

노랑비행

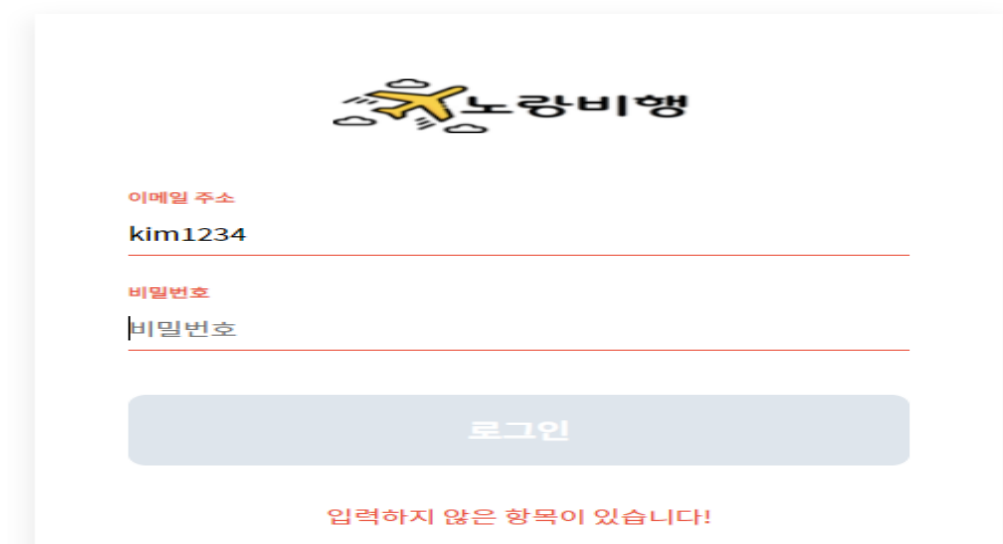
2022년 8월 5일

김종현

5조(코알라)

개인 담당 페이지

로그인 페이지



The image shows a login page for a service called '노랑비행' (Yellow Flight). At the top center is a logo featuring a yellow airplane and the text '노랑비행'. Below the logo are two input fields. The first field is labeled '이메일 주소' (Email Address) in red and contains the text 'kim1234'. The second field is labeled '비밀번호' (Password) in red and contains the text '비밀번호'. Below these fields is a light blue button with the text '로그인' (Login). At the bottom of the form, there is a red error message: '입력하지 않은 항목이 있습니다!' (There are items that have not been entered!).

- 회원가입 정책에 맞게 간단한 유효성 검사를 진행.
- '공란 존재' / '비밀번호 길이 부족' / '이메일에 @, . 누락' 을 차례로 검사하도록 설정
- 부적절한 입력을 예방하기 위해, 제출 버튼은 유효한 값을 입력해야만 활성화
- 유효하지 않은 입력에 대해서는 그에 맞는 메시지를 실시간으로 보여주어 사용자가 올바른 입력을 하도록 장려
- 카카오톡 간편 로그인 기능을 포함

```

function btnActive() {
    // 유효성 검사 --> 로그인 버튼 비/활성화
    // 이메일, 비밀번호가 공란 or 비밀번호가 8자리 이상인지
    if (
        !
        emailBtn.value &&
        pwBtn.value &&
        pwBtn.value.length > 7 &&
        emailBtn.value.search("@") !== -1 &&
        emailBtn.value.search(/[/./g) !== -1
    ) {
        // 유효하지 않으면
        login.disabled = true; // 로그인 버튼 비활성화
        login.style.backgroundColor = "#dee5ec"; // 로그인 버튼 비활성화 --> gray
        login.style.cursor = "no-drop"; // 커서 모양 변경
        login.style.color = "#ffffff";

        emailTitle.style.color = "#eb5a46"; // 유효하지 않을 때 --> 글자, 밑줄 red
        pwTitle.style.color = "#eb5a46";
        emailBtn.style.borderBottom = "2px solid #eb5a46";
        pwBtn.style.borderBottom = "2px solid #eb5a46";

        // 유효성 검사 msg는 1개만 보여주기
    }
}

