



itTrip

여행 계획 웹 프로젝트

Travel Planning Web Project

Content

1. 팀원 소개
2. 개발 순서
3. 주제 선정
4. 개발 환경 구성
5. 화면 설계
6. 기능정의서
7. 테이블 설계
8. 프로젝트 구성

1. 팀원소개

강병준

메인페이지 구현, OpenAPI RESTAPI 구현
JWT활용 로그인,로그아웃 구현

김현용

DB 설계
클라이언트-서버 REST API 구현

이효용

OAuth2 소셜로그인 구현
네이버 map활용 지도 상호작용 구현

장환석

프론트엔드 전반

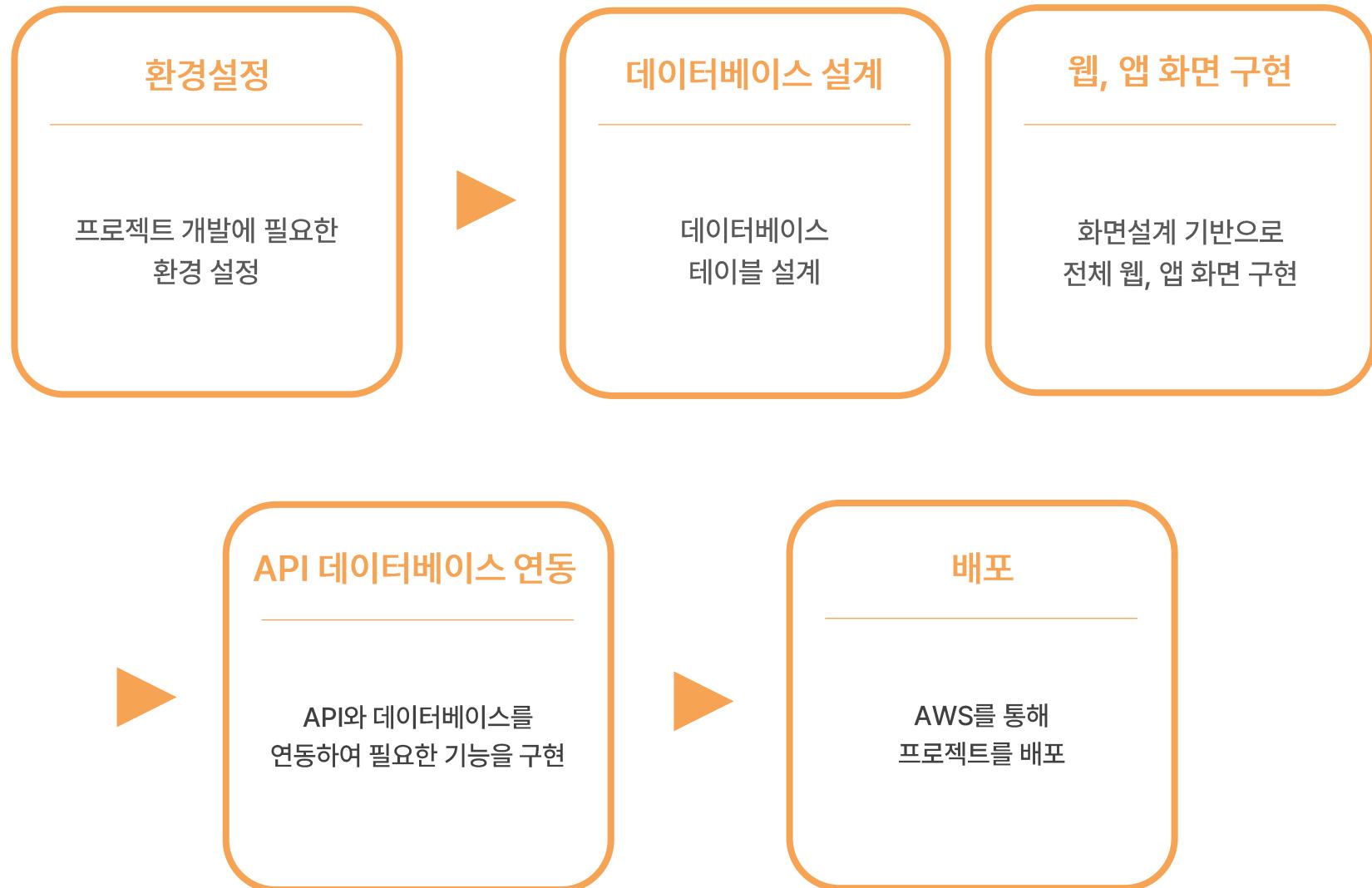
황혜진

웹 퍼블리싱 전반

2. 개발 순서



2. 개발 순서



3. 주제 선정

여행 계획

◀ 선정

날짜를 지정하여 날마다의 여행 경로 지도 표시,
준비할 것 체크리스트, 지역별 여행 추천 서비스 기능

맛집 정보

가보고 싶은 곳, 가본곳 즐겨찾기 할 수 있고,
게시판 후기 입력 및 확인 가능, 주변맛집 추천하는 기능

운동 관리

인바디 앱 정보를 가져와서 gtp가 추천해주는 운동계획으로
계획 리스트를 짜주는 기능

레시피 추천

현재 가지고 있는 재료를 입력하면
해당 재료로 만들 수 있는 레시피를 추천해 주는 기능

가계부

결제 문자내역을 붙여넣어 쉽게 기록하도록 하며,
기본적인 계산 기능과 그 기반으로 통계 자료 활용

3. 주제 선정

여행 계획 애플리케이션



Main page

각 지역별 추천관광지 및 음식 숙박업소 추천



My trip

생성한 여행에 대한 조회, 수정, 삭제



New trip

여행제목, 날짜, 여행경로, 체크리스트 생성

User

회원가입, 로그인, 소셜로그인, 마이페이지

수정관리



4. 개발 환경 구성

FrontEnd



Node



HTML



CSS



Javascript



React

BackEnd



JAVA



MySQL



Spring JPA



Spring Boot

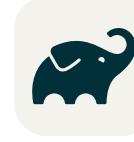
Build & Deploy



Nginx



npm



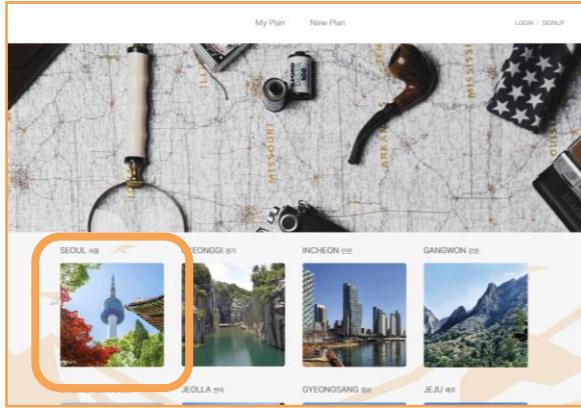
Gradle



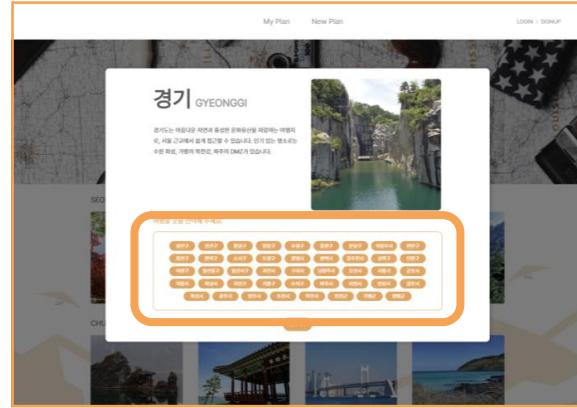
AWS

5. 화면설계

메인 페이지



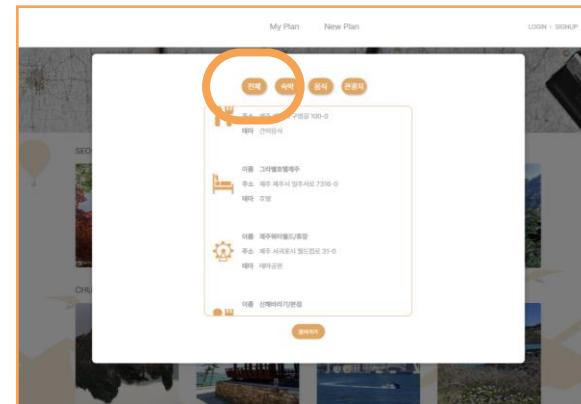
지역 정보 모달



지역 사진 클릭 시 모달창

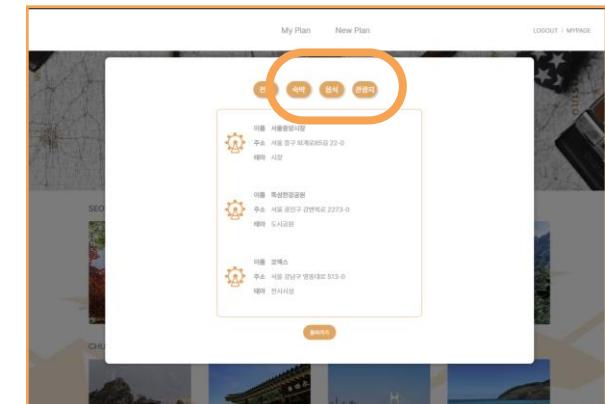
해당 지역의 시군구 정보 제공
시군구 버튼 클릭시 모달창 이동

지역 관광 정보 모달



해당 시군구의
전체 여행 정보 제공

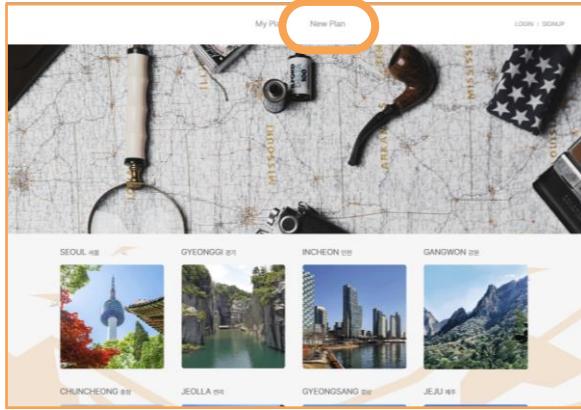
지역 관광 정보 모달(분류)



