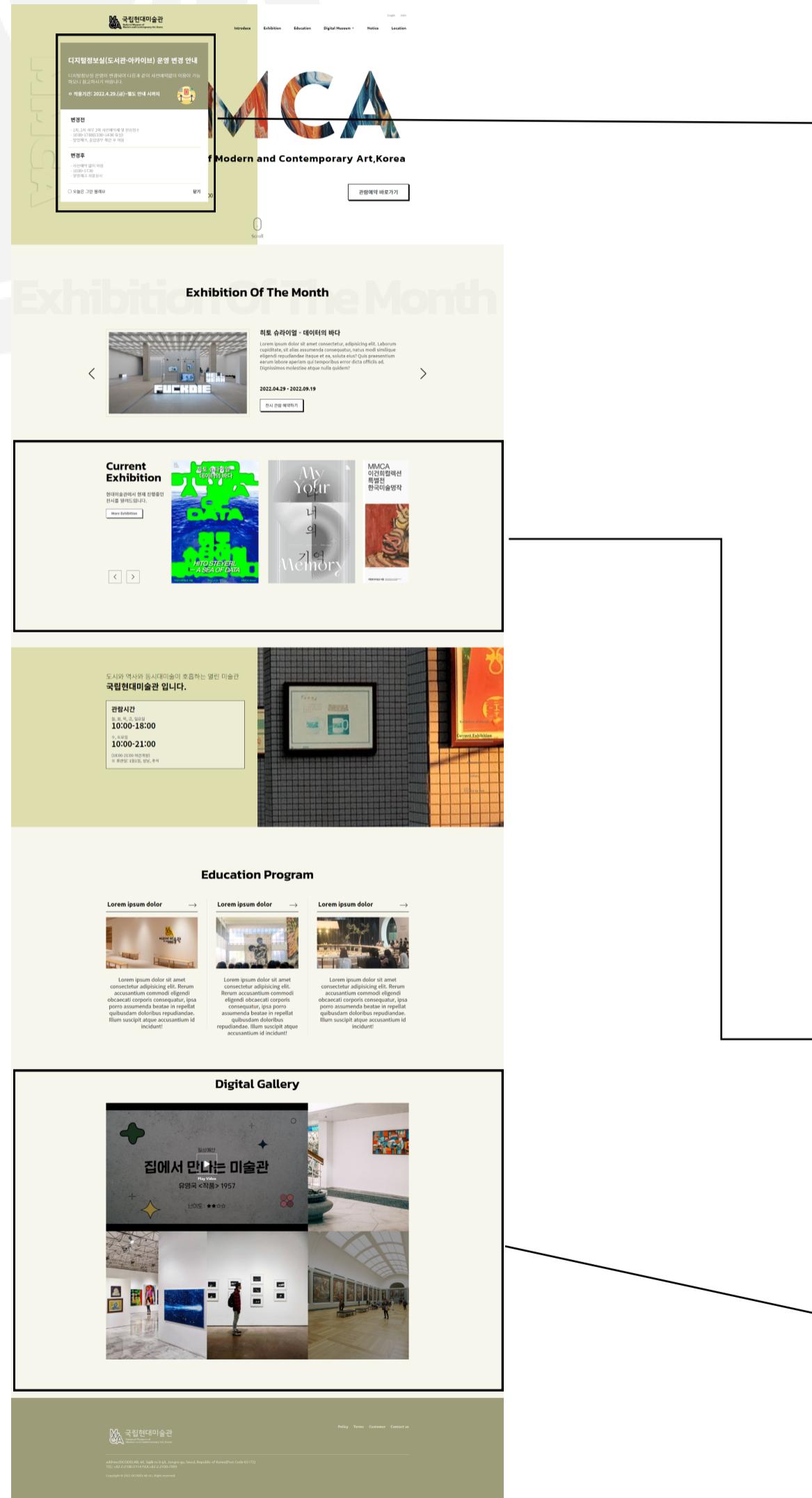
tablet, mobile 사이즈 일때
bar 아이콘 활성화.
아이콘 클릭 시 메뉴 활성화
되어 서브페이지로 이동가능

