

moobpl

Moving Planner Web App

- ✓ Case Study
- ✓ About Us
- ✓ Trouble Shooting
- ✓ Userflow
- ✓ Wireframes
- ✓ Design
- ✓ UI - Mobile & Components



1 Case Study

물풀(moobpl)은 이사를 준비하는 사람들을 위해 탄생한 플래너입니다.
사람들은 어떤 일을 시작하기 전 계획을 세우곤 합니다.
여행 전이나 결혼 준비를 할 때, 혹은 하루를 시작할 때도 말입니다.
내가 사는 곳이 변화되는 일, 이사.
이사를 앞둔 사람들에게는 얼마나 많은 준비가 필요할까요?
물풀은 그런 분들을 위해 출발하게 되었습니다.



01 Research

서비스 기획전 유사한 어플 및 기능, 디자인에 관해 조사하고
리서치한 어플에 부족한 기능 및 정보들을 정리하여 문제점을
보완한 서비스를 하려고 노력하였습니다.

1-1 The Problem

- ✖ 이사와 관련된 플래너 어플이 없음
- ✖ 신규 주거지 정보를 확인할 수 있는 통합 플랫폼 없음
- ✖ 인터넷 발달에 따른 정보의 다양화를 정리할 필요가 있음
- ✖ 젊은세대의 실생활에 필요한 정보 부족, 생활정보 없음

1-2 Solution

- ✓ 이사 준비에 특화된 플래너 어플리케이션
- ✓ 이사 일정 선택 & 남은 기간 계산
- ✓ 이사 지역 선택 & 지역별 맞춤 정보
- ✓ 이사 준비에 필요한 체크리스트
- ✓ 생활 맞춤 정보
- ✓ 회원가입 & 로그인 후 회원별 이사 일정 추가, 삭제

2 About Us



Position

- Project Manage
- UX & UI
- Frontend

Tech

- React
- Javascript

Kim Han Bit



Position

- Frontend & Backend
- UX & UI

Tech

- React
- Javascript
- Node js / Express js

Wi Hyang Hoon

01 Tech Stack

Front-end	React
Back-end	Express
DB	Mongo DB
Web Framework	Express

02 Groupware

git	소스코드 관리(파일 병합, 로그 관리)
gather town	화상 회의
notion	프로젝트 자료, 회의록 관리 및 기록
vscode liveshare	코드 오픈 협업

03 Convention

- 컴포넌트명 : 파스칼케이스
- 파일명 : 스네이크케이스
- 폴더명 : 스네이크케이스

04 Git flow

브랜치 구조

- master
 - dev
 - hanbit
 - hoon

각 브랜치의 역할

- master : 오류 없는 개발 완료 상태
- dev : 최종 완료 전 코드 병합, 백업용
- hanbit : 개인작업
- hoon : 개인작업

05 Library

Front-end

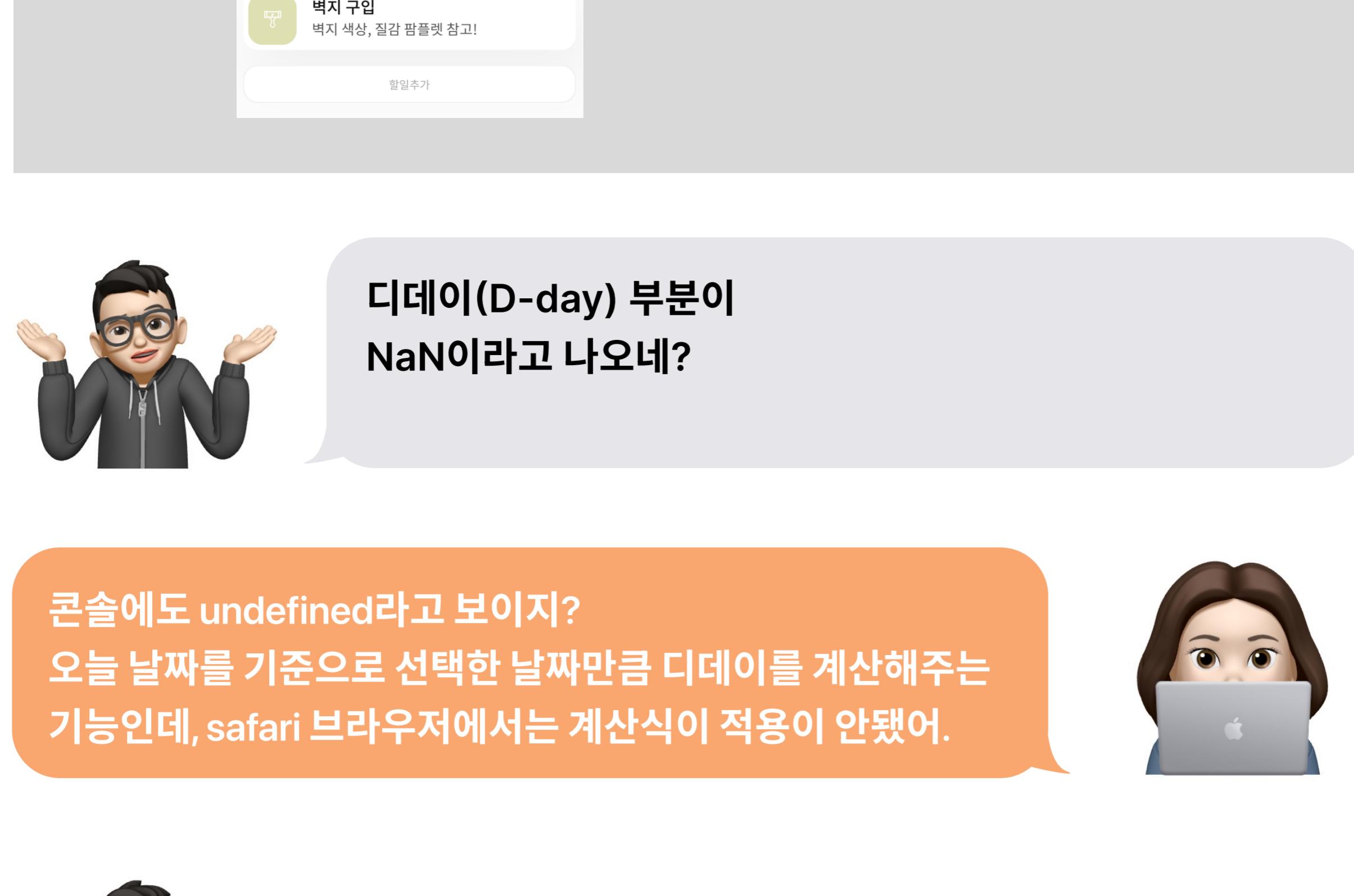
axios	서버통신
redux-toolkit	상태관리, 미들웨어
react-router-dom	화면 전환
styled-components	컴포넌트 스타일링
moment	날짜 계산, 날짜 형식 변환
date-fns	달력 기능 한국어 변환
swiper	배너 슬라이드
react-date-range	달력