해당 시군구의
숙박, 음식, 관광지 정보 제공

5. 화면설계

메인 페이지



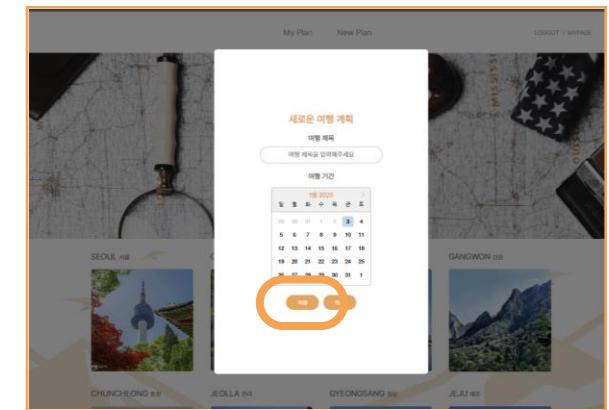
New plan 클릭시 로그인여부 확인

로그인 페이지



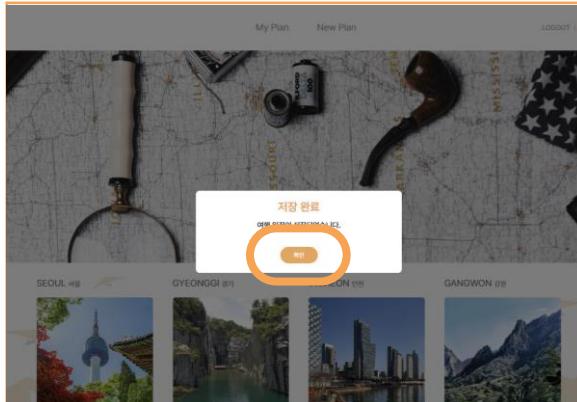
로그인 전 상태 시
자동으로 로그인 페이지 이동

날짜 모달



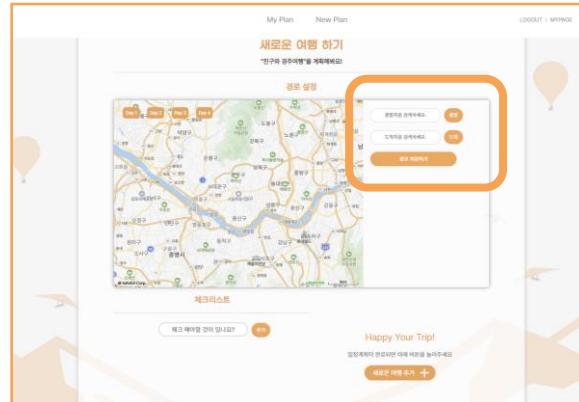
로그인 후 자동 모달 팝업,
정보입력 후 페이지 이동

저장완료



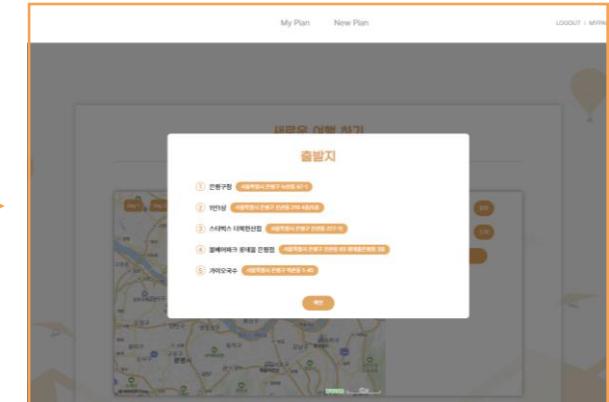
저장완료 모달창 팝업
새로운일정 페이지로 이동

새로운일정 페이지



여행 경로 출발지 경유지
도착지 입력

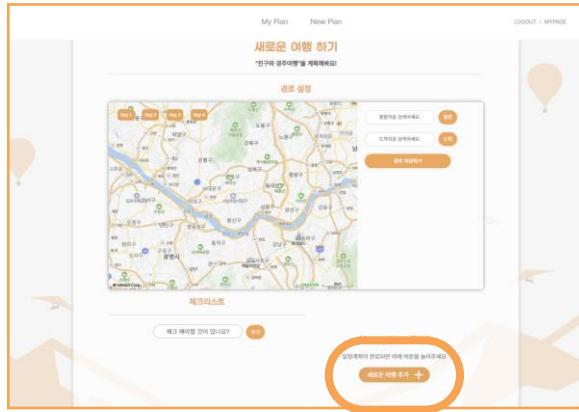
지역 검색 결과



지역 검색 결과 모달창 팝업

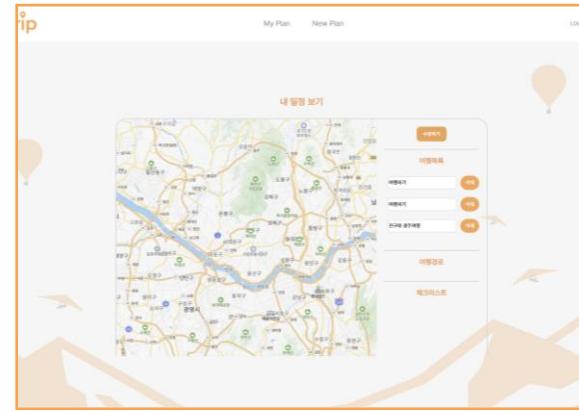
5. 화면설계

새로운일정 페이지



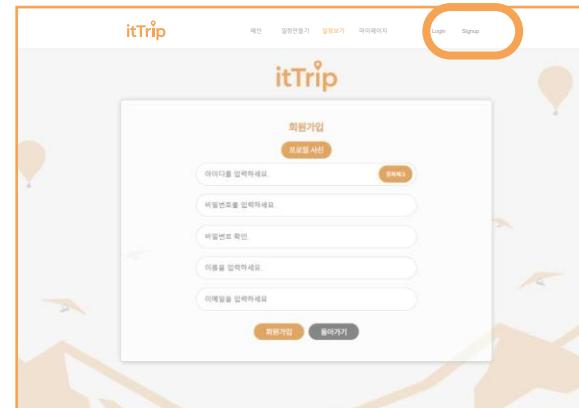
일자별 경로, 체크리스트
입력후 새로운 일정 추가 버튼

내일정 페이지



일정 저장 후
내일정 페이지로 자동 이동

회원가입



비 로그인시 Signup 클릭으로
회원가입 이동

마이페이지



로그인상태에서 마이페이지 클릭
개인정보 수정 가능

6. 기능정의서

1. 기능설명

분류	기능	설명
회원가입	프로필사진 설정	사진파일 업로드
	아이디 중복체크	아이디 유효성 확인 (고유값 확인)
	비밀번호 확인	비밀번호, 비밀번호 확인란에 일치 여부 확인
	이메일 입력란	이메일 정규식 확인
로그인 / 로그아웃	소셜로그인	Naver, Google, Kakao 로그인
	일반로그인	유효성 확인
	로그아웃	токン삭제
마이페이지	회원정보 수정	일반로그인 : 프로필사진, 비밀번호, 이름, 이메일 수정기능 (일반로그인 사용자) 소셜로그인 : 프로필사진, 이름, 이메일 수정
	회원 탈퇴	회원정보 삭제
메인페이지	8도 지역 선택 버튼	클릭한 지역에 맞는 이미지와 해당 지역 소개 시,군,구 버튼 생성
	시,군,구 선택 버튼	전체, 숙박, 음식, 관광지 버튼 생성 시,군,구마다 한국관광공사 API에서 월별 선정한 음식, 숙박, 관광지 60개 추천
	비로그인시 관리	상단 로그인페이지, 회원가입 페이지 버튼 기능 제공
	로그인시 관리	상단 로그아웃페이지, 마이페이지 버튼 기능 제공
Header	My Plan	생성한 여행들을 조회, 수정, 삭제가 가능한 페이지로 이동
	New Trip	새로운 여행 생성 페이지로 이동

6. 기능정의서

1. 기능설명

분류	기능	설명
New Trip	여행기간, 여행제목 입력 모달	여행제목, 출발날짜, 도착날짜 지정 모달 렌더링
