

# 보고서

## 보고서 및 논문 윤리 서약

1. 나는 보고서 및 논문의 내용을 조작하지 않겠습니다.
2. 나는 다른 사람의 보고서 및 논문의 내용을 내 것처럼 무단으로 복사하지 않겠습니다.
3. 나는 다른 사람의 보고서 및 논문의 내용을 참고하거나 인용할 시 참고 및 인용 형식을 갖추고 출처를 반드시 밝히겠습니다.
4. 나는 보고서 및 논문을 대신하여 작성하도록 청탁하지도 청탁받지도 않겠습니다.

나는 보고서 및 논문 작성 시 위법 행위를 하지 않고, 명지인으로서 또한 공학인으로서 나의 양심과 명예를 지킬 것을 약속합니다.

보고서명 : 진행보고서(상세설계보고서)

학 과 : 컴퓨터 공학과

과 목 : 팀 프로젝트 1


담당교수 : 한승철 교수

제 출 일 : 2021 년 11 월 15 일

팀 명 : 4조참치

팀 장 학번 : 60182151

이름 : 김윤기



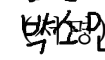
팀 원 학번 : 60182131

이름 : 김명비



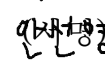
팀 원 학번 : 60182167

이름 : 박소민



팀 원 학번 : 60185132

이름 : 안선영





## 4조참치

설계/프로젝트명: 지하철 노선도 웹사이트  
발주자: 명지대학교 컴퓨터공학과 한승철 교수  
보고기관: 명지대학교 컴퓨터공학과  
설계팀: 4조참치  
작성일자: 2021.11.15  
문서 버전: V1.0

## 요약문

### 요약문

기존 지하철 노선도 서비스에 추가적인 기능을 구현하는 것을 목표로 하는 프로젝트입니다. 사용자들에게 좀 더 편리한 앱을 제공하고자 설문조사를 통해 도출한 사용자 요구사항을 만족하는 서비스를 구현하는 것을 목적으로 설정하였고, 기존의 기능뿐만 아니라 추가적인 기능을 설계하여 사용자가 편리함을 느끼는데 가장 큰 주안점을 두고 있습니다. 리액트, 익스프레스, MySQL을 통한 서비스 구현을 MVP로 하며, MVP 구현 완료 후 여유가 된다면 도커, Nginx, DNS, HTTPS를 사용해 배포까지 완료할 예정입니다.

### Abstract

The 'Subway Line Map' project is not only just a map that users find the way of route but it also provides additional features. For users to think that this is more convenient application than existing one, the goal of this project is to implement several additional services that satisfy the user requirements derived from the survey. This project is planing to be made with MVP using React, Express, MySQL and use Docker, Nginx DNS, HTTPS for further deployment in the future.

# < 목 차 >

1 서론 .....	1
1.1 목적 .....	1
1.2 주안점 .....	1
1.3 설계추진 및 지원주체와 사용자 및 기타 관련자 목록 .....	1
1.4 팀 구성 .....	1
1.4.1 팀 명 .....	1
1.4.2 팀 구성원과 배경 .....	2
1.5 프로젝트 .....	2
1.5.1 필요성 및 기대효과 .....	2
1.5.2 시장성 .....	2
1.5.3 목표 .....	2
2 현실적 제한조건 .....	3
3 설계 목적 .....	4
4 설계 결정에 대한 진행 사항 .....	4
4.1 프로젝트 개발환경 및 플랫폼 선정 .....	4
4.2 기능선정 및 구현 .....	5
4.2.1 사용자 요구사항 고려한 기능 선정 .....	5
4.2.2 시장성 및 경제성 등을 고려한 기능선정 .....	8
4.3 현재 구현된 기능 .....	9
4.4 문제해결에 적용된 이론 .....	12
4.5 자료구조 사용 .....	12
4.5.1 플로이드 와샬 .....	12
4.5.2 Back tracking .....	13
4.6 SW 이용한 설계 .....	14
4.6.1 UseCase Diagram .....	14
4.6.2 OmniGraffle을 이용한 Class Diagram .....	15
4.6.3 ASCII tree generator를 이용한 Class Diagram .....	16
4.7 화면 설계 .....	22
4.7.1 전체화면설계 .....	22
4.7.2 실제 구현화면 .....	23
5 설계 계획 .....	37
5.1 일정표 .....	37
5.2 업무 분담 .....	37
6 설계 평가 .....	38
7 참고자료 .....	40
7.1 국내도서/ 논문 .....	40
7.2 해외도서/ 논문 .....	40
7.3 기타참고자료 .....	40

## 〈표 목차〉

표 1 팀 구성도 및 경력	1
표 2 현실적 제한조건	3
표 3 개발환경 및 타겟플랫폼 가중순위비교 표	4
표 4 타 어플리케이션 푸(pugh)방법비교 표	8
표 5 기능/부품의 설계 및 구현 계획 표	9
표 6 현재 구현된 기능 표	10
표 7 플로이드와샬 사용 표	12
표 8 Back tracking 사용표	13
표 9 유스케이스 명세서	15
표 10 프로젝트 전체 파일 구조	16
표 11 프로젝트 백엔드 파일 구조	16
표 12 프로젝트 프론트엔드 파일 구조	19
표 13 개발 일정표	37
표 14 팀원 업무 분담 표	37
표 15 사용자평가표	38
표 16 자체평가표	38