```

- 회원가입 정책은 정규식을 활용하여 엄격하게 진행
- 반면, 로그인에서의 유효성 검사는 회원가입 정책이 변경됨에 따라 얼마든지 수정 가능성 존재
- 또한, 로그인에서의 지나친 유효성 검사는 사용자의 편의성에 오히려 저촉될 우려
- 따라서, 유연한 유지보수가 가능하도록 간단한 정규식과 JS 위주로 소스 코드 준비

```

pwBtn.style.borderBottom = "2px solid #c0392b";

// 유효성 검사 msg는 1개만 보여주기

if (!(emailBtn.value && pwBtn.value)) {
    // 이메일 or 비밀번호 공란
    blankMsg.style.display = "block"; // 항목이 비어있다는 msg
    shortMsg.style.display = "none";
    missingMsg1.style.display = "none";
    missingMsg2.style.display = "none";
} else if (pwBtn.value.length <= 7) {
    // 이메일과 비밀번호 모두 입력했지만, 비밀번호가 7자리 이하
    blankMsg.style.display = "none";
    shortMsg.style.display = "block"; // 비밀번호 길이 부족 msg
    missingMsg1.style.display = "none";
    missingMsg2.style.display = "none";
} else if (emailBtn.value.search("@") === -1) {
    // 이메일에 @누락
    blankMsg.style.display = "none";
    shortMsg.style.display = "none";
    missingMsg1.style.display = "block"; // 이메일 @ 누락 msg
    missingMsg2.style.display = "none";
} else if (emailBtn.value.search(/[.]/g) === -1) {
    // 이메일에 .누락
    blankMsg.style.display = "none";
    shortMsg.style.display = "none";
    missingMsg1.style.display = "none";
    missingMsg2.style.display = "block"; // 이메일 . 누락 msg
}
} else {

```

- 사용자가 혼란을 느끼지 않도록 유효하지 않은 입력값에 대해서는 그에 맞는 메시지를 1개만 보여주도록 지정

```

    }
} else {
    // 이메일, 비밀번호 공란XX and 비밀번호 8자리 이상 and 이메일에 '@','.' 포함
    login.disabled = false; // 로그인 버튼 활성화
    login.style.backgroundColor = "#ffc72c"; // 로그인버튼 활성화 --> yellow
    login.style.cursor = "pointer"; // 커서 모양 포인터로 변경
    login.style.color = "#222";
    emailTitle.style.color = "#ffc72c"; // 유효할 때 --> 글자, 밑줄 yellow
    pwTitle.style.color = "#ffc72c";
    emailBtn.style.borderBottom = "2px solid #ffc72c";
    pwBtn.style.borderBottom = "2px solid #ffc72c";

    // 유효성 검사 msg는 1개만 보여주기
    blankMsg.style.display = "none"; // 항목이 비어있는 msg --> hide
    shortMsg.style.display = "none"; // 비밀번호 길이 부족 msg --> hide
    missingMsg1.style.display = "none"; // 이메일 @ 누락 msg --> hide
    missingMsg2.style.display = "none"; // 이메일 . 누락 msg --> hide
}
}

```

- 사용자가 로그인을 하며, 입력값의 유효 여부를 직관적으로 파악할 수 있도록 글자와 테두리 색의 변화를 이용
- 부적절한 값을 사전에 방지하기 위해, 유효한 입력에 대해서만 제출버튼을 활성화

이메일(아이디) / 비밀번호 찾기 페이지

아이디 찾기
비밀번호 찾기

이름
//

휴대폰 번호
12

인증번호 발송

확인

회원가입 시 등록된 휴대폰 번호를 입력해주세요!

아이디 찾기
비밀번호 찾기

이메일
ff

확인

회원가입 시 등록된 이메일을 입력해주세요!

이메일에 '@' 문자가 누락되어 있습니다!

간편하게 로그인하기

- 로그인 페이지에서 로그인 버튼 하단의 '아이디/비밀번호 찾기'를 클릭하여 접속 가능
- 페이징 기법을 통하여, '아이디 찾기'와 '비밀번호 찾기' 기능을 한 페이지에 구현
- 이 방법을 통해 게시판, 마이페이지 등에서도 페이지 수 절약과 효율적인 정보 집약을 도모 (아래 이미지는 해외 게시판의 페이징 예시)

Japan

Tokyo



599,000 원 ~
[이국적인 풍경 속의 오감만족]

Osaka



649,000 원 ~
[다시떠나는 그리움동경]

Osaka



699,000 원 ~
[거리 곳곳 감성 가득한 동경]