Back-end

bcript	비밀번호 암호화
cookie-parser	쿠키 해석, 쿠키 객체 생성
cors	교차출처 리소스 공유
dotenv	환경변수
mongodb	데이터베이스
passport	로그인
uuid	데이터베이스 데이터 고유 아이디 생성

3 Trouble Shooting

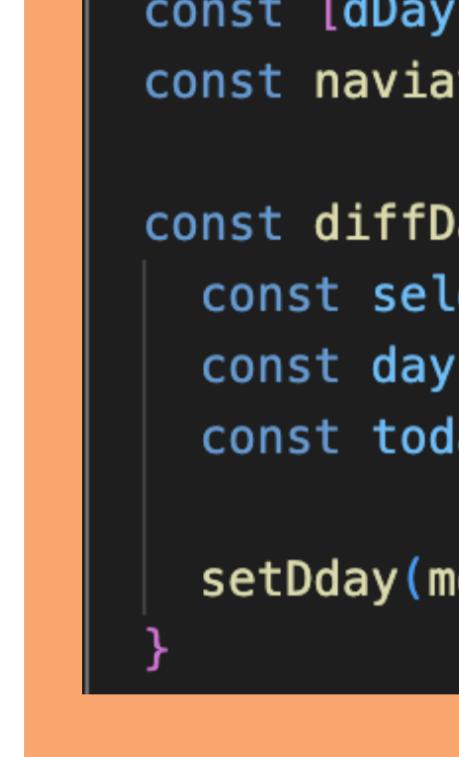
01 FrontEnd Trouble



디데이(D-day) 부분이
NaN이라고 나오네?

콘솔에도 undefined라고 보이지?

오늘 날짜를 기준으로 선택한 날짜만큼 디데이를 계산해주는
기능인데, safari 브라우저에서는 계산식이 적용이 안됐어.



그래서 어떻게 해결했어?

날짜 구현을 위해 초기에는 'day js'라는 라이브러리를 썼어.

```
import dayjs from "dayjs";
const WidgetCard = ({ data = [] }) => [
  console.log(data.date),
  const [dDay, setDday] = useState('');
  const naviate = useNavigate();

  const diffDay = () => {
    const selectDay = String(data.date).split('-').map(str => Number(str));

    const dayset = new Date();
    const today = dayjs(dayset).format('YYYY-MM-DD').split('-').map(str => Number(str));

    const todaySec = new Date(today).getTime();
    const setdaySec = new Date(selectDay).getTime();
    setDday(Math.ceil(todaySec - setdaySec) / (1000 * 60 * 60 * 24));
  }
]
```