반응형 미디어 쿼리 사이즈 정의

tablet : 1179px

mobile : 539px

2. Main Page Structure



page 로드시 popup display:block;
쿠키값 제어로 오늘은 그만블래요 체크 후 닫기버튼 클릭 시
생성된 쿠키의 time값을 0으로 설정해서 현재시간으로 만료시간 덮어쓰기

```
popClose.addEventListener("click", e=>{
    e.preventDefault();
```

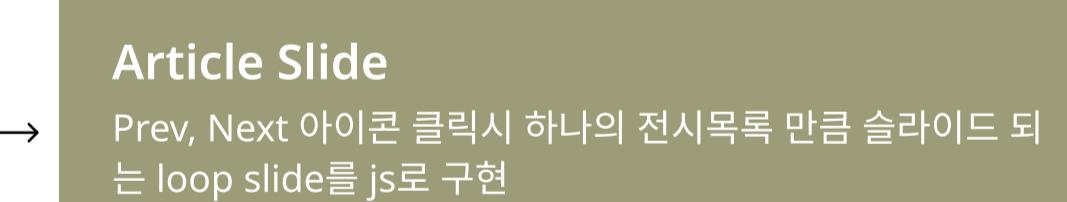
```
let isChecked = popup.querySelector("input[type=checkbox]").checked;
if(isChecked) setCookie("today", "done", 1);
popup.style.display = "none";
```

```
});
```

```
function setCookie(cookieName, cookieValue, time){
    const today = new Date();
    const date = today.getDate();
    today.setDate(date+ time);

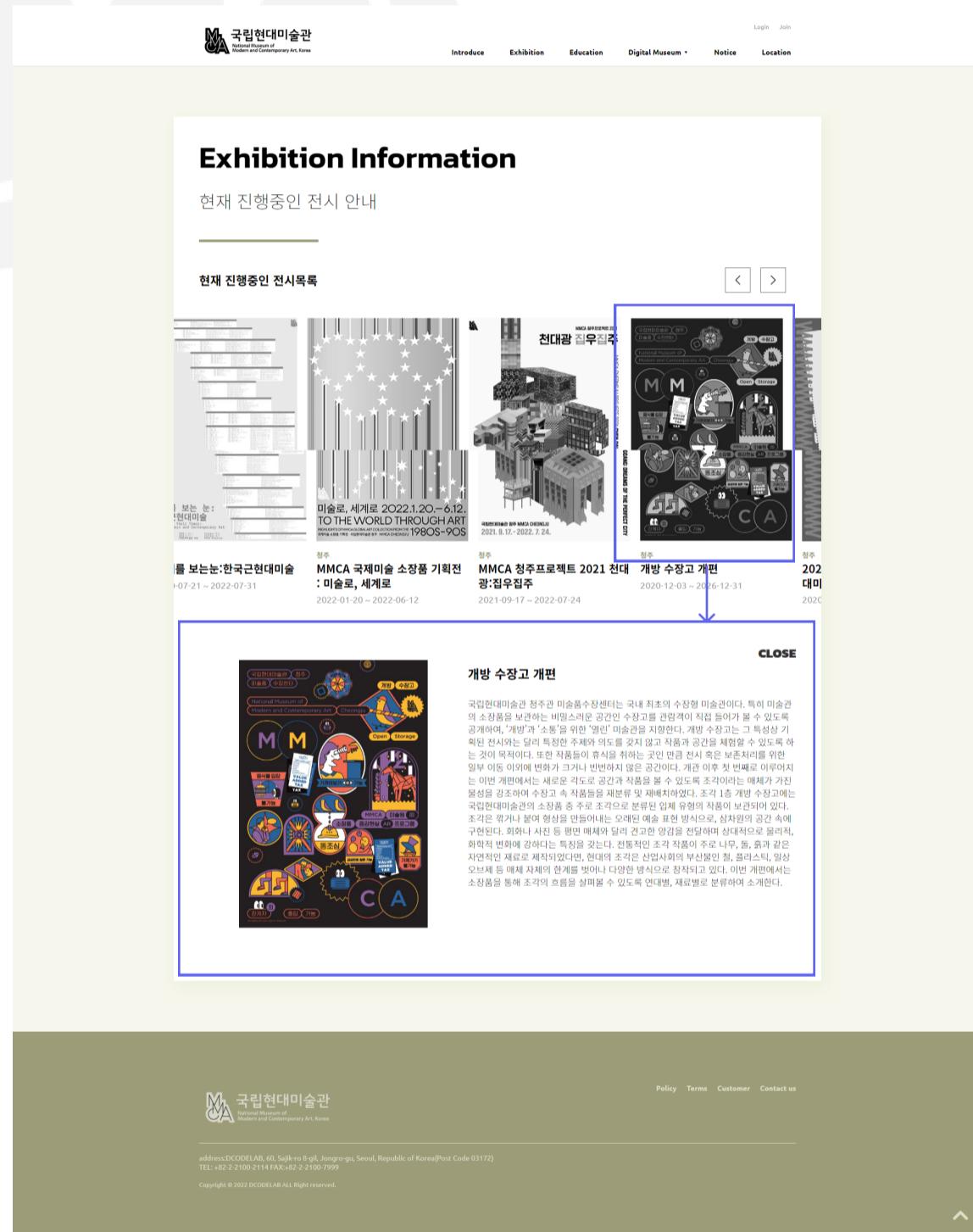
    const duedate = today.toGMTString();

    document.cookie=`${cookieName}=${cookieValue}; path="/"; expires=${duedate}`;
}
```