## 〈그림 목차〉

그림 1 지하철 앱(또는 웹)을 사용해 본적이 있으십니까?	5
그림 2 앱(또는 웹)을 얼마나 자주 이용하십니까?	6
그림 3 지하철 앱(또는 웹)의 만족도는?	6
그림 4 지하철 앱에 추가적으로 있었으면 하는 기능은?	7
그림 5 UseCase diagram	14
그림 6 전체화면설계도	22
그림 7 페이지 헤더	23
그림 8 메인 화면	23
그림 9 메인화면-경유지 선택	24
그림 10 메인화면-노선도에서 역 선택	24
그림 11 메인화면-에러메시지 위치	25
그림 12 검색결과-경유지 설정X	25
그림 13 검색결과-경유지 설정O	26
그림 14 회원가입차	26
그림 15 회원가입창-에러 메시지 위치	27
그림 16 로그인창	28
그림 17 로그인창-에러메시지 위치	28
그림 18 인증 이메일 재발송 창	29
그림 19 인증 이메일 재발송 창-에러 메시지 위치	30
그림 20 최근 검색내역 모달창	30
그림 21 유실물센터 조회창	31
그림 22 물품보관함 조회창	32
그림 23 즐겨찾기 창	32
그림 24 마이페이지 창	33
그림 25 마이페이지 창-에러메시지 위치	34
그림 26 민원창	35
그림 27 민원창-에러메시지 위치	35
그림 28 DB ERD	36

# 1. 서론

## 1.1. 목적

이 프로젝트는 기존 지하철 노선도 서비스를 필요로 하는 사용자들이 해당 서비스에서 최소 비용 경로, 경유지 지정경로, 편의시설 위치 등 더 많은 정보를 정확하게 얻음으로써 보다 빠르고 정확한 지하철 이용을 할 수 있도록 하는 것을 목표로 합니다. 설문조사를 통해 도출한 사용자 요구사항인 최소비용 경로검색, 최근 검색내역, 경로 즐겨찾기, 경유지 설정, 분실물센터, 물품보관함 찾기 기능을 포함하는 지하철 노선도 웹사이트를 구현해 사용자가 원하는 옵션에 대한 경로를 더 빠르게 찾을 수 있고, 지하철 역에서 제공하는 편의시설을 헤매지 않고 더 편리하게 사용할 수 있게 하는 것이 목적입니다.

## 1.2. 주안점

2019년 서울교통공사의 통계에 따르면 서울에서만 하루에 747만명에 육박하는 사람들이 지하철을 이용하고 있습니다. 그에 따라 수많은 지하철 노선도 애플리케이션이 상용화되어 있습니다. 그럼에도 불구하고 즐겨찾기, 분실물센터와 물품보관함 위치, 경유지 설정, 최소비용으로 이동과 같은 유저가 필요로 하는 기능들이 기존 지하철 노선도 애플리케이션에는 구현되어 있지 않습니다. 따라서 이번 프로젝트를 통해 기존 지하철 노선도 애플리케이션에 구현되어 있었던 최단거리, 최단시간 노선 검색과 더불어 위에서 언급했던 기존 지하철 노선도 서비스에서는 구현되어 있지 않은 기능들을 구현하고자 합니다.

## 1.3. 설계추진 및 지원주체와 사용자 및 기타 관련자 목록

설계추진 : 김윤기, 김명비, 박소민, 안선영

지원주체 : 명지대학교 컴퓨터공학과

사용자 : 명지대학교 학생

설문조사 : 명지대학교 학생

## 1.4. 팀 구성

### 1.4.1. 팀 명

팀명은 '4조참치'입니다.

## 1.4.2. 팀 구성원

직책	성명	학번	이메일	연락처	경험 및 능력
팀장	김윤기	60182151	kyk990328@gmail.com	010-3687-1878	C++, Javascript, React, Express, AWS(RDS, EC2, S3), mysql, JQuery
팀원	김명비	60182142	jkd2584@gmail.com	010-5390-6071	C, C++, Java
팀원	박소민	60182167	chloesominpark@gmail.com	010-9916-1355	C, Java, Javascript
팀원	안선영	60185132	skditjsdud35@naver.com	010-2375-1595	C, Java

< 표 1 팀 구성도 및 경력 >

## 1.5. 프로젝트

### 1.5.1. 필요성 및 기대효과

기존 지하철 노선도 서비스는 사용자가 해당 서비스를 통해 경로를 찾고자 하는 목적을 충족하고 있지만, 설문조사를 통해 유저들이 경로를 검색하는 기능 뿐만 아니라 분실물센터와 물품보관함 찾기, 경유지를 포함한 경로 검색, 최근 검색내역, 경로 즐겨찾기 기능 역시 필요로 한다는 것을 새롭게 알게 되었습니다. 따라서 본 프로젝트에서는 이런 기능들을 구현하여 사용자가 더 빠르게 경로를 검색하고 지하철 역이 제공하는 편의시설을 찾아 헤매지 않고 더 명확히 파악하게 하고자 합니다.

### 1.5.2. 시장성

위에서 언급했듯이 2019년 기준 서울에서만 하루에 약 747만명의 사람들이 지하철을 이용하고 있습니다. 또한 기존 지하철 서비스는 사용자가 해당 서비스를 통해 목적지에 도착할 수 있는 경로를 찾고자 하는 목적을 충족하고 있습니다. 하지만 지하철 노선도에 대한 설문을 한 결과 유저들은 경로 즐겨찾기, 유실물센터와 물품보관함 위치, 경유지를 포함한 경로검색, 최소비용 경로를 단순한 경로 검색 기능과 더불어 필요로 한다는 것을 알게 되었습니다.