디데이를 계산하기 위해선

1. 'day js' 라이브러리로
2. 오늘날짜 - 선택날짜를 계산해주는 함수를 만들면 되는데
safari 브라우저에서는 'day js'라이브러리의 호환과
getTime()함수의 작동이 원활하지 않은 것이 주요 버그야.

```
import moment from 'moment'; //momentjs
import 'moment/locale/ko';

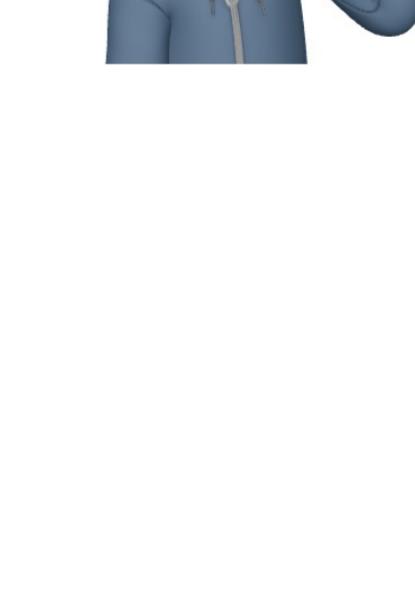
const WidgetCard = ({ data = [] }) => [
  console.log(data.date)
  const [dDay, setDday] = useState('');
  const naviate = useNavigate();

  const diffDay = () => {
    const selectDay = data.date;
    const dayset = new Date();
    const today = moment(dayset).format('YYYY-MM-DD');

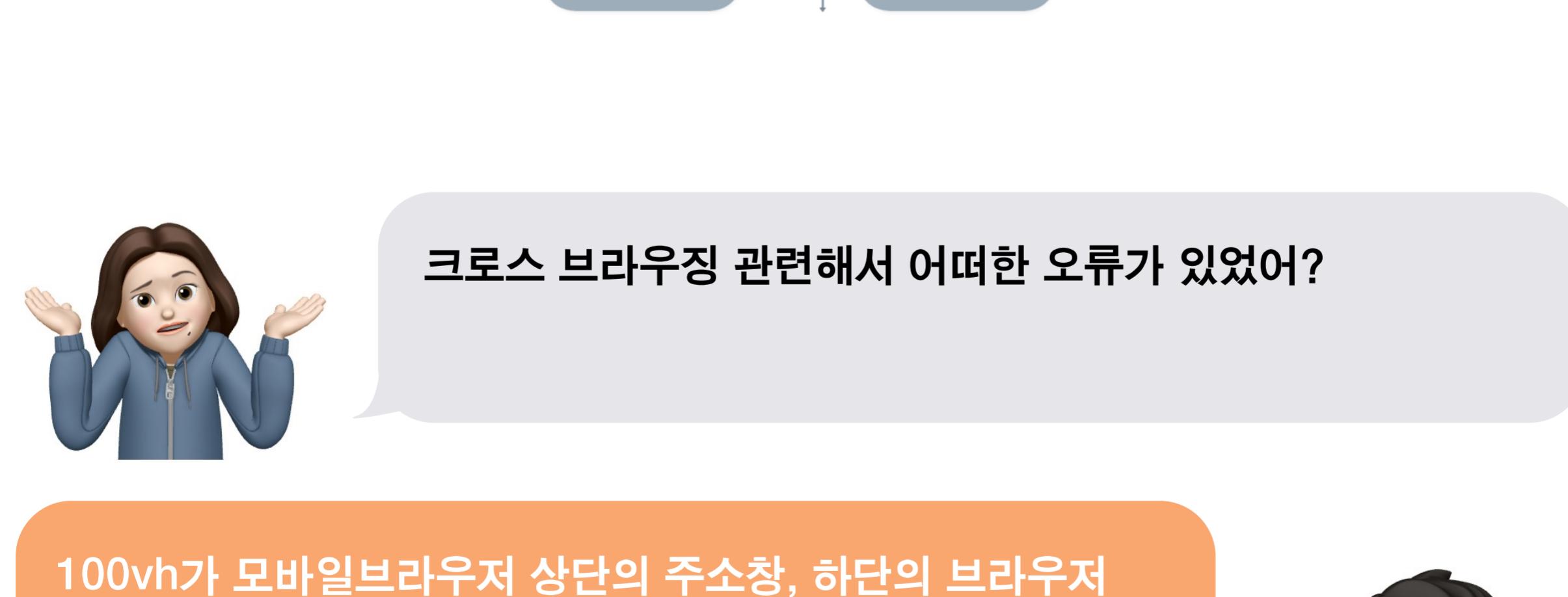
    setDday(moment(today).diff(moment(selectDay), 'days'));
  }
]
```

이 문제를 해결하기 위해

'day js' → 'moment js'로 라이브러리를 변경했고,
moment js 라이브러리에서 제공하는 기본 diff함수로 더욱
간단하게 코드를 수정하고, safari와 chrome에서 문제 없이
작동하게 됐어!



02 FrontEnd Trouble



버튼 선택 시 스타일을 줄 때,
어떤 오류가 있었던거야?

각기 다른 5가지의 카테고리 버튼 클릭 시 스타일을 적용하려
했는데, 스타일이 클릭한 버튼에만 적용되는게 아니라 버튼 전
체에 적용되는 문제가 있었어.



그래서 어떻게 해결했어?

선택여부를 알 수 있는 state 값을 통해 클릭이벤트에 조건식
을 넣었어.