3. Subpage Structor

1. Exhibition



페이지 진입 시 자동 롤링 슬라이드 진행
마우스 호버 시 슬라이드 멈춤과 동시에 이미지 색상 on
호버된 이미지 클릭 시 하단에 전시 정보 open 구현

```
function createList(url){  
  fetch(url)  
  .then(data=>{  
    return data.json();  
  })  
  .then(json=>{  
  
    let items = json.imgSrc;  
    let tags =";  
  
    items.forEach(item=>{  
      tags+=`  
      ...  
    });  
  })  
  
  createList("data.json");  
}
```

json 으로 저장해놓은 전시 DB 를 불러와 tag를 만들어 리스트에 넣어줌

```
rolling.addEventListener("mouseenter", ()=>{  
  clearInterval(timer);  
});  
  
rolling.addEventListener("mouseleave", ()=>{  
  timer = setInterval(move, 50);  
});  
  
rolling.addEventListener("click", (e)=>{  
  e.preventDefault();  
  
  if(!e.target.closest(".pic img"))  
    return;  
})
```

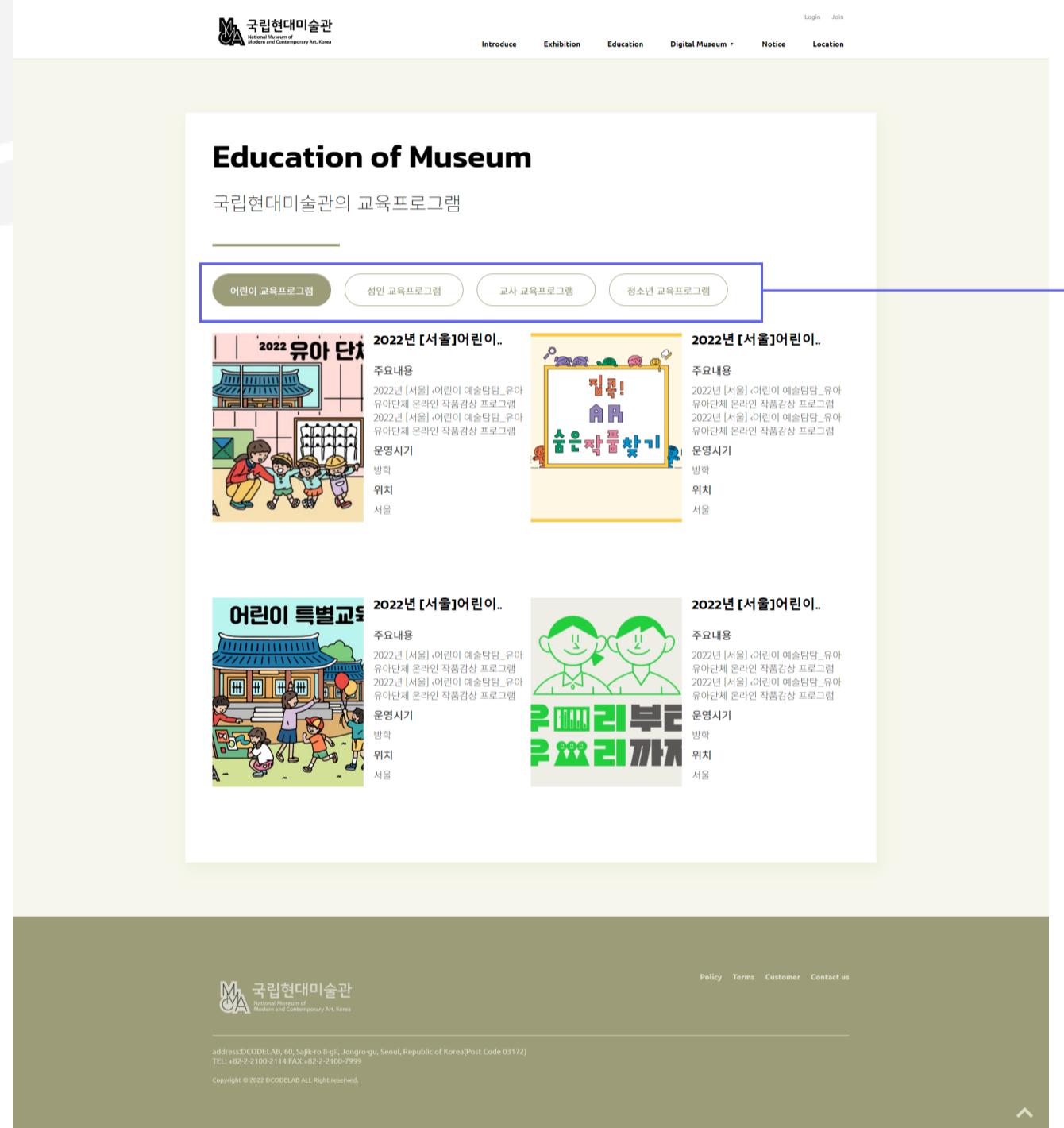
슬라이드 rolling 설정

```
const imgsrc = e.target.getAttribute("src");  
createContent("data.json", imgsrc);  
  
new Anime(btnclose, {  
  prop:"opacity",  
  value:1,  
  duration:500,  
});
```

전시 이미지 클릭 시 하단에 전시 상세 내용 보이기

3. Subpage Structor

2. Education



```
btns.forEach((el,index)=>{
  el.addEventListener("click", (e)=>{
    e.preventDefault();

    let isOn = e.currentTarget.classList.contains("on");
    if(isOn == true){
      return;
    }
    if(enableClick){
      enableClick = false;
      activation(btns, index);
      activation(contents, index);
    }

    new Anime(main, {
      prop:"height",
      value: matchHeight(index),
      duration:500
    })
  })
})
```

Tab 버튼 클릭 시 content 리스트에 on 클래스 추가.
클릭한 탭에 맞는 컨텐츠 opacity: 1;

```
function activation(arr, index){
  for(const item of arr){
    item.classList.remove("on");
  }
  arr[index].classList.add("on");

  setTimeout(()=>{
    enableClick = true;
  },delay)
}
```