	날짜 별 지도생성	서울시청 중심으로 맵 렌더링, 여행기간 만큼의 DAY 버튼 생성
	DAY 버튼	여행기간만큼의 생성된 DAY 버튼 클릭 후 각 DAY별 경로 지정 DAY별 미저장시 경고창 발생
	경로지정	출발지, 경유지, 도착지의 검색어 입력 후 버튼 클릭시 주소 검색 결과 모달 렌더링 출발지, 경유지, 도착지 비설정시 입력요청 경고창 발생
	주소 검색 결과 모달	지역 검색 API 결과 5가지 도출 : 상호명, 주소 표시 주소 클릭시 출발지, 경유지, 도착지 지정 기능
	Check List	여행 시 준비사항 추가 및 체크, 삭제 기능
	새로운 여행 추가 버튼	현재 창의 모든 여행정보를 저장 DAY별 미저장시 경고창 발생
My Plan	날짜별 지도 정보	저장된 여행 정보를 중심으로 맵 렌더링 여행기간만큼의 DAY버튼 생성
	수정하기 버튼	버튼 클릭시 여행 제목, DAY별 경로, 체크리스트 추가 수정 및 삭제 기능
	여행목록	New Trip에서 생성한 여행 계획의 제목별 리스트 생성
	체크리스트 렌더링	수정, 삭제 버튼 클릭 시 추가, 수정, 삭제 기능

6. 기능정의서

2. 기능구현

분류	기능	구현
회원가입	클라이언트 데이터 처리	<p>컨트롤러: @PostMapping("/signup") 클라이언트에서 전송된 회원가입 데이터를 처리합니다. @RequestParam을 사용하여 아이디, 비밀번호, 이름, 이메일, 프로필 사진을 각각 받아옵니다.</p> <p>UserDTO 객체를 생성하고, 서비스 계층의 create() 메서드를 호출하여 비밀번호는 BCryptPasswordEncoder를 사용해 암호화한 후 저장합니다.</p> <p>사진파일 처리 및 저장은 서버의 현재작업 디렉토리를 기준으로 uploads/profilePhotos 경로를 생성합니다. 디렉토리가 존재하지 않으면 mkdirs()를 사용하여 디렉토리를 생성합니다. 업로드된 파일의 이름이 중복되지 않도록 현재 시간을 기반으로 고유한 파일 이름을 생성합니다.</p> <p>MultipartFile의 transferTo() 메서드를 사용하여 파일을 지정된 디렉토리에 저장합니다. 저장된 파일은 /uploads/profilePhotos/{파일명}의 경로로 접근할 수 있도록 설정됩니다. 모든 데이터는 암호화 및 검증 과정을 거쳐 안전하게 데이터베이스에 저장됩니다.</p> <p>성공 또는 실패 여부에 따라 적절한 HTTP 응답을 반환하여 클라이언트와 상호작용합니다.</p>
	아이디 중복체크	<p>컨트롤러: @GetMapping("/check") 클라이언트로부터 전달받은 아이디를 쿼리 파라미터로 받아옵니다.</p> <p>서비스 계층의 duplicate() 메서드를 호출하여 아이디 중복 여부를 확인합니다. 중복일 경우 true, 중복이 아닐 경우 false를 반환합니다.</p>