유저들은 서비스를 사용할 때 더 나은 대체재가 나오면 기존 서비스를 버리고 새로 출시한 서비스를 사용합니다. 배달 서비스인 배달통(2021년 1월 기준 점유율 0.72%)이 2010년에 첫 서비스를 시작했지만 그로부터 9년 뒤인 2019년에 출시한 쿠팡이츠(2021년 1월 기준 점유율 13.56%)보다도 한참 낮은 점유율을 점유하고 있는 것이 그 예시입니다. 따라서 위에서 언급했던 기능들을 추가적으로 구현한다면 기존 지하철 노선도 서비스를 사용하고 있던 유저들이 본 프로젝트를 통해 구현된 서비스로 유입될 것이라 판단됩니다.

### 1.5.3. 목표

본 프로젝트는 기존의 지하철 노선도 서비스에서 구현된 기능인 최단거리, 최소시간 경로검색 기능과 더불어 최소비용 검색, 최근 검색 즐겨찾기, 분실물센터와 물품보관함 위치, 경유지 설정 등 유저가 필요로 하는 기능을 추가로 구현함으로써 유저들이 더 편리하게 목적지에 도달할 수 있는 경로를 찾거나 지하철 역이 제공하는 편의시설을 이용하게 하는 것을 목표로 합니다.

## 2. 현실적 제한조건

현실적 제한조건	상세 제한조건	내 용
관리적 제한	시간	• 제품 개발기간을 12주로 합니다.
	예산	기존에 사용 중이던 컴퓨터를 사용함에 따라 추가적인 예산은 들지 않을 것으로 판단됩니다.
	인적자원	• 제품 개발에 투입되는 MM(Man-Month)은 12M/M입니다(팀원 4명, 개발 기간 3달) • 각 업무를 분담하여 개발합니다.
	물적자원	• 제품 제작에 사용될 컴퓨터의 사양 -맥북 프로 16인치 2019 i7, RAM 16GB 512GB SSD -윈도우 컴퓨터 i5, RAM 8GB, 256GB SSD -윈도우 컴퓨터 i5, RAM 8GB, 256GB SSD -윈도우 컴퓨터 i5, RAM 8GB, 128GB SSD -서버: Ubuntu 20.04LTS 맥미니 2011 mid, i5, RAM 8GB, 128GB SSD
시스템적 요구	성능	• 서비스 요청에 대한 응답시간을 1초 이하로 합니다. • 코딩 표준 규약을 사용해 유지보수성을 높입니다.
	기능성	• 사용자가 쉽게 사용할 수 있는 구성. • 복잡하지 않고 핵심적이고 단순한 기능들로 구성.
	윤리성	• 사용자의 권익을 침해하는 비도덕적, 비윤리적 행위는 하지 않으며 상업적 저작권을 위배하는 행위는 절대 하지 않는다.
	유지보수성	• 수정 예상 LOC(Line Of Codes)가 4000 이하이도록 합니다

〈 표 2 현실적 제한조건 〉



### 3. 설계 목적

본 프로젝트는 기존 지하철 노선도에 포함되었는 최소시간, 최단거리 경로 찾기 기능과 유저들이 추가로 필요로하는 기능인 물품보관센터, 즐겨찾기, 경유지 설정, 분실물 센터 등을 추가로 구현해 사용자가 본 서비스를 통해 더 많은 정보를 얻을 수 있게 하는 것을 목적으로 합니다. 또 한 직관적인 UX디자인으로 유저들의 진입 장벽을 낮추고 지속적인 서버 관리를 통해 사용자에게 안정적인 서비스를 제공한다.

### 4. 설계 결정에 대한 진행 사항

#### 4.1. 프로젝트 개발환경 및 플랫폼 선정

이렇게 정한 아이디어를 구현할 개발환경과 플랫폼을 선정하기로 하였습니다. 3가지의 플랫폼들을 아이디어 가중 순위 비교를 통해 비교하기로 하였습니다. 3가지 개발환경은 JAVA(컴퓨터), 안드로이드(안드로이드 핸드폰), 스위프트(iPhone) 이렇게 3가지로 구별해 보았고 기준은 3가지로 정하였고 이는 다음과 같습니다.

- 구현 가능성: 개발이 얼마나 어려운지
- 시장성: 현재 해당 프로그램 주제 관련 시장의 규모
- 실용성: 실제로 많은 사람들이 사용할지

각각의 기준마다 점수를 다르게 두었습니다. 첫 번째 구현가능성에 40점이라는 높은 점수를 주었는데 그 이유는 6주라는 짧은 기간이 제한조건으로 붙었기 때문이다. 나머지 시장성과 실용성의 경우 같은 30점을 부여하였는데 이 기준의 경우 서로 비슷하게 중요하다고 생각되어 이렇게 점수를 부여하고 비교를 진행했습니다.

개발환경 및 타겟플랫폼	판정기준	구현 가능성	시장성	실용성	총점 (100)	최종 순위
		40	30	30		
Java(Android)		10	6	10	88	2
Swift(IOS)		6	9	10	84	3
REACT(WEB)		8	10	10	92	1

< 표 3 개발환경 및 타겟플랫폼 가중순위비교 표 >

그 결과 리액트를 이용하여 지하철 웹사이트를 만드는 것이 선정되었습니다.