on 클래스를 add, remove 동작 함수 정의

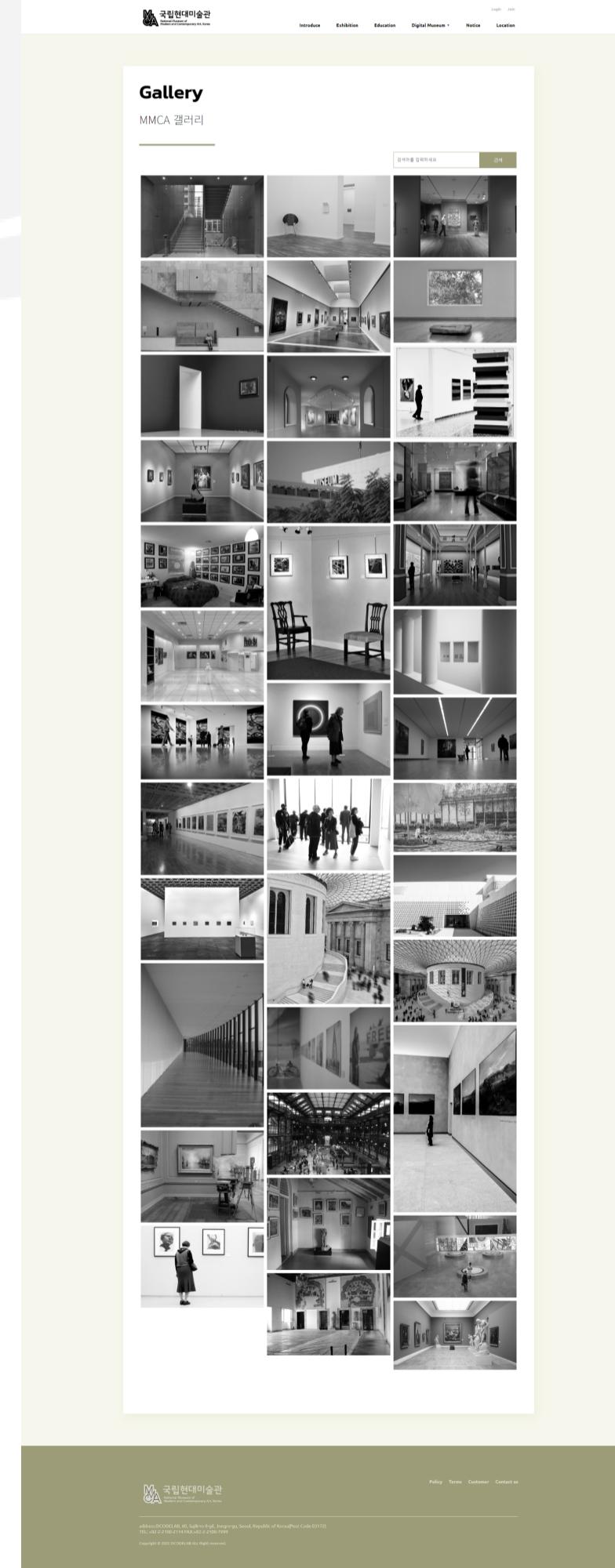
```
function matchHeight(index){
  let ht = "";
  ht = getComputedStyle(contents[index]).height;
  ht = parseInt(ht);

  return ht;
}
```

on 이 붙은 컨텐츠 높이값 구하는 동작 함수 정의

3. Subpage Structor

3. Gallery



- 이미지 공유 플러그인 FLICKR의 메서드를 활용해 이미지 관리 및 검색 기능 구현
- 효과적인 이미지 배치를 위한 그리드 레이아웃 플러그인 isotope을 사용

```
function callData(url) {
```

```
    loading.classList.remove("off");  
    frame.classList.remove("on");
```

```
    fetch(url)  
        .then(data => {  
            return data.json();  
        })  
        .then(json => {  
            let items = json.photos.photo;  
            console.log(items);
```

```
            if (items.length > 0) {  
                imgLoaded();  
                // document.body.classList.remove("msgWrap");  
            } else {
```

```
                let msgs = `  
                    <div class="errorMsg">  
                        ...  
                    </div>  
                `;
```

```
            };
```