1. 버튼 선택 여부를 알 수 있는 state값을 주고
2. onClick함수에는 조건식으로 버튼이 가지고 있는 아이디
값과 선택값 state가 같으면 해당 state를 반환해서
3. className에 조건으로 선택값과 버튼 아이디가 같으면
클래스를 부여하는거야!

```
{categoryList.map((item) => {
  return(
    <li
      className={selected === item.id ? "is-click" : null}
      key={item.name}
      style={{backgroundColor: `${item.color}`}}
      onClick={(event)=> {
        const {name, alt} = event.target;
        event.preventDefault();
        setCategory(name);
        setColor(alt);
        if(item.id === selected) {
          return selected;
        } else {
          setSelected(item.id);
        }
      }}
    >
```



03 FrontEnd Trouble

크로스 브라우징 관련해서 어떠한 오류가 있었어?

100vh가 모바일브라우저 상단의 주소창, 하단의 브라우저
네비게이션 바 까지 height값을 인식해서 밑에 부분이 짤
리는 현상이 있었어.

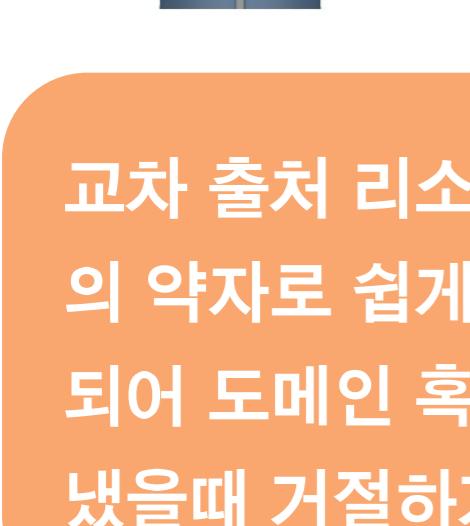
그래서 어떻게 해결했어?

```
@supports (-webkit-touch-callout: none) {
  height: -webkit-fill-available;
}
```

위에 코드를 사용하면 해결이 가능했어.
height값을 브라우저 상단의 주소창 하단의 네비게이션 바를
제외한 safearea를 height로 인식하게 해줘

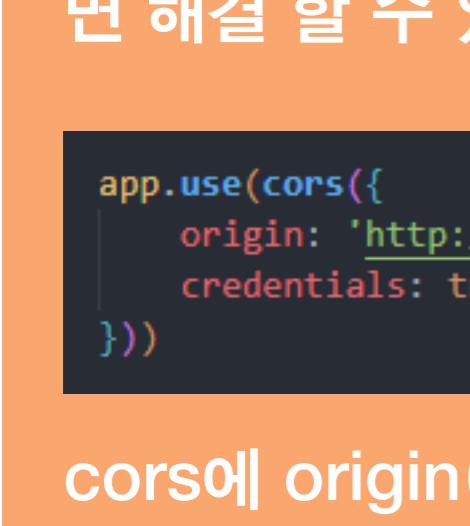
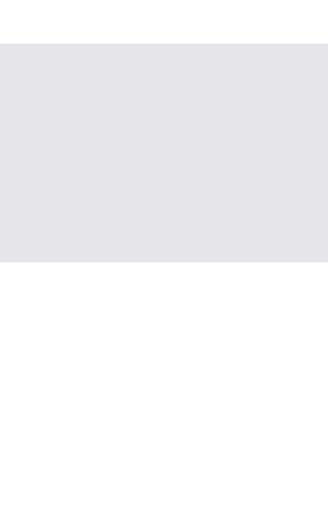
04 BackEnd Trouble

```
* Access to XMLHttpRequest at 'http://localhost:8080/login' from origin  
  'http://localhost:4200' has been blocked by CORS policy: No 'Access-  
  Origin' header is present on the requested resource.  
* ▶ POST http://localhost:8080/login net::ERR_FAILED zone-ev  
* ▶ ERROR  
  ▶ HttpErrorResponse {headers: HttpHeaders, status: 0, statusText: "Ur  
rl: "http://localhost:8080/Login", ok: false, ...}
```



CORS?
어떤 오류인거야?

교차 출처 리소스 공유 (Cross-origin resource sharing)
의 약자로 쉽게 말해 프론트서버와 백엔드 서버가 서로 분리
되어 도메인 혹은 포트번호 프로토콜이 다를경우 요청을 보
냈을때 거절하거나 동의 하는 구조를 뜻해



그래서 어떻게 해결했어?