Java의 경우 한국 시장에서 대부분의 스마트폰 유저가 안드로이드를 사용하는 만큼 유저 확보에는 용이하다 판단했습니다. 하지만 아이폰 사용자의 인앱 결제 비율이 안드로이드 유저 보다 높기 때문에 시장성이 없다 생각하여 많은 점수를 잃어 최종순위 2위를 기록했습니다.

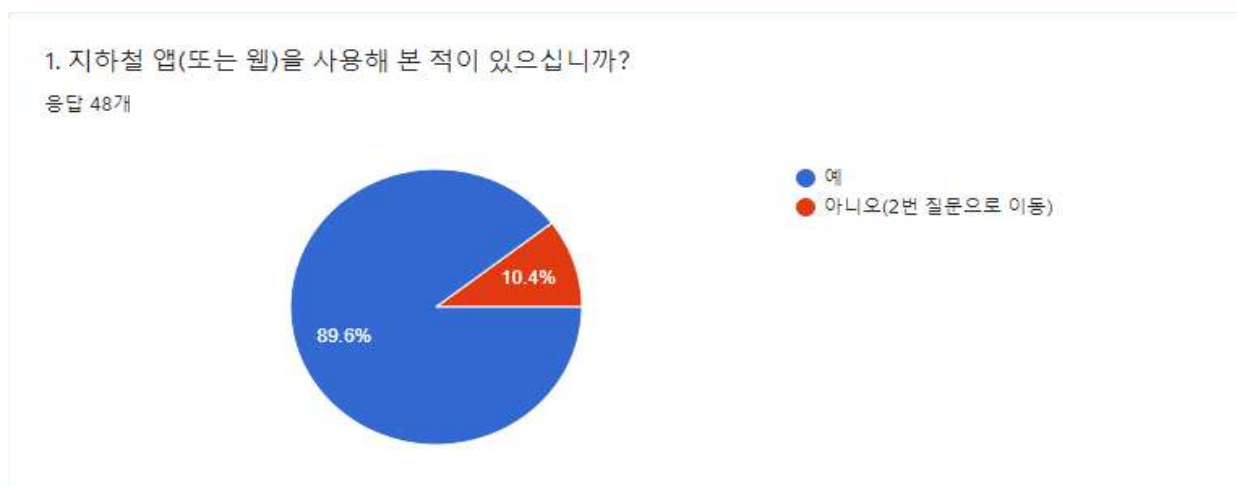
스위프트의 경우 시장성과 실용성은 높으나 팀원 중 맥을 사용하는 팀원이 오직 1명이라 개발에 용이하지 않다고 판단했습니다. 그 결과 최종순위 3위를 기록했습니다.

최종적으로 시장성과 실용성에서 높은 점수를 차지하고 OS구분 없이 유저들이 접근할 수 있다고 생각한 리액트가 선택되었습니다.

## 4.2. 기능 선정 및 구현

### 4.2.1. 사용자 요구사항 고려한 기능 선정

기능 선정에 앞서 사용자 요구사항을 더욱 확실하게 알아보기 위하여 이전에 조사했던 설문 조사를 다시 한 번 살펴보기로 하였습니다.



< 그림1 지하철 앱(또는 웹)을 사용해 본 적이 있으십니까? >

<그림 1>은 48명을 대상으로 지하철 노선도 서비스 설문 조사한 그림으로 지하철 앱을 사용해 본 사람이 89.6%로 압도적으로 많습니다.

### 1-1. 지하철 앱(또는 웹)을 얼마나 자주 이용하십니까?

응답 45개



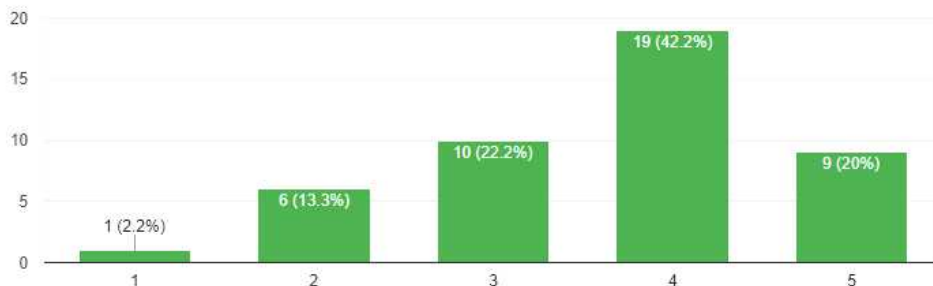
<그

### 림2 지하철 앱(또는 웹)을 얼마나 자주 이용하십니까? >

지하철 노선도 서비스를 얼마나 자주 이용하냐는 질문에는 26.7%의 사람이 주 1~2회, 33.3%의 사람이 주 3~4회, 28.9%의 사람이 5~6회, 그 밖의 사람들이 불규칙적으로 사용한다는 답변을 했습니다. 여기서 62.2%의 설문 참여인원이 주 3회 이상 지하철 노선도 서비스를 사용하는 것을 알 수 있습니다.