Sapporo



849,000 원 ~
[자연의 온기가 가득한]

```

// 유효성 검사 -----

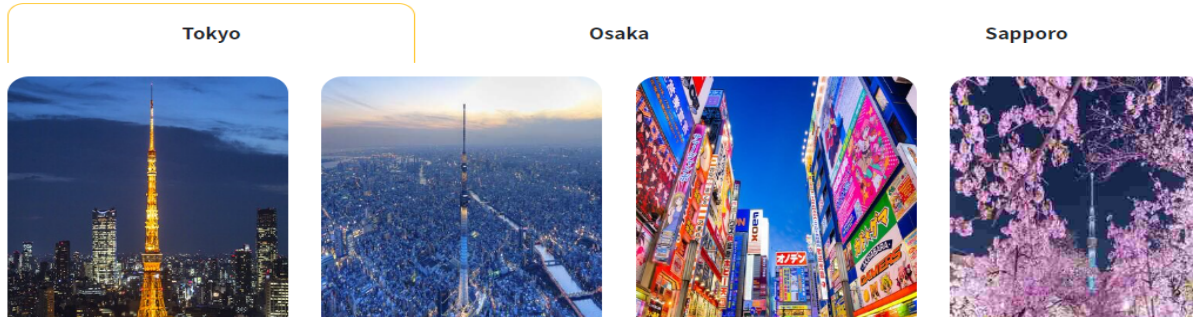
// 이메일 찾기 유효성 검사-----
function btnActive1() {
    // 유효성 검사 --> 확인 버튼 비/활성화
    // 이름, 휴대폰 번호가 공란
    if (
        !(
            nameBtn.value &&
            phoneBtn.value &&
            nameBtn.value.search(/^[a-zA-Z가-힣]/g) === -1 &&
            phoneBtn.value.search(/^[0-9]/g) === -1
        )
    ) {
        // 유효하지 않으면
        idSearchBtn.disabled = true; // 확인 버튼 비활성화
    }
}

```

- 전체적인 입력값에 대한 유효성 검사의 로직과 모습은 로그인 페이지와 유사
- 본 프로젝트에서는 아이디를 회원 등록 시 이메일로 정의
- '이메일 찾기' 기능에서 입력해야 하는 이름과 휴대폰 번호(- 제외 입력)은 각각 영문, 한글 / 숫자를 제외한 값은 입력할 수 없도록 간단한 정규식을 활용

해외 게시판

Japan



Japan

Tokyo	Osaka		Sapporo
			
399,000 원 ~ [다시만나는 너의 이름은 오사카] ◇온천호텔+1일자유◇ 오사카, 교토, 아라시야마 3/4일	679,000 원 ~ [데미가 있는 색다른 오사카] ◇테마파크+1일자유◇ 유니버설 스튜디오 제팬+교토 3일	1,099,000 원 ~ [연휴에 떠나는 오사카] ◇교토 숙박+교토 하버랜드◇ 오사카, 교토, 교토, 아라시야마 4일	549,000 원 ~ [하러함 속 김치가 가득한 오사카] ◇온천호텔+크루즈+관덴열차◇ 오사카, 교토, 교토 3일

- 여행사이트 특성 상 많은 슬라이드 배너를 포함
- 지나치게 많은 슬라이드 배너는 시각적으로 단조롭고, 사용자에게 혼란을 초래할 우려
- 따라서 JS를 활용해 화면 상으로 고정된 공간에 여러 정보를 담을 수 있는 간단한 페이지 처리
- 페이지들의 통일성을 고려해 마이페이지, 이메일/비밀번호 찾기 페이지 등에서도 재사용


```

main / webapp / js / destinationChoice(overseas).js / ...
// 이미지 확장자명 통일
// 예를 들어 확장자명 jpg이면 아래와 같이 이미지 파일명 지정
// 이미지 경로 반드시 "img/(영문국가명)(한글장소명).jpg"으로 할 것
// ex) img/france에펠탑.jpg

//프랑스 이미지 이름 --> 경로에 추가할 예정-----
const parisPlace = [
  "에펠탑(맑은하늘)",
  "에펠탑(노을)",
  "에펠탑(저녁)",
  "에펠탑(밤)",
];
const bordeauxPlace = [
  "파리전경",
  "프로방스길거리",
  "프로방스분수",
  "알렉상드르3세다리",
];
const lyonPlace = ["뤽상부르공원", "몽마르뜨언덕", "상젤리제", "에펠탑(노을)"];
// 프랑스 도시별 이미지 담을 배열
const parisImg = [];
const bordeauxImg = [];
const lyonImg = [];

// 프랑스 도시별 제목 담을 배열
const parisTitle = [];
const bordeauxTitle = [];
const lyonTitle = [];
// 프랑스 도시별 설명 담을 배열
const parisContent = [];
const bordeauxContent = [];
const lyonContent = [];