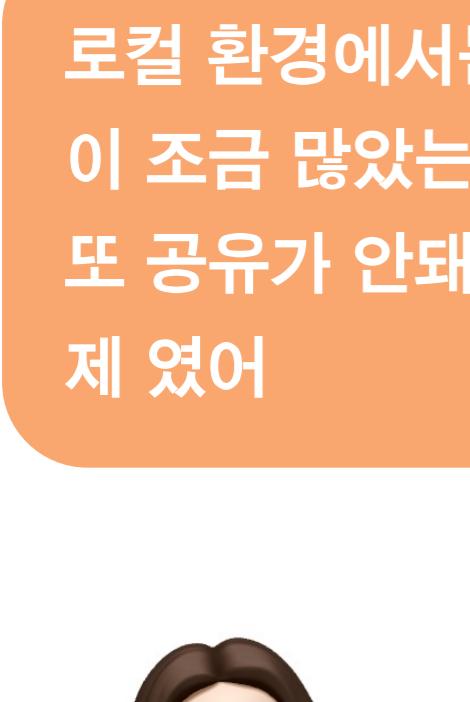
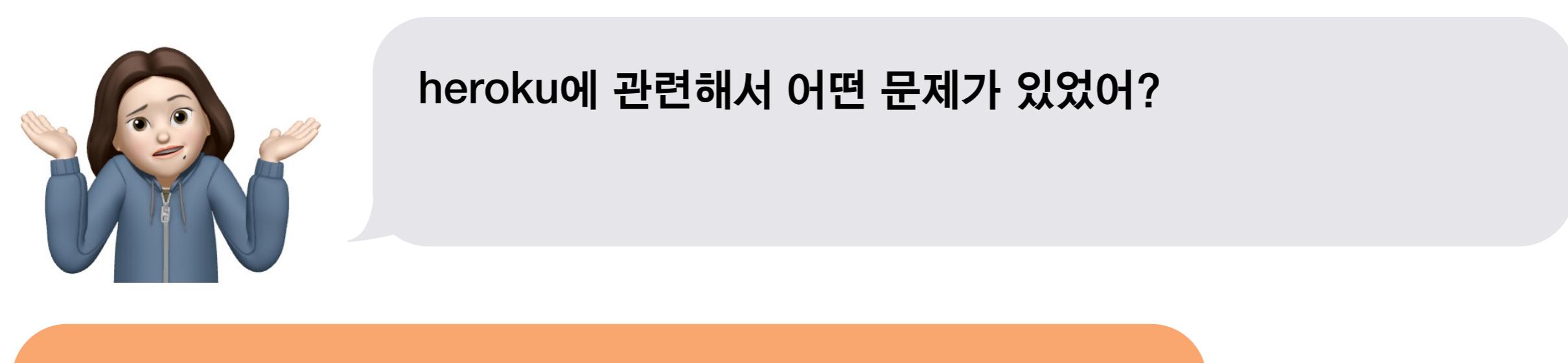
Express.js 라이브러리 중에 cors라는 라이브러리를 이용하
면 해결 할 수 있었어

```
app.use(cors({  
  origin: 'http://localhost:3000',  
  credentials: true,  
}));
```

cors에 origin이라는 key값에 요청을 허용할 URL을 적어주
면 해결이 가능했어 또, 서버가 서로 다를경우 로그인시 백엔
드에서 발행해주는 쿠키가 프론트 서버로 전달되지 않는 현
상도 있었는데 그건 credentials를 이용해서 프론트서버에
쿠키값을 공유 할 수 있도록 했어

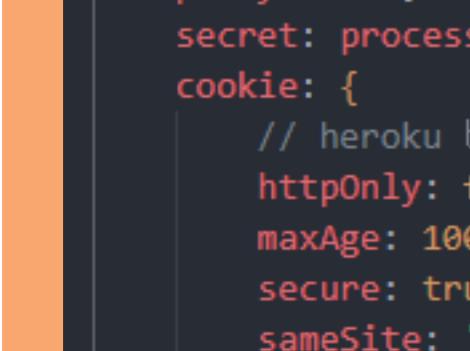
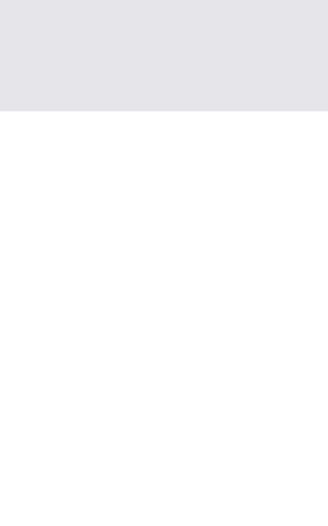


05 BackEnd Trouble



heroku에 관련해서 어떤 문제가 있었어?

로컬 환경에서는 잘되던 문제들이 배포 후 오류가 생긴것들
이 조금 많았는데 일단 첫번째로 헤로쿠에 배포한뒤 쿠키가
또 공유가 안돼는 문제가 있었어 CORS 설정과는 또다른 문
제 였어

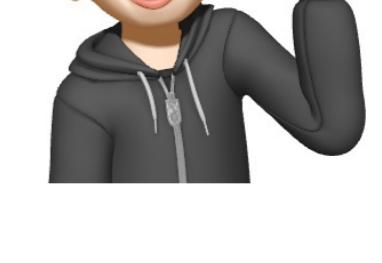


그래서 어떻게 해결했어?