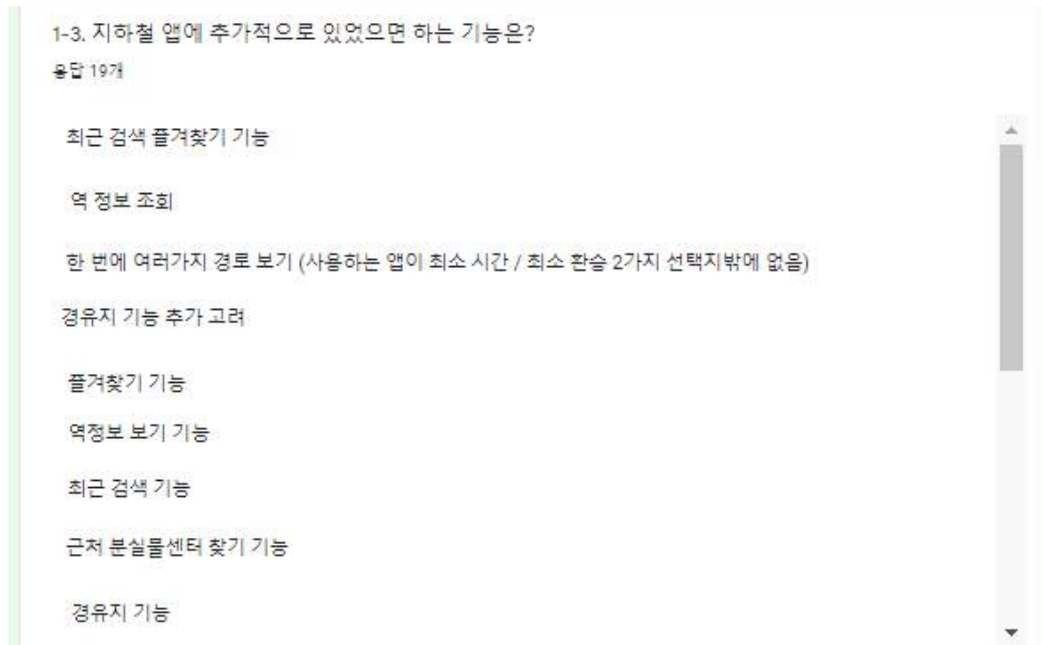
### 1-2. 현재 사용하는 지하철 앱(또는 웹)의 만족도는?

응답 45개



### < 그림3 지하철 앱(또는 웹)의 만족도는? >

<그림 3>을 통해 사람들의 기존 지하철 앱에 대한 평균 만족도가 다소 높은 것을 알 수 있습니다. 하지만 5점의 척도를 매우 불만족, 불만족, 보통, 만족, 매우 만족의 척도로 판단했을 때 만족 미만의 비율이 37.7%로 만족도가 낮은 경우도 무시 못 할 정도의 수치를 보여주었습니다.



#### <그림 4 지하철 앱에 추가적으로 있었으면 하는 기능은? >

<그림 4>에서 유저들이 기존 지하철 노선도 서비스에 있는 기능인 최소시간 검색과 최단 경로 검색 뿐만 아니라 검색기록, 즐겨찾기, 분실물센터, 경유지를 포함하는 경로 검색 등을 원한다는 것을 알게 되었습니다. 따라서 본 서비스는 기존 지하철 서비스의 기능을 구현함과 동시에 유저가 추가로 원하는 기능 역시 구현할 계획입니다.

#### 4.2.2. 시장성 및 경제성 등을 고려한 기능선정

소비자들의 요구사항을 만족시키는 기능을 알아본 우리들은 위의 기능과 더불어 다른 카페 어플리케이션의 기능을 비교해보고 기능을 추가시켜서 더 차별성 있는 어플리케이션을 만들기로 하였습니다.

타 회사 기능	서울교통공사	인천교통공사	카카오지하철	사이버스테이션
최단거리 경로검색	+	-	-	+
최단시간 경로검색	+	-	+	+
최소비용 경로검색	-	-	-	-
즐거찾기	-	-	+	-
검색기록	-	-	+	-
물품보관함	-	-	-	+
유실물센터	-	-	+	-
민원	-	-	-	-
총점	-4	-8	0	-2

< 표 4 타 어플리케이션 푸(pugh)방법비교 표 >

시중에서 서비스 되고 있는 타 서비스들을 비교한 결과 카카오지하철을 제외한 나머지 서비스는 모두 마이너스를 기록했습니다. 특히 인천교통공사가 제공하는 지하철 노선도 서비스의 경우 말 그대로 지하철 노선도 만을 보여주는 서비스였습니다. 서울교통공사는 공사가 제공하는 지하철노선도 서비스 중에서는 준수한 기능들을 보여주었지만 이 역시 즐겨찾기, 검색 기록, 물품보관함 등의 기능을 제공하고 있지는 않았습니다.

위 표에 있는 4개의 타 서비스 중 이용자 수가 가장 많은 카카오지하철의 경우 나머지 3개의 서비스들에 비하면 구현된 기능이 여러 존재하지만 <표 4>에 명시되어 있는 본 서비스를 통해 구현하고자 하는 모든 기능을 포함하고 있지는 않았습니다.

사이버스테이션은 공사가 제공하는 지하철노선도 서비스에 비하면 다양한 기능을 포함하지만 유저 개인이 생성한 데이터를 즐겨찾기, 검색기록을 통해 다시 볼 수 있는 기능이 없습니다.

결론적으로 <표 4>에 명시된 모든 기능을 본 서비스에서 구현한다면 타 지하철노선도 서비스에 비해 충분한 경쟁력을 갖출 수 있을 것이라 판단됩니다.