6. 기능정의서

2. 기능구현

분류	기능	구현
로그인	일반사용자 로그인	<p>컨트롤러: @PostMapping("/signin") 클라이언트가 입력한 아이디와 비밀번호를 JSON 형태로 받아옵니다.</p> <p>서비스 계층의 getByCredentials() 메서드를 호출하여 사용자 ID를 조회합니다 . 존재하지 않으면 UsernameNotFoundException을 발생시킵니다. 입력된 비밀번호와 데이터베이스에 저장된 암호화된 비밀번호를 비교합니다. 비밀번호가 일치하면 JWT 토큰을 생성하고 반환합니다. 비밀번호가 일치하지 않으면 예외를 발생시킵니다.</p> <p>로그인 성공 시 사용자 ID, 이름, 이메일, 프로필 사진, 그리고 JWT 토큰을 포함한 응답을 반환하고 실패시 에러메세지를 반환합니다 .</p>
	소셜로그인 사용자 로그인	<p>Spring Security가 OAuth2 인증 요청을 가로채고, 해당 소셜 제공자(카카오, 네이버, 구글)로 인증 요청을 보냅니다. 이과정에서 Spring Security가 자동으로 OAuth2AuthorizationRequestRedirectFilter를 통해 소셜 제공자 인증 URL로 리다이렉트합니다. 소셜 제공자가 인증을 성공적으로 처리하면 사용자 정보를 포함한 응답을 Spring Boot로 반환합니다.</p> <p>Spring Security가 소셜 제공자로부터 받은 인증코드를 사용해 엑세스 토큰을 요청합니다 . 엑세스토큰을 통해 사용자 정보를 가져옵니다 . 이단계에서 OAuthUserServiceImpl 클래스(DefaultOAuth2UserService를 확장한 클래스)의 loadUser() 메서드가 호출되어 사용자 정보를 처리하고 데이터베이스에 저장하거나 업데이트합니다 ApplicationOAuth2User 객체를 생성하여 사용자의 정보를 Spring Security의 인증컨텍스트에 저장합니다 .</p> <p>인증이 성공하면 OAuthSuccessHandler가 호출되고 JWT 토큰을 생성하고 클라이언트를 지정된 리다이렉트 URL (window.location.origin(http://localhost:3000) react에서 명시되어있는 url) 로 리다이렉트됩니다</p>

6. 기능정의서

2. 기능구현

분류	기능	구현
My Trip	여행목록 표시	<p>컨트롤러: @GetMapping("/trips") New Trip을 누르게 되면 초기설정 되어있는 맵과 get매핑을 통해 가져온 여행title이 목록에 표시한다.</p> <p>tripController에 @GetMapping("/trips")을 이용해서 @AuthenticationPrincipal 어노테이션을 통해 userId를 가져와서 tripService의 getTrips에서 tripRepository의 getTripsByUserId를 통해 받아온 userId를 가지고 trip 정보를 가져온다.</p>
	여행목록에 여행 title 클릭시 여행정보 표시	<p>컨트롤러: @GetMapping("/maps/{tripIdx}"), @GetMapping("/checklist/{tripIdx}") 클릭시 GetMapping maps를 통해 tripIdx에 맞는 지도에는 day의 마커와 폴리라인 그리고 소요시간과 여행거리 를 표시해주고 GetMapping checklist를 통해 tripIdx에 맞는 체크리스트들을 보여준다.</p> <p>tripController에 @GetMapping("/maps/{tripIdx}") 을 이용해서 @AuthenticationPrincipal 어노테이션을 통해 userId 와 @PathVariable의 tripIdx를 가져와서 tripService의 getMaps에서 userId와 tripIdx를 가지고 mapRepository의 getLocation을 통해 위치정보를 가져온다.</p> <p>tripController에 @GetMapping(" /checklist/{tripIdx} ") 을 이용해서 @AuthenticationPrincipal 어노테이션을 통해 userId 와 @PathVariable의 tripIdx를 가져와서 tripService의 getCheckLists에서 userId와 tripIdx를 가지고 checkListRepository의 getCheckListByIdAndTripIdx를 통해 체크리스트를 가져와 entit로 만들어 tripService의 parseItems에 entit의 getItems를 사용하여 빈 문자열이면 바로 반환해 주고 아니라면 ,를 기준으로 항목을 나눈 뒤 :로 구분해서 id, text, checked값을 추출하여 Items 객체 생성 후 리스트에 추가해 준다.</p>

6. 기능정의서

2. 기능구현

분류	기능	구현
My Trip	수정하기 버튼	버튼을 누르면 ReadOnly가 풀리고 수정하기 버튼이 수정완료 버튼으로 교체되면서 여행경로와 체크리스트들을 수정할 수 있게 AddData로 바뀌면서 고치고싶은 여행경로와 체크리스트들을 수정할 수 있다.
	수정완료 버튼	컨트롤러: @PutMapping("/maps"), @PutMapping("/checklist") 버튼을 누르면 put매핑 maps에는 수정된 mapObject put매핑 checklist에는 수정된 checklist를 DB 테이블에 저장하게 된다. tripController에 @PutMapping("/maps")을 이용해서 @AuthenticationPrincipal 어노테이션을 통해 userId 와 @RequestBody MapDTO dto를 가져와서 tripRepository의 getByIdx dto의 Id를 가져와서 trip저장하고 userRepository의 findById를 통해 Id를 가져와 user에 저장하고 tripService의 getMaps에 userId와 tripId를 가지고 mapRepository의 getLocation을 통해 위치기반 정보를 가져오고 getMapObject를 이용하여 mapList사이즈와 getMapObject의 사이즈를 비교하여 같다면 day에 맞는 mapObject에 days, startPlace, startAddress, startPoint, goalPlace, goalAddress, goalPoint를 추가하고 wayPoint가 있다면 wayPoint도 추가하여 currentEntity에 user, trip, days, startPlace, startAddress, startPoint, goalPlace, goalAddress, goalPoint, wayPoints를 set하여 mapRepository의 save를 통해 DB에 저장한다.
		tripController에 @PutMapping("/checklist")을 이용해서 @AuthenticationPrincipal 어노테이션을 통해 userId 와 @RequestBody CheckListDTO dto를 가져와서 tripRepository의 getByIdx dto의 Id를 가져와서 trip저장하고 userRepository의 findById를 통해 Id를 가져와 user에 저장하고 tripService의 getCheckLists에 userId와 tripId를 가지고 checkListRepository의 getCheckListByUserIdAndTripIdx를 통해서 checkList를 가져오고 items에 추가하여 user, trip, items를 set하여 checkListRepository의 save를 통해 DB에 저장한다.
	삭제버튼	컨트롤러: @DeleteMapping("/trip/{idx}") tripController에 @PathVariable의 idx를 가져와서 tripRepository의 deleteById를 통해 DB에서 삭제된다.