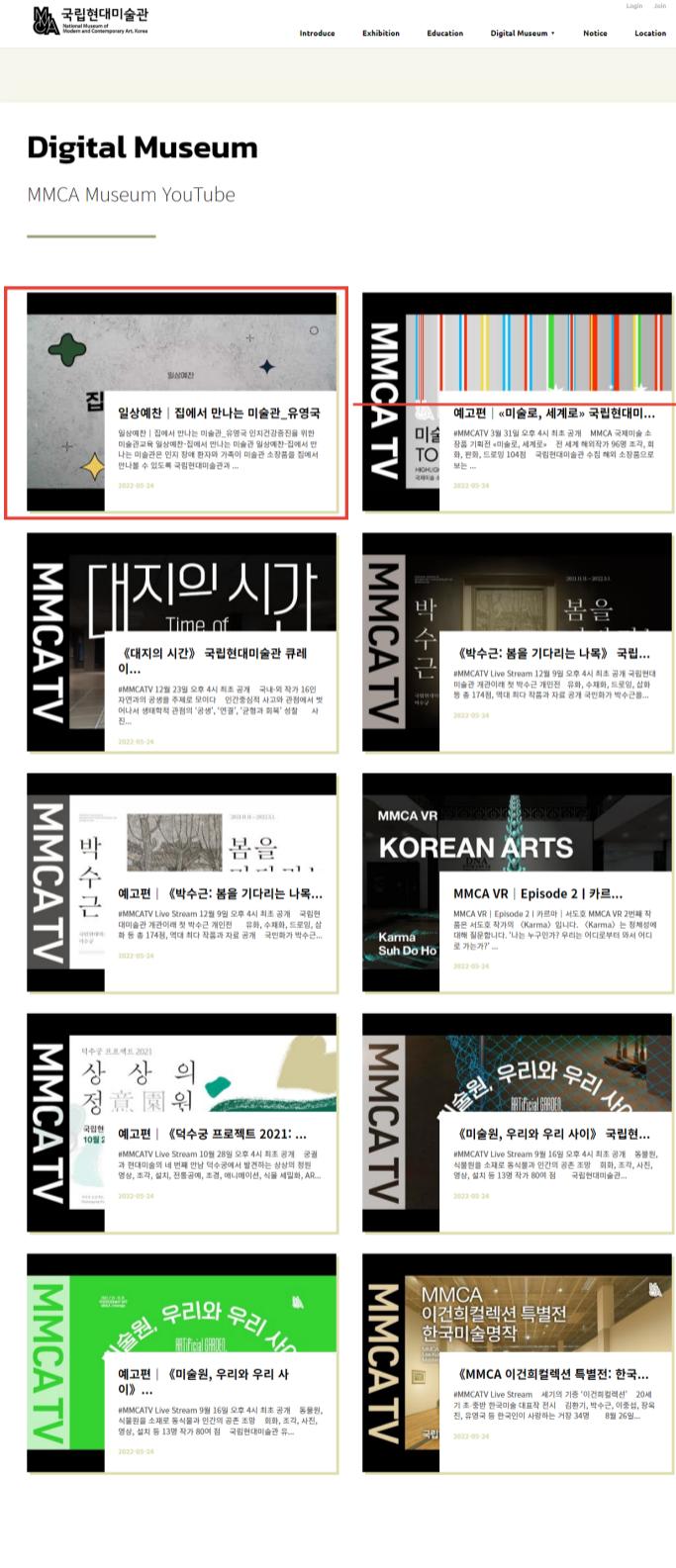
Flickr 리스트 call 함수 정의

검색하신 검색어의 이미지가 없습니다

보여줄 리스트가 없을 경우, 리스트가 없다는 이미지 보여주기

3. Subpage Structor

4. Youtube



```

fetch(url)
.then(data=>{
  return data.json();
})
.then(json=>{
  items.map(item=>{
    ...
    if(desc.length > 120){