### 4.3. 현재 구현된 기능

번호	기능/부품	계획 사항 (설계/구현/향후)	설계 난이도 (상/중/하)	구현 난이도 (상/중/하)
1	메인 페이지를 길 찾기 페이지로 한다.	구현	하	하
2	모든 페이지에 공통된 페이지 헤더가 들어간다.	구현	하	하
3	시작점, 도착점을 명시하고 세 옵션 중 하나를 선택해 결과를 모달창으로 조회한다.	구현	중	중
4	길 찾기 결과는 사용자가 선택한 세 옵션중 하나와 그에 대응되는 시작점, 도착점, 노선 요약을 보여준다	구현	중	중
5	길 찾기를 위해 입력한 데이터를리셋하는 버튼이 있다.	구현	하	하
6	이메일과 비밀번호를 통해 회원가입을한다. .	구현	중	중
7	로그인을 한다.	구현	중	중
8	회원가입 완료를 위해 확인 메일을전송한다.	구현	중	중
9	확인 메일에서 완료 버튼을 눌러야지만회원가입을 완료시킨다.	구현	중	중
10	경유지 사용 버튼 클릭, 경유지, 시작점, 도착점을 명시하고, 세 옵션 중 하나를 선택해 결과를 모달창으로 조회한다.	구현	중	중
11	로그인을 한 후 최근 검색 결과를 조회한다.	구현	중	중
12	로그인을 한 후 경로 즐겨찾기를 조회한다.	구현	중	중
13	기존 즐겨찾기 목록에서 노란색 별모양을 누르면 즐겨찾기에서 삭제된다.	구현	중	중
14	마이페이지에서 회원가입시 입력했던 메일과 비밀번호를 수정한다.	구현	중	중

번호	기능/부품	계획 사항 (설계/구현/향후)	설계 난이도 (상/중/하)	구현 난이도 (상/중/하)
15	회원가입 페이지에서 '회원이신가요? 로그인 하세요' 라는 문구를 클릭했을 때 로그인 페이지로 이동한다.	구현	하	하
16	로그인 페이지에서 '회원이 아니신가요? 가입 하세요' 라는 문구를 클릭하면 회원가입 페이지로 이동한다.	구현	하	하
17	로그인을 한 후 로그아웃을 한다	구현	중	중
18	유실물센터 목록을 조회한다.	구현	하	하
19	물품보관함 목록을 조회한다.	구현	하	하
20	지하철 관련 민원을 넣는다	구현	중	중
21	검색기록모달창에서 즐겨찾기에 추가된 항목이면 노란별을, 아니면 회색별을 띄운다	구현	중	중
22	검색기록 모달창에서 검색기록 항목 하나를 클릭하면 검색결과 모달창으로 이동한다.	구현	중	중

〈 표 5 기능/부품의 설계 및 구현 계획 표 〉

위의 설계 구현 계획표를 작성하고 이를 통해 본 서비스에 다양한 기능들을 추가할 수 있었습니다. 현재 구현된 기능들은 다음과 같습니다.

## 현재 구현된 기능

기능	구현여부	기능	구현여부
메인 페이지를 길 찾기 페이지로 한다.	○	로그인을 한다.	○
모든 페이지에 공통된 페이지 헤더가 들어간다.	○	회원가입 완료를 위해 확인 메일을 전송한다.	○
시작점, 도착점을 명시하고 세 옵션 중 하나를 선택해 결과를 모달창으로 조회한다	○	확인 메일에서 완료 버튼을 눌러야지만 회원가입을 완료시킨다.	○
길 찾기 결과는 사용자가 선택한 세 옵션중 하나와 그에 대응되는 시작점, 도착점, 노선요약을 보여준다.	○	경유지 사용 버튼 클릭, 경유지, 시작점, 도착점을 명시하고, 세 옵션 중 하나를 선택해 결과를 모달창으로 조회한다	○
회원가입 페이지에서 '회원이신가요? 로그인하세요' 라는 문구를 클릭했을 때 로그인 페이지로 이동한다.	○	로그인 페이지에서 '회원이 아니신가요? 가입하세요' 라는 문구를 클릭하면 회원가입 페이지로 이동한다.	○
로그인을 한 후 로그아웃을 한다.	○	검색기록모달창에서 즐겨찾기에 추가된 항목이면 노란별을, 아니면 회색별을 띄운다	○
이메일과 비밀번호를 통해 회원가입을한다.	○	로그인을 한 후 최근 검색 결과를조회한다.	○
검색기록모달창에서 검색기록 항목 하나를 클릭하면 검색결과모달창으로 이동한다.	○		

< 표 6 현재 구현된 기능 표 >

현재 구현된 기능은 위와 같습니다. 기존 지하철 노선도 서비스에서 구현되어 있는 최단거리, 최단시간을 기준으로 한 경로 검색을 구현하였습니다. 또한 설문조사를 통해 파악한 유저의 요구사항인 검색기록, 즐겨찾기 등을 구현하기 위해 회원가입과 로그인을 구현하였습니다. 그 후 검색기록 모달창과 그에 해당하는 기능을 모두 완성했습니다.