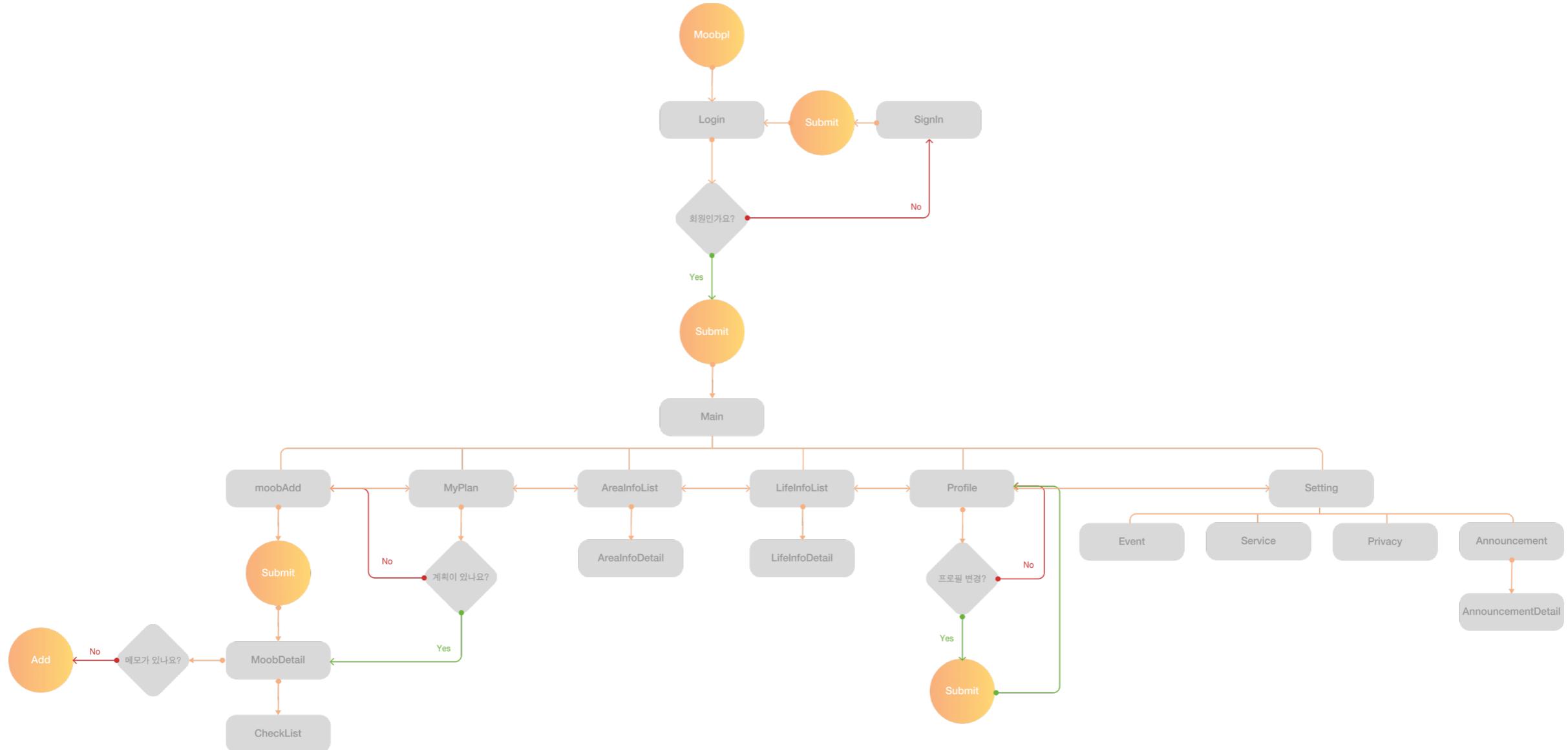
찾아본 바로 session의 쿠키옵션을 설정해주어야 한다고해

```
app.use(session({  
  saveUninitialized: false,  
  resave: false,  
  proxy: true,  
  secret: process.env.COOKIE_SECRET,  
  cookie: {  
    // heroku 배포  
    httpOnly: false,  
    maxAge: 1000 * 60 * 10,  
    secure: true,  
    sameSite: 'none',  
  }  
}));
```

또, 프론트엔드 주소와 백엔드의 주소가 도메인이 같게 설정
하는것도 방법이라고해.



4 User Flow



5 Wireframes

The wireframes illustrate the following screens:

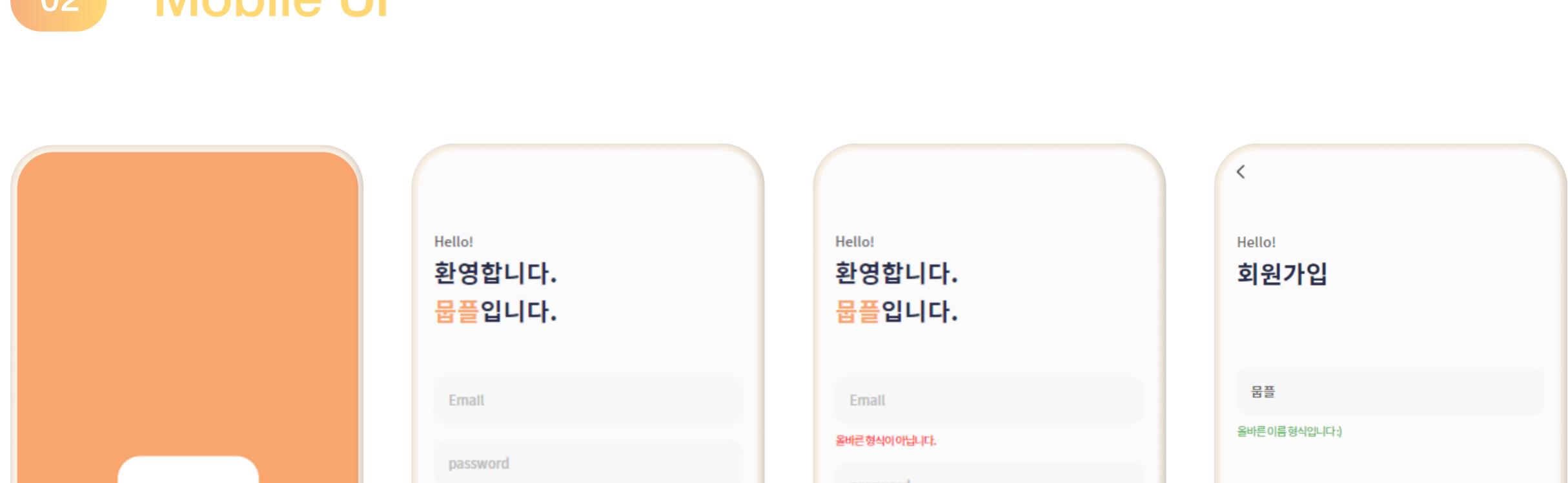
- Sign-up Screen:** Shows fields for '아이디 입력 창' (ID Input), '비밀번호 입력 창' (Password Input), and '비밀번호 확인 입력 창' (Password Confirmation Input). Buttons for 'Login' and 'sign up' are present.
- Welcome Screen:** Displays 'Hello!' and 'Welcome to 서울살이' (Welcome to SeoulLife).
- Profile Settings Screen:** Shows a placeholder for 'bigbit88' profile picture, a '내 계획 0' (0 Plans) button, and a sidebar with navigation items: 도시정보 (City Information), 생활정보 (Lifestyle Information), 이벤트 (Events), and 설정 (Settings).
- Profile Details Screen:** Displays email 'bigbit831@gmail.com' and a placeholder for '닉네임' (Nickname).
- My Profile Screen:** Shows a placeholder for '서울살이 내 계획' (SeoulLife My Profile) and a list of recent entries: '서울시 영등포구 문래동' (Seoul City Yeoongpo-gu Moneundong), '서울시 강남구 암구정동' (Seoul City Gangnam-gu Amgujeong-dong), '서울시 강남구 논현동' (Seoul City Gangnam-gu Nonhyeon-dong), and '서울시 강남구 대치동' (Seoul City Gangnam-gu Daechi-dong).
- Weather Forecast Screen:** Shows a large 'D- 190' (190 days away) and a summary card for 'D- 190' with '당일 날씨 24°C' (Today's Weather 24°C) and '체크리스트 10/10' (Checklist 10/10).
- Event Creation Screen:** Shows a placeholder for '이사, 언제 가시나요?' (When are you moving?), a '선택 완료' (Selection Complete) button, and a list of events: 'D-81 / 1.21 수' (D-81 / 1.21 수) and 'D-80 / 1.20 화' (D-80 / 1.20 화).
- Event Detail Screen:** Shows a detailed view of an event with a title like '이삿날 D- 190' and a summary card for 'D- 190'.
- Category Selection Screen:** Shows a placeholder for '제목' (Title) and '내용' (Content), with a '카테고리' (Category) section containing icons for house, people, etc.
- Event List Screen:** Shows a list of events: 'D-81 / 1.21 수' and 'D-80 / 1.20 화'.
- Event Details Screen:** Shows a detailed view of an event with a title like '하나씩, 천천히 준비해 보세요' (One by one, prepare slowly).
- Community Screen:** Shows a placeholder for '도시 정보' (City Information) and a list of cities: '영등포구 탁트인 영등포' (Yeoongpo-gu Excellent Yeoongpo), '종로구 사람 중심 명품도시 종로' (Jongno-gu People-centered Premium City Jongno), '강남구 나(ME), 너(ME), 우리...' (Gangnam-gu You(ME), I(ME), We...), '강동구 사람이 아름다운 강동' (Gangdong-gu Beautiful Gangdong), '강서구' (Gangseo-gu), and '강북구' (Gangbuk-gu).
- Setting Screen:** Shows a placeholder for '서울살이 설정' (SeoulLife Settings) and a '설정' (Settings) button.
- Information Agreement Screen:** Shows a large '정보 제목입니다.' (Information Title) and a list of sections: '제 1 조 (목적)' (Article 1 (Purpose)), '제 2 조 (회원정보의 변경)' (Article 2 (Change of member information)), '제 3 조 (개인정보보호 의무)' (Article 3 (Obligation to protect personal information)), and '제 4 조 (개인정보보호 의무)' (Article 4 (Obligation to protect personal information)).
- Service Usage Agreement Screen:** Shows a large '서비스 이용약관입니다.' (Service Usage Agreement) and a list of sections: '제 1 조 (서울살이 이용약관)' (Article 1 (SeoulLife Usage Agreement)), '제 2 조 (개인정보의 수집 및 이용)' (Article 2 (Collection and use of personal information)), '제 3 조 (개인정보의 보호 및 관리)' (Article 3 (Protection and management of personal information)), and '제 4 조 (개인정보의 제공 및 활용)' (Article 4 (Providing and utilizing personal information)).