6. 기능정의서

2. 기능구현

분류	기능	구현
New Trip	로그인 리다이렉션	newtrip클릭시 로그인이 되어있는지 확인하고 로그인이 되어 있지 않다면 login페이지로 돌려보내 로그인 후 이용할 수 있게 한다.
	여행기간 입력 모달	로그인이 되어 있는 것을 확인하면 여행제목과 여행기간을 입력하는 모달창이 나오고 저장버튼을 누르면 여행제목과 여행기간을 입력되어 있는지를 확인하고 newtrip페이지로 이동하게 된다.
	출발지 검색	<p>컨트롤러: @GetMapping("/local") 클라이언트가 검색창에 검색어를 입력하고 출발 버튼을 입력하면 검색어를 query로 받아오게 됩니다.</p> <p>네이버 developers의 검색 api에 있는 지역검색 open api를 이용하여 query에 검색어(query)와 검색 결과 개수(display)와 검색시작위치(start)와 검색결과 정렬방법(sort)을 header에는 clientId, clientSecret를 보내어 검색결과를 가지고 온 후 filter를 이용하여 "번출구"가 속해 있는 결과를 제외 후에 검색결과를 보여주게 되고 지번 주소를 클릭하게 되면 검색결과가 검색창에 주소가 나오게 된다. Map.js에 네이버 지도 api에 있는 geocode를 이용하여 지번주소를 좌표로 변환하여 저장한다.</p>
	도착지 검색	<p>컨트롤러: @GetMapping("/local") 클라이언트가 검색창에 검색어를 입력하고 도착 버튼을 입력하면 검색어를 query로 받아오게 됩니다.</p> <p>네이버 developers의 검색 api에 있는 지역검색 open api를 이용하여 query에 검색어(query)와 검색 결과 개수(display)와 검색시작위치(start)와 검색결과 정렬방법(sort)을 header에는 clientId, clientSecret를 보내어 검색결과를 가지고 온 후 filter를 이용하여 "번출구"가 속해 있는 결과를 제외 후에 검색결과를 보여주게 되고 지번 주소를 클릭하게 되면 검색결과가 검색창에 주소가 나오게 된다. Map.js에 네이버 지도 api에 있는 geocode를 이용하여 지번주소를 좌표로 변환하여 저장한다.</p>
	경유지 검색	<p>컨트롤러: @GetMapping("/local") 클라이언트가 검색창에 검색어를 입력하고 경유 버튼을 입력하면 검색어를 query로 받아오게 됩니다.</p> <p>네이버 developers의 검색 api에 있는 지역검색 open api를 이용하여 query에 검색어(query)와 검색 결과 개수(display)와 검색시작위치(start)와 검색결과 정렬방법(sort)을 header에는 clientId, clientSecret를 보내어 검색결과를 가지고 온 후 filter를 이용하여 "번출구"가 속해 있는 결과를 제외 후에 검색결과를 보여주게 되고 지번 주소를 클릭하게 되면 검색결과가 검색창에 주소가 나오게 된다. Map.js에 네이버 지도 api에 있는 geocode를 이용하여 지번주소를 좌표로 변환하여 저장한다. 삭제 버튼을 누르면 삭제도 가능하다.</p>

6. 기능정의서

2. 기능구현

분류	기능	구현
New Trip	경로저장하기	<p>컨트롤러: @GetMapping("/directions/withwaypoint") 만약 출발지나 도착지나 경유지의 주소가 비어있게 되면 경고하는 모달창이 나와 출발지와 도착지 그리고 경유지를 꼭 입력하게 해준다.</p> <p>경로 저장하기 버튼을 누르게 되면 putObject가 되면서 days, startPlace, startAddress, startPoint, goalPlace, goalAddress, goalPoint, wayPoints가 MapObject에 값이 저장되게 된다.</p> <p>Map.js에서 mapObject를 찾아서 day에 맞는 데이터를 foundData에 저장하여 출발지의 title에는 startPlace, address에는 startAddress, latlng에는 startPoint를 저장하고 도착지의 title에는 goalPlace, address에는 goalAddress, latlng에는 goalPoint를 저장하고 경유지에는 wayPoints들을 저장하게된다.</p> <p>이렇게 저장된 startPoint와 goalPoint와 wayPoint를 이용하여 네이버 맵에 마커를 찍게하고 각각 start,goal,waypoints에 담아 params로 보내 duration(소요시간)과 distance(여행거리)와 path를 가지고와서 path를 이용하여 네이버 맵에 폴리라인을 그려주면서 마커와 폴리라인이 한눈에 보이도록 해주고 소요시간과 여행거리를 표현해준다.</p>
	day이동하기	<p>만약 경로 저장하기를 누르지 않고 day를 이동하려고 하면 "저장이 안된 일정이 있습니다. 넘어가시겠습니까?"라는 모달창이 나오게 되며 경고를 하게 해준다.</p> <p>경로 저장하기를 눌렀다면 자연스럽게 다음 day로 넘어가면서 map에 있는 마커, 폴리라인, 소요시간, 여행거리를 초기화 해주며 출발지 도착지 경유지도 빈칸으로 초기화 시켜준다.</p>
	체크리스트	<p>만약 경로 저장하기를 누르지 않고 day를 이동하려고 하면 "저장이 안된 일정이 있습니다. 넘어가시겠습니까?"라는 모달창이 나오게 되며 경고를 하게 해준다.</p> <p>경로 저장하기를 눌렀다면 자연스럽게 다음 day로 넘어가면서 map에 있는 마커, 폴리라인, 소요시간, 여행거리를 초기화 해주며 출발지 도착지 경유지도 빈칸으로 초기화 시켜준다.</p>