```

```

// 배열에 도시별 이미지 담기
// 인자 : (국가이름 문자열, 장소 배열, 이미지 배열)
function pushImg(country, place, imgPath) {
  for (let i = 0; i < place.length; i++) {
    imgPath.push("img/" + country + place[i] + ".jpg");
  }
}

// 프랑스 도시별 이미지 배열
pushImg("france", parisPlace, parisImg);
pushImg("france", bordeauxPlace, bordeauxImg);
pushImg("france", lyonPlace, lyonImg);
// 일본 도시별 이미지 배열

```

- JS에서 변수의 타입이 자유로운 점을 이미지 관리에 적극적으로 이용
- 이미지의 파일명 규칙(확장자 포함)을 통일
- 이미지의 파일명 중 변동되는 문자열 부분을 배열에 담아 관리
- 파일명 규칙을 통일함으로써 반복문을 통해 실제 이미지 경로들을 별도의 배열에 담아 관리
- 이를 통해, img태그에 이미지를 반복문을 통해 손쉽게 저장 가능
- 추후 수정 사항 발생 시 유지보수에도 매우 유리하다고 판단

```

*/
function showPics(
  activeBtn,
  disableBtn1,
  disableBtn2,
  pics,
  imgPath,
  beforeTitle,
  beforeContent,
  newTitle,
  newContent
) {
  activeBtn.addEventListener("click", () => {
    for (let i = 0; i < pics.length; i++) {
      // 선택된 버튼의 도시에 맞는 사진으로 변경
      // 선택된 도시에 맞는 제목 / 설명으로 변경
      pics[i].setAttribute("src", imgPath[i]);
      beforeTitle[i].innerText = newTitle[i];
      beforeContent[i].innerText = newContent[i];
    }
    //선택한 버튼 --> black, border : yellow
    activeBtn.style.border = "2px solid #ffc72c";
    activeBtn.style.borderBottom = "#ffffff";
    activeBtn.style.color = "#000000";
    // 선택하지 않은 버튼 --> gray, border : white
    disableBtn1.style.border = "2px solid #ffffff";
    disableBtn1.style.color = "#bec8d2";
    disableBtn2.style.border = "2px solid #ffffff";
    disableBtn2.style.color = "#bec8d2";
  });
}

```

```

3  */
4  function showPics(
5      activeBtn,
6      disableBtn1,
7      disableBtn2,
8      pics,
9      imgPath,
10     beforeTitle,
11     beforeContent,
12     newTitle,
13     newContent
14 ) {
15     activeBtn.addEventListener("click", () => {
16         for (let i = 0; i < pics.length; i++) {
17             // 선택된 버튼의 도시에 맞는 사진으로 변경
18             // 선택된 도시에 맞는 제목 / 설명으로 변경
19             pics[i].setAttribute("src", imgPath[i]);
20             beforeTitle[i].innerText = newTitle[i];
21             beforeContent[i].innerText = newContent[i];
22         }
23         //선택한 버튼 --> black, border : yellow
24         activeBtn.style.border = "2px solid #ffc72c";
25         activeBtn.style.borderBottom = "#ffffff";
26         activeBtn.style.color = "#000000";
27         // 선택하지 않은 버튼 --> gray, border : white
28         disableBtn1.style.border = "2px solid #ffffff";
29         disableBtn1.style.color = "#bec8d2";
30         disableBtn2.style.border = "2px solid #ffffff";
31         disableBtn2.style.color = "#bec8d2";
32     });

```

- 본 프로젝트에서 구현한 페이징 처리의 실제 원리는 클릭 이벤트를 통해 `img src`와 `text`를 변경한다는 점
- 실제 HTML 구조에 나머지 부분들이 숨겨진 것이 아니고, 처음부터 4개의 영역만 존재하고 이들의 속성 값만 변경
- 그 결과, HTML 문서에서 불필요한 반복 태그 구조들을 생략 가능
- 추후 상세 페이지로의 링크까지 추가할 경우 `a`태그를 추가한 후, 위의 함수에서 `href`의 속성을 이벤트를 통해 변경하도록 설정할 예정