6 Design

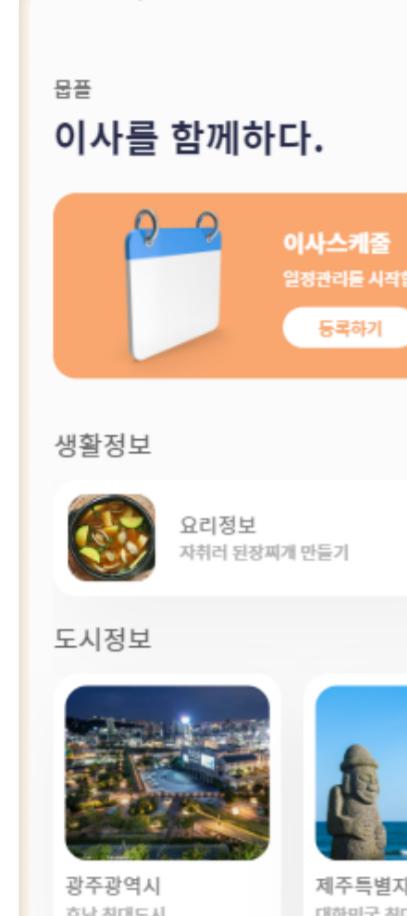
포인트 색상과 화이트 그리고 모노톤, 그림자 효과들을 활용하여 깔끔하고 심플한 느낌의 디자인을 기획하고 특히 그림자 효과를 이용해 입체적인 부분을 강조한 뮤플만의 느낌을 만들었습니다.

01 Style Guide

Colors



Typhography



Font Family :
Noto Sans KR

Styles :
Regular
Medium
Bold

Regular

이사를 준비하다. 맞춤형 이사 계획 플래너, 뮤플.

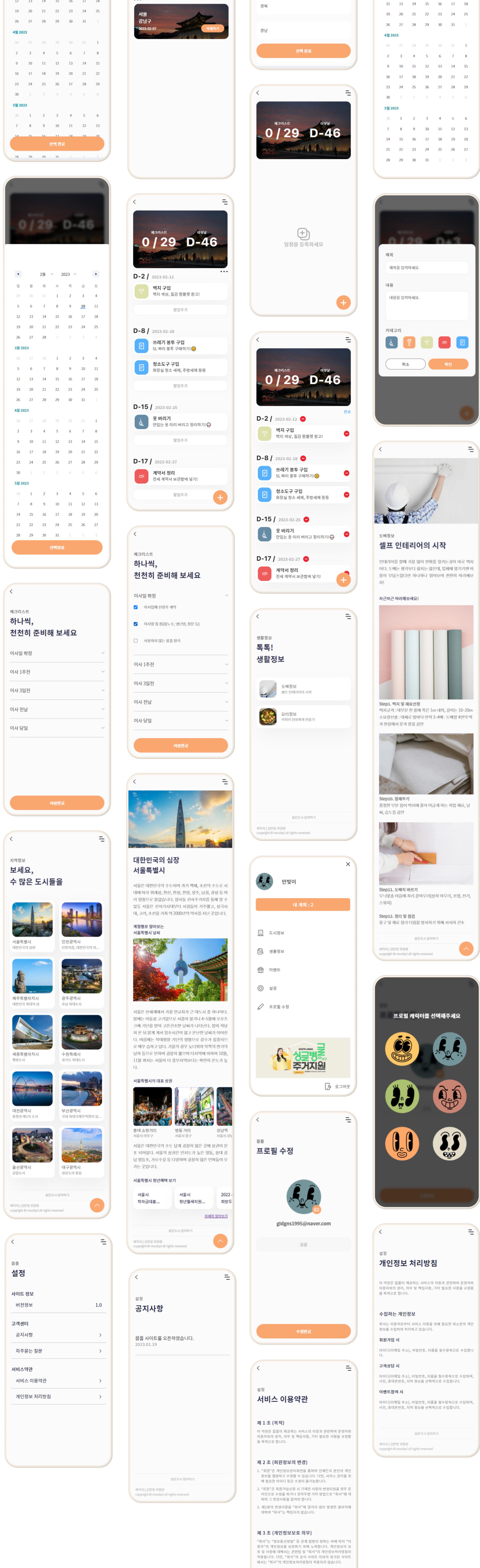
Medium

이사를 준비하다. 맞춤형 이사 계획 플래너, 뮤플.

Bold

이사를 준비하다. 맞춤형 이사 계획 플래너, 뮤플.

02 Mobile UI



03 UI Components