6. 기능정의서

2. 기능구현

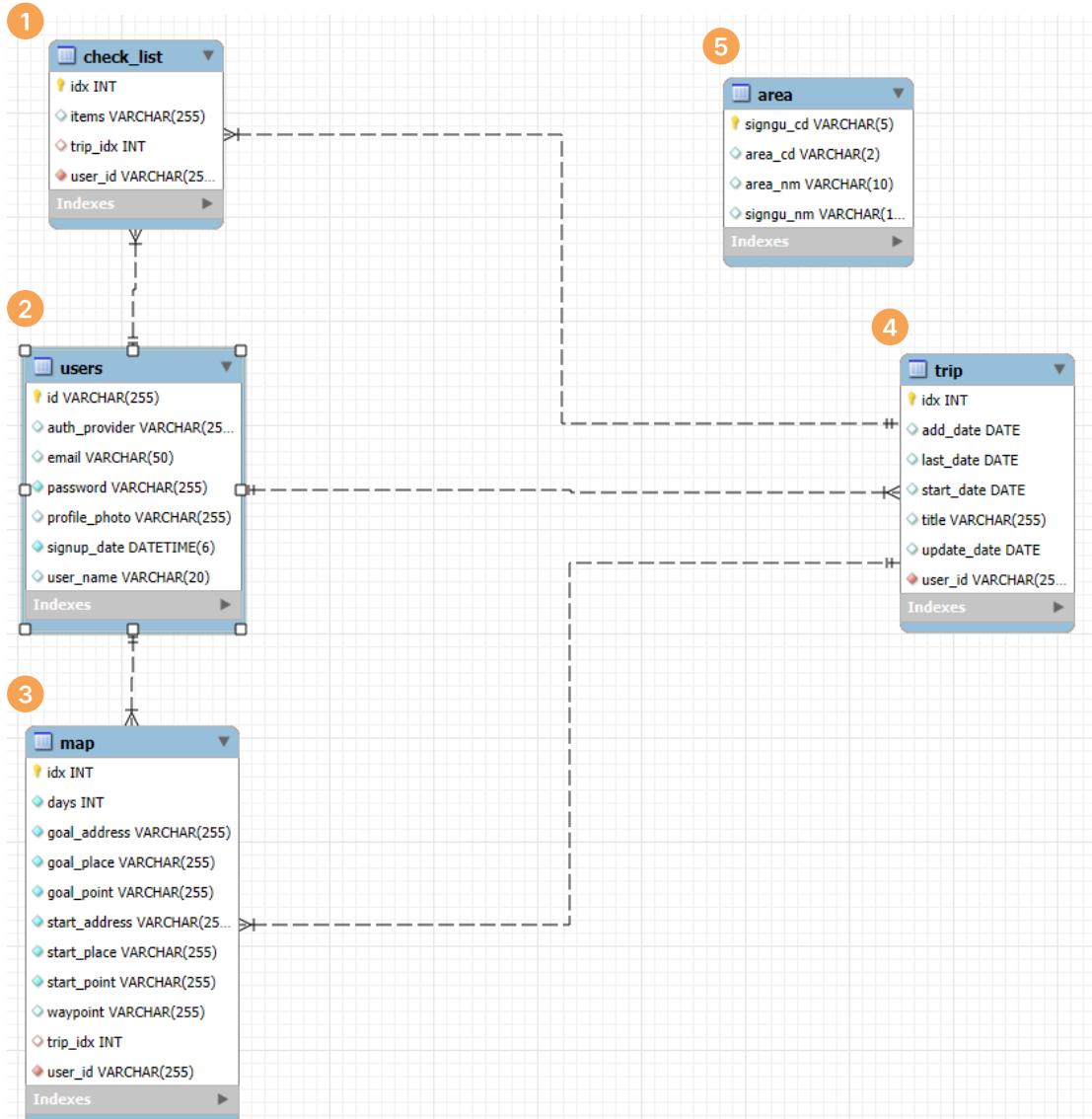
분류	기능	구현
마이페이지	마이페이지 정보 조회	<p>컨트롤러: @GetMapping("/mypage") 클라이언트(React)는 /mypage 엔드포인트로 GET 요청을 보냅니다. 요청 헤더에는 JWT 토큰이 포함되어 있으며, 이를 통해 인증된 사용자임을 증명합니다. Spring Security는 요청 헤더의 JWT 토큰을 검증, 인증이 성공하면 SecurityContextHolder에 인증 객체를 저장합니다.</p> <p>컨트롤러는 @AuthenticationPrincipal 어노테이션을 사용하여 현재 로그인된 사용자의 ID를 가져옵니다.</p> <p>컨트롤러는 서비스 계층의 getById() 메서드를 호출, 해당 사용자 ID를 기반으로 데이터베이스에서 사용자 정보를 조회합니다.</p> <p>클라이언트 응답: 조회된 사용자 정보를 JSON 형식으로 클라이언트(React)에 반환합니다.</p> <p>동작 흐름 요약</p> <p>클라이언트가 /mypage 엔드포인트로 GET 요청을 보냅니다. 요청 헤더에 포함된 JWT 토큰이 Spring Security에 의해 검증됩니다. 인증 성공 시, 컨트롤러는 @AuthenticationPrincipal로 사용자 ID를 가져옵니다. 서비스 계층의 getById() 메서드를 호출하여 사용자 정보를 조회합니다. 조회된 정보를 JSON 형식으로 클라이언트에 반환합니다.</p>
	회원정보 수정	<p>클라이언트에서 "내정보 수정" 버튼 클릭 시, 입력된 데이터를 기반 /user 엔드포인트에 PUT 요청합니다. 요청 데이터는 FormData 객체로 구성되며, 다음 항목을 포함합니다: 사용자 ID (읽기 전용). , 수정된 이름, 이메일. , 비밀번호 (일반 로그인 사용자만 해당). , 새로 업로드된 프로필 사진 파일.</p> <p>요청 헤더에는 JWT 토큰이 포함되어, Spring Security는 이를 검증, 인증된 사용자임을 확인합니다. 인증이 성공하면 SecurityContextHolder에 저장된 로그인된 사용자의 ID를 가져옵니다.</p> <p>컨트롤러는 서비스 계층의 modify() 메서드를 호출하여 데이터베이스에서 기존 사용자 정보를 조회하고, 요청 데이터를 기반으로 업데이트합니다. 이름과 이메일은 직접 업데이트됩니다. , 비밀번호는 일반 로그인 사용자만 암호화하여 저장합니다. 프로필 사진은 서버의 로컬 디렉토리에 저장되며, 파일 경로가 데이터베이스에 반영됩니다.</p> <p>응답 처리: 수정 성공 시 상태 코드 200과 성공 메시지 반환, 클라이언트는 메인페이지로 리다이렉션</p>

6. 기능정의서

2. 기능구현

분류	기능	구현
마이페이지	회원정보 삭제	<p>클라이언트에서 "회원탈퇴" 버튼을 클릭하면 /user 엔드포인트에 DELETE 요청을 보냅니다. 요청 헤더에는 JWT 토큰이 포함되어 있으며, 이를 통해 인증된 사용자임을 증명합니다.</p> <p>Spring Security는 요청 헤더에 포함된 JWT 토큰을 검증하여 인증된 사용자임을 확인합니다. 인증이 성공하면 SecurityContextHolder에 저장된 로그인된 사용자의 ID를 가져옵니다.</p> <p>컨트롤러는 서비스 계층의 delete() 메서드를 호출하여 사용자 데이터를 삭제합니다. delete() 메서드는 데이터베이스에서 사용자 ID를 기반으로 해당 UserEntity를 조회합니다. 조회된 엔티티를 데이터베이스에서 삭제합니다.</p> <p>삭제 성공 시 상태 코드 200과 함께 "삭제 성공" 메시지를 반환합니다. 클라이언트는 이를 통해 성공 메시지를 표시하고 로그인 페이지로 리다이렉션됩니다.</p>

7. 테이블 설계



1

Column Name	Data Type	Nullable	Key	Max Length
idx	int	NO	PRI	NULL
items	varchar	YES		255
trip_idx	int	YES	UNI	NULL
user_id	varchar	NO	MUL	255

2

Column Name	Data Type	Nullable	Key	Max Length
id	varchar	NO	PRI	255
auth_provider	varchar	YES		255
email	varchar	YES		50
password	varchar	NO		255
profile_photo	varchar	YES		255
signup_date	datetime	NO		NULL
user_name	varchar	YES		20

3

Column Name	Data Type	Nullable	Key	Max Length
idx	int	NO	PRI	NULL
days	int	NO		NULL
goal_address	varchar	NO		255
goal_place	varchar	NO		255
goal_point	varchar	NO		255
start_address	varchar	NO		255
start_place	varchar	NO		255
start_point	varchar	NO		255
waypoint	varchar	YES		255
trip_idx	int	YES	MUL	NULL
user_id	varchar	NO	MUL	255

4

Column Name	Data Type	Nullable	Key	Max Length
idx	int	NO	PRI	NULL
add_date	date	YES		NULL
last_date	date	YES		NULL
start_date	date	YES		NULL
title	varchar	YES		255
update_date	date	YES		NULL
user_id	varchar	NO	MUL	255