## 4.4. 문제해결에 적용된 이론

위의 과정을 통해 앞으로 구현할 웹사이트와 그 기능들을 설계하였습니다. 경로를 찾는 기능에서는 플로이드 와샬 알고리즘을 적용했습니다. 플로이드 와샬 알고리즘을 적용한 이유는 사용자가 많아질수록 경로를 찾는 대상이 중복되게 되고 매번 같은 연산을 해야 한다는 문제를 해결하기 위해서입니다. 따라서 110개의 역에 대해 permutation을 구현해 모든 정거장에서 갈 수 있는 모든 정거장에 대한 경우의 수를 구하고, 하나의 정거장에서 다른 하나의 정거장으로 가는데 드는 각 최소 비용을 플로이드 와샬을 통해 구한 뒤, permutation으로 구한 모든 정거장 케이스를 통해 경로를 역추적하는 방식으로 모든 경로와 모든 비용에 대한 최적의 해를 구했습니다.

## 4.5. 자료구조 사용

### 4.5.1. 플로이드 와샬

플로이드와샬을 사용해 구현한 내용은 다음과 같습니다.

구현 기능	플로이드와샬 사용 방법
경로의 최단거리	DB에 넣은 경로의 최단거리를 구할 때 사용
경로의 최소비용	DB에 넣을 경로의 최소비용을 구할 때 사용
경로의 최단시간	DB에 넣을 경로의 최단시간을 구할 때 사용

< 표 7 플로이드와샬 사용 표 >

위에서 설명한 바와 같이 플로이드와샬을 사용해 최소비용을 미리 구해놓으면 매 요청마다 최소비용을 구하는 시간을 아낄 수 있습니다.

## 4.5.2 Back tracking

역추적을 사용해 구현한 내용은 다음과 같습니다.

구현 기능	Back tracking 사용 방법
경로의 최단거리 경로	DB에 넣을 각 최단거리에 해당하는 경로를 구할 때 사용
경로의 최소비용 경로	DB에 넣을 각 최소가격에 해당하는 경로를 구할 때 사용
경로의 최단시간 경로	DB에 넣을 각 최당시간에 해당하는 경로를 구할 때 사용

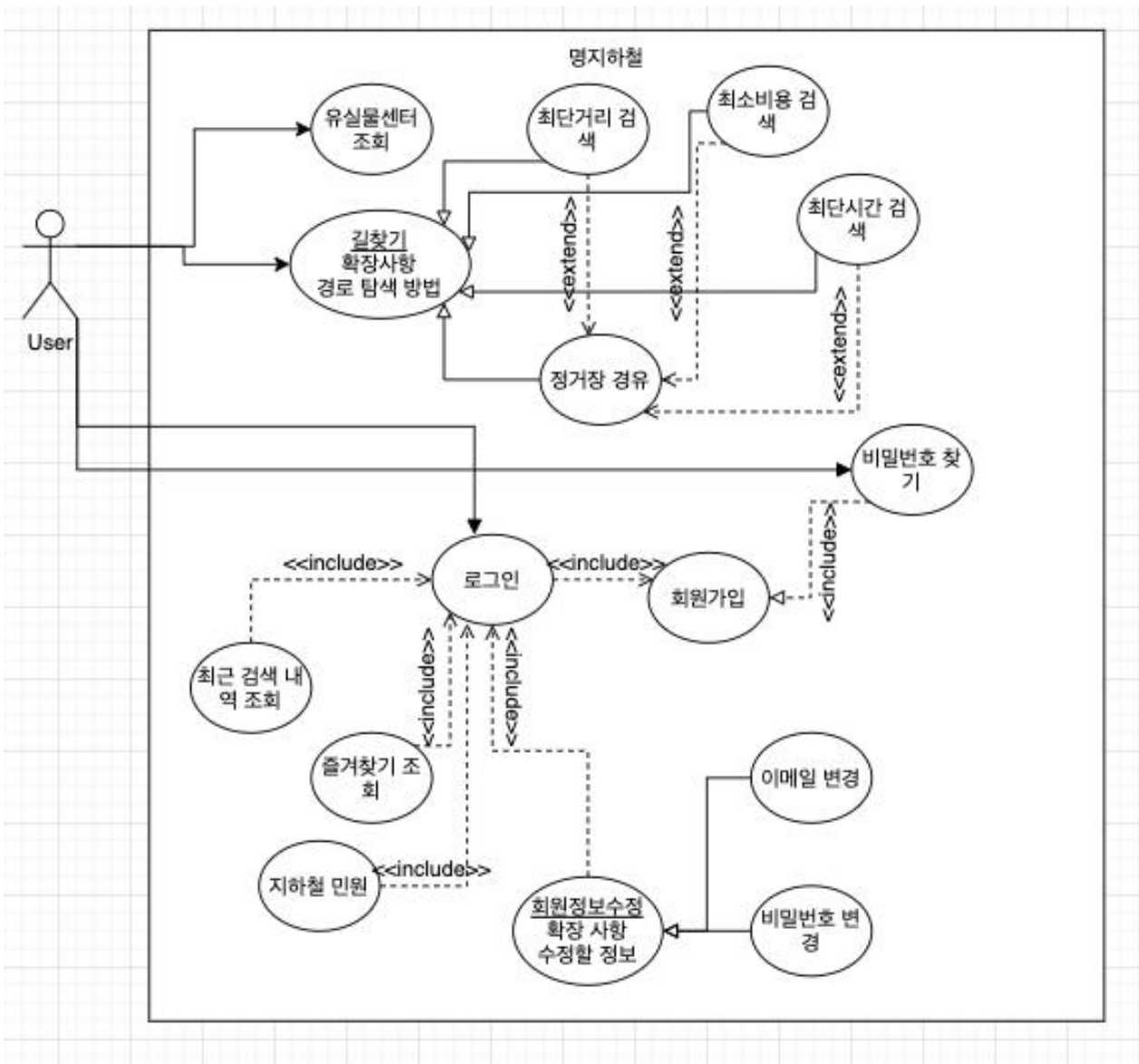
< 표 8 Back tracking 사용 표 >

플로이드와샬만을 사용하면 경로에 해당하는 최소비용만 구할 수 있기 때문에 플로이드와샬 알고리즘을 사용하면서 그에 해당하는 경로 역시 저장을 한 뒤 모든 경우의 수에 대해 역추적을 하여 실제 경로를 구했습니다.