      desc = desc.substr(0,120)+"...";
    }
  }
}

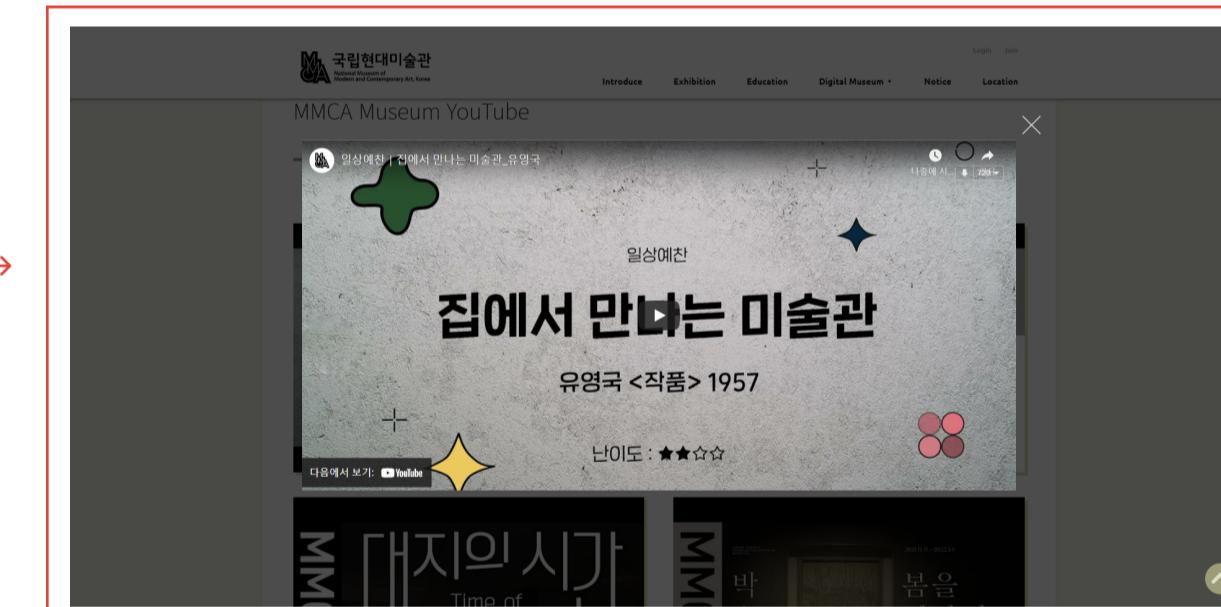
```

fetch로 불러온 youtube list 데이터로 레이아웃구현

```

result+=`<article>
<a href="#" data-
vid="${item.snippet.resourceId.videoId}" class="pic">
...
</article>`;
vid_list.innerHTML = result;
});
}