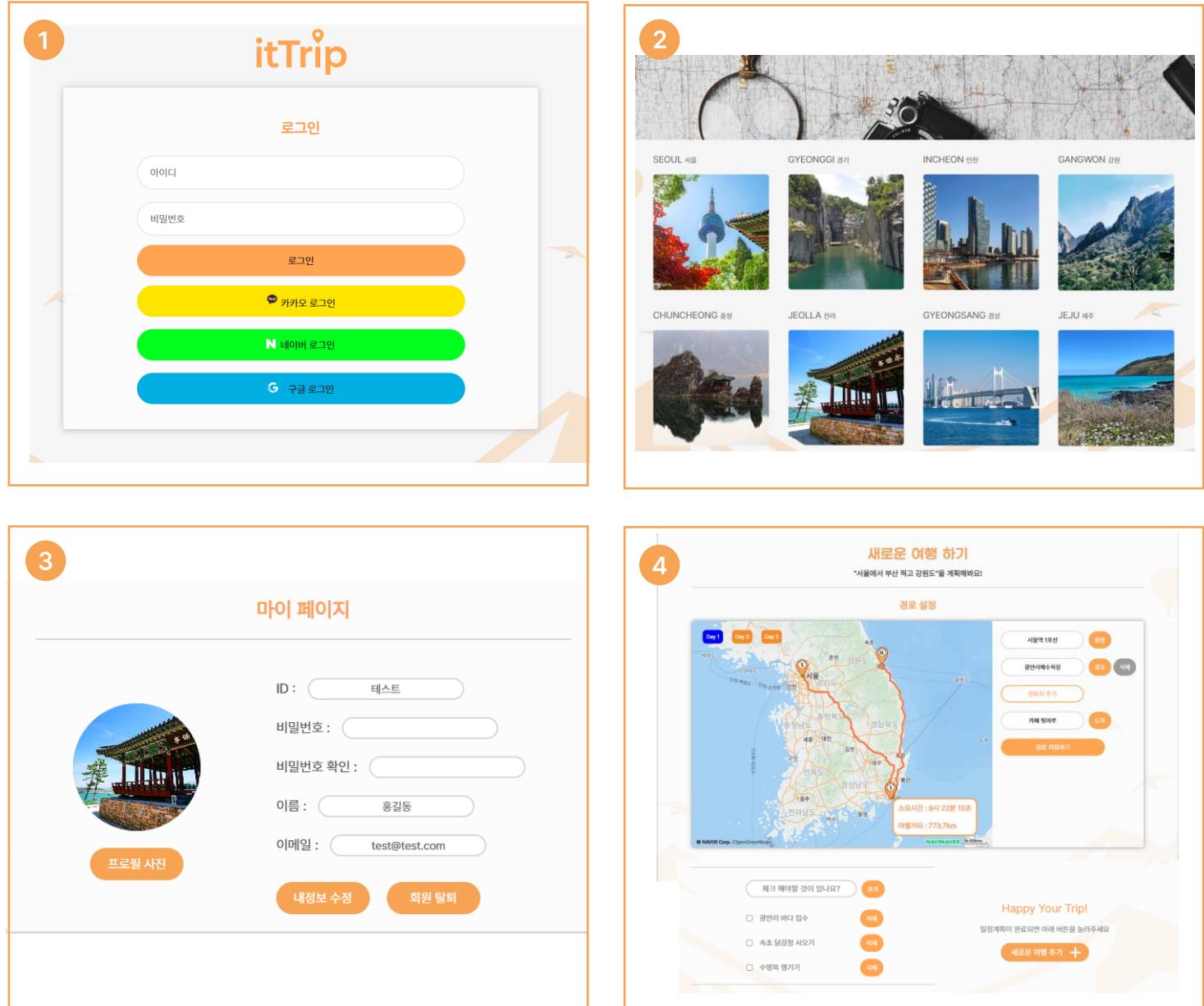
5

Column Name	Data Type	Nullable	Key	Max Length
signgu_cd	varchar	NO	PRI	5
area_cd	varchar	YES		2
area_nm	varchar	YES		10
signgu_nm	varchar	YES		10

8. 프로젝트 구성

1. Screen

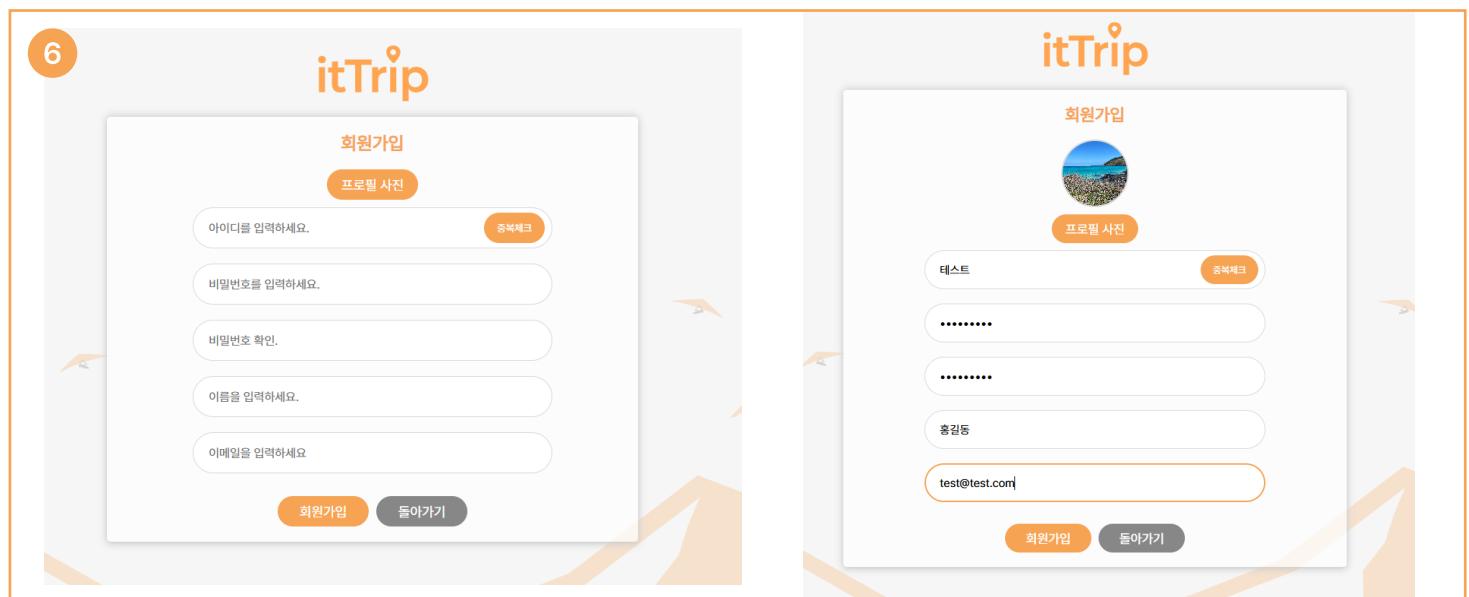
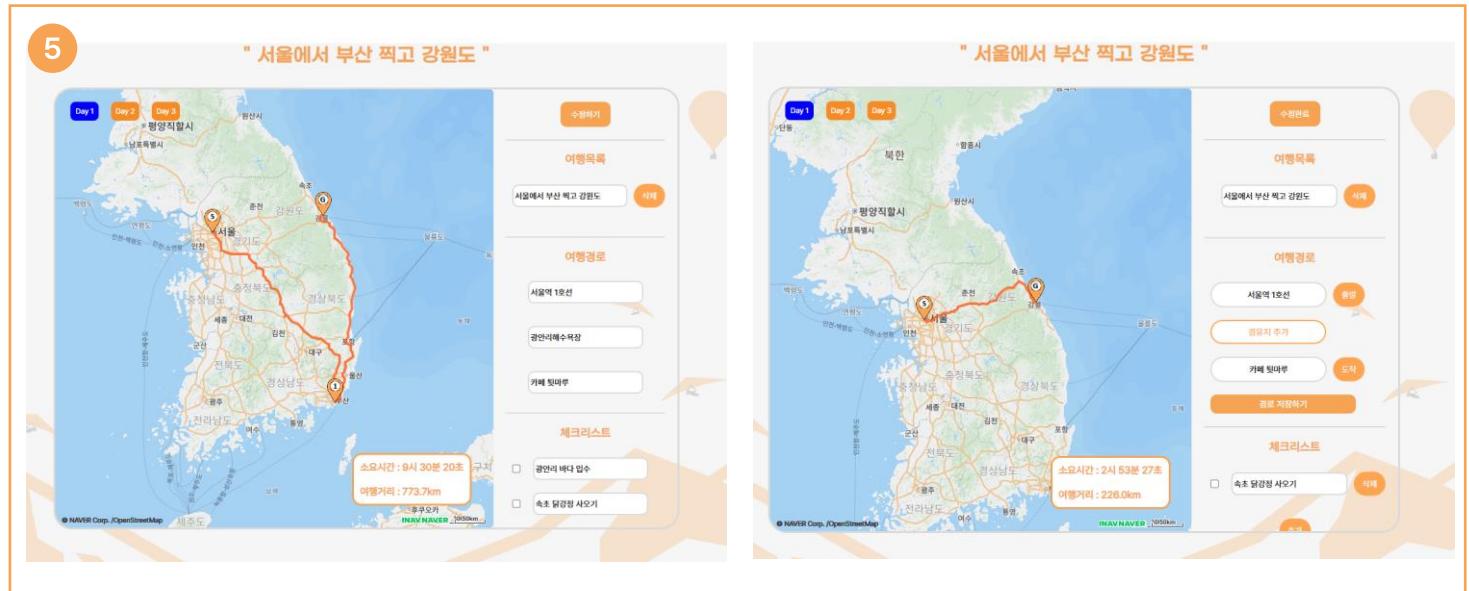
```
▽ src
  ▽ components
    JS AddData.js
    JS AddRoot.js
    JS CheckList.js
    JS DateCheck.js
    JS Header.js
    JS Logo.js
    JS MainLocal.js
    JS MainSlider.js
    JS Map.js
    JS Modal.js
    JS Scroll.js
    JS SocialLogin.js
  ▽ context
    JS ProjectContext.js
    JS useModal.js
  > css
  > img
  ▽ screen
    JS EntirePlan.js
    JS Login.js • 1
    JS Maintest.js • 2
    JS MyPage.js • 3
    JS NewTrip.js • 4
    JS SignUp.js
  # App.css
  JS App.js
```



8. 프로젝트 구성

1. Screen

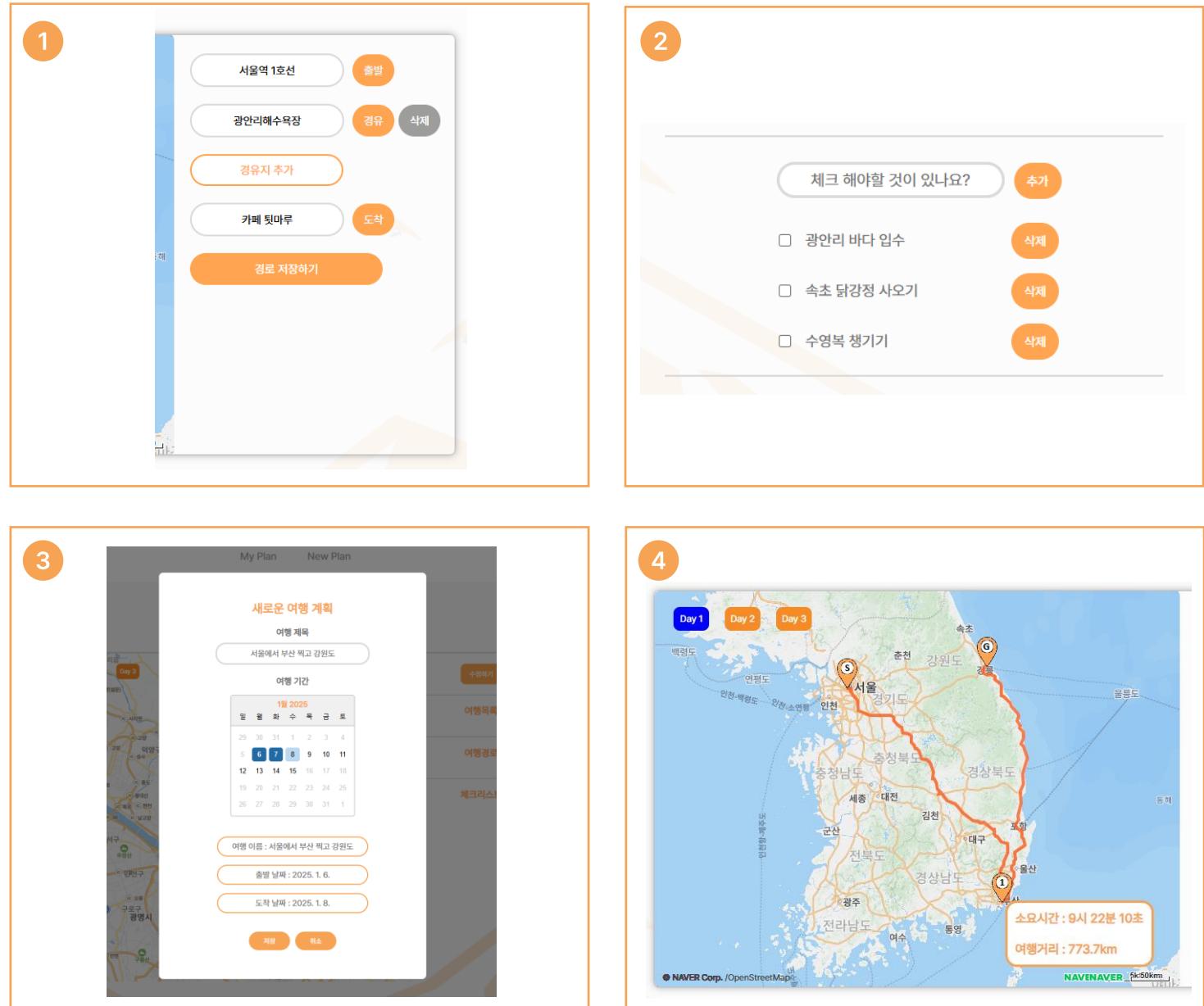
```
src
  components
    AddData.js
    AddRoot.js
    CheckList.js
    DateCheck.js
    Header.js
    Logo.js
    MainLocal.js
    MainSlider.js
    Map.js
    Modal.js
    Scroll.js
    SocialLogin.js
  context
    ProjectContext.js
    useModal.js
  > css
  > img
  screen
    EntirePlan.js • 5
    Login.js
    Maintest.js
    MyPage.js
    NewTrip.js
    SignUp.js • 6
  # App.css
  App.js
```



8. 프로젝트 구성

2. Components

```
▽ src
  ▽ components
    JS AddData.js • 1
    JS AddRoot.js
    JS CheckList.js • 2
    JS DateCheck.js • 3
    JS Header.js
    JS Logo.js
    JS MainLocal.js
    JS MainSlider.js
    JS Map.js • 4
    JS Modal.js
    JS Scroll.js
    JS SocialLogin.js
  ▽ context
    JS ProjectContext.js
    JS useModal.js
  > css
  > img
  ▽ screen
    JS EntirePlan.js
    JS Login.js
    JS Maintest.js
    JS MyPage.js
    JS NewTrip.js
    JS SignUp.js
  # App.css
  JS App.js
```



8. 프로젝트 구성

2. Components

```
▽ src
  ▽ components
    JS AddData.js
    JS AddRoot.js
    JS CheckList.js
    JS DateCheck.js
    JS Header.js • 5
    JS Logo.js
    JS MainLocal.js • 6
    JS MainSlider.js
    JS Map.js
    JS Modal.js
    JS Scroll.js
    JS SocialLogin.js • 7
  ▽ context
    JS ProjectContext.js
    JS useModal.js
  > css
  > img
  ▽ screen
    JS EntirePlan.js
    JS Login.js
    JS Maintest.js
    JS MyPage.js
    JS NewTrip.js
    JS SignUp.js
  # App.css
  JS App.js
```