## 4.6. SW 이용한 설계

### 4.6.1. UseCase Diagram

본 서비스에서 제공하는 기능과 유저 액터의 상호작용을 명시적으로 나타내기 위해 유스 케이스 다이어그램을 사용했습니다. 각 유스케이스에 대한 설명은 4.7.2 유스케이스 명세서에서 볼 수 있습니다.



< 그림 5 UseCase Diagram >

## 4.6.2. 유스케이스 명세서

유스케이스 번호	유스케이스명	자유명세
UC001	유실물 센터 조회	유실물 센터 위치 리스트를 보여준다
UC002	최단거리 검색	유저가 최단거리 검색을 선택하고 출발점, 도착점을 명세한 후 최단거리 path를 조회한다
UC003	최소금액 검색	유저가 최소금액 검색을 선택하고 출발점, 도착점을 명세한 후 최소금액 path를 조회한다
UC004	최단시간 검색	유저가 최소금액 검색을 선택하고 출발점, 도착점을 명세한 후 최소금액 path를 조회한다
UC005	정거장 경유	유저가 최소거리, 최소금액, 최단시간 중 하나를 선택하고 시작점, 도착점, 경유지를 명세한 후 선택한 최소 비용에 해당하는 경로 조회
UC006	회원가입	유저가 이메일, 비밀번호, 비밀번호 확인을 입력해 회원가입을 한다. 입력한 이메일로 인증을 한 후 회원가입을 완료한다
UC007	로그인	유저가 닉네임, 비밀번호를 통해 회원가입을 한다.
UC008	최근 검색 내역 조회	유저가 최근 검색한 검색 내역 7개를 조회한다. 검색 내역을 클릭하면 해당 검색 결과 모달창이 띄워진다. 검색 내역이 즐겨찾기에 추가되어 있으면 노랑별, 아니면 회색별이 보여진다.
UC009	즐거찾기 조회	유저가 즐겨찾기 추가한 경로를 본다.
UC010	이메일 변경	유저가 가입된 이메일을 변경한다
UC011	비밀번호 변경	유저가 사용 중인 비밀번호를 변경한다
UC012	비밀번호 찾기	유저 이메일로 비밀번호 변경 메일을 발송한다.
UC013	지하철 민원	사용자가 사용자의 이메일, 노선 번호, 민원내용을 입력하면 민원내용이 접수되고 접수한 민원의 내용은 이메일로 발송된다.

< 표 9 유스케이스 명세서 >

### 4.6.3. ASCII tree generator를 이용한 Class Diagram

ASCII tree generator라는 웹사이트를 이용해서 파일들 간의 관계를 정의하여 구체적으로 프로그래밍을 할 때 필요한 정보들을 정리해 클래스 다이어그램을 만들었습니다.

아래의 파일 구조는 대략적인 프로젝트 구조입니다. backend, frontend 디렉토리는 파일 구조가 복잡하기 때문에 별도로 서술하였습니다.

```
teamproject1/
├── data/
│   ├── certbot/
│   ├── nginx/
│   │   └── app.conf
│   └── thumbnail.png
├── frontend/
├── backend/
├── docker-compose.yml
├── init-letsencrypt.sh
├── init.sql
└── README.md
```

< 표 10 프로젝트 전체 파일 구조 >

Docker를 사용해 배포를 할 예정이기 때문에 docker-compose.yml이 프로젝트 루트폴더에 존재하게 됩니다. data 디렉토리는 nginx설정파일과 HTTPS인증서 발급과 관련된 certbot파일이 들어가 있습니다. init-letsencrypt.sh는 nginx를 구동하기 위해 인증서를 발급받는 셸스크립트입니다. https를 사용할 경우 인증서가 없으면 nginx가 구동되지 않습니다. 따라서 더미 인증서를 만들고 nginx를 구동한 다음 더미 인증서를 삭제하고 실제 인증서를 발급받아야 합니다.

```
backend/
├── .husky/
│   ├── _/
│   │   └── .gitignore
│   │   └── husky.sh
│   └── pre-commit
├── dist/
├── node_modules/
├── src/
│   ├── config/
│   │   └── passport/
│   │       └── index.ts
│   │       └── localStrategy.ts
│   └── controllers/
│       └── auth.controllers.ts
```

```

| | | └─ bookmark.controllers.ts
| | | └─ path.controllers.ts
| | | └─ searchHistory.controllers.ts
| | | └─ user.controllers.ts
| | └─ middleWares/
| | | └─ validateEmailReauthorize.middleware.ts
| | | └─ validateStation.middleWare.ts
| | | └─ validateUserInfo.middleware.ts
| | | └─ verifyAccessToken.middleware.ts
| | | └─ verifyRefreshToken.middleware.ts
| | | └─ validateEmailResetPassword