```



```
vid_list.addEventListener("click", (e)=>{
  e.preventDefault();
}
```

list 클릭 시 클릭한 동영상의 attribute를 받아와 클릭한 리스트의 youtube 동영상을 popup으로 띄워주는 기능 구현

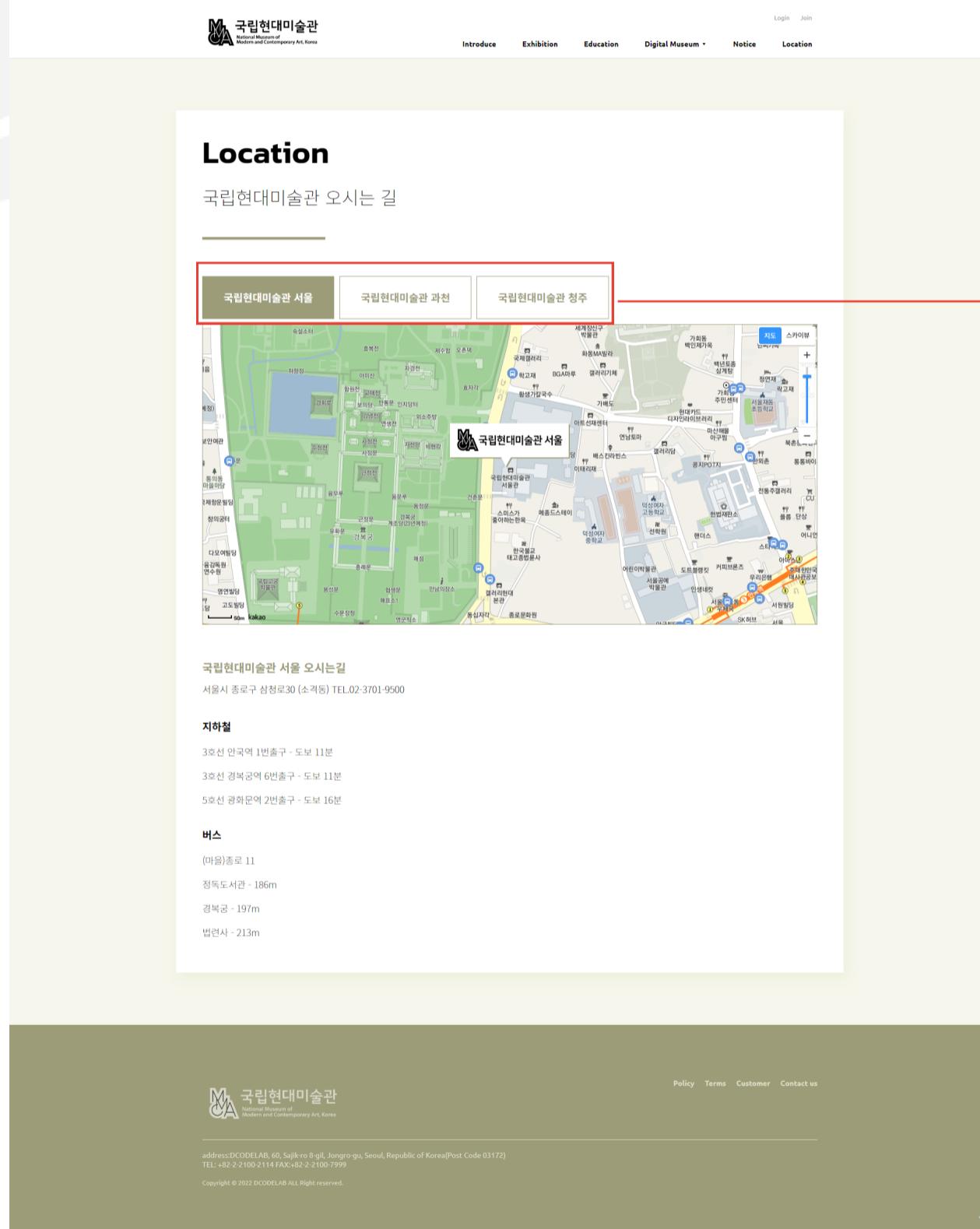
```

vidCon.innerHTML="";
e.display = 'block';
const vidId = e.target.parentElement.getAttribute("data-vid");
vidCon.innerHTML += `<iframe width="100%" height="100%" src="https://www.youtube.com/embed/${vidId}" frameborder="0" allowfullscreen></iframe>`;
}

```

3. Subpage Structor

5. Location



Contact 페이지 구현 기능

- Kakao map Api 호출
- 화면 리사이즈시 마커 중앙표시 유지
- 지도 확대 축소 기능
- 위치 마커 이미지 수정
- 버튼 클릭 시 해당 지점 위치 지도 호출

지도 호출

```
var mapOption = {  
    center: new kakao.maps.LatLng(37.57855540814129, 126.98008665570575),  
    level: 3  
};  
var map = new kakao.maps.Map(mapContainer, mapOption);
```

```
for(let i=0; i<markerOptions.length; i++){  
    var marker = new kakao.maps.Marker({  
        map: map,  
        position: markerOptions[i].latlng,  
        title : markerOptions[i].title,  
        image : new kakao.maps.MarkerImage(markerOptions[i].imgSrc, markerOptions[i].imgSize, markerOptions[i].imgPos)  
    });  
}
```

```
markerOptions[i].button.addEventListener("click", e=>{  
    e.preventDefault();  
    for(let branch of branch_btns){  
        branch.classList.remove("on");  
    }  
    for(let content of contents){  
        content.classList.remove("on");  
    }  
    markerOptions[i].button.classList.add("on");  
    moveTo(markerOptions[i].latlng)  
  
    contents[i].classList.add("on");  
})  
}
```

버튼 클릭시 클릭한 버튼의 지도 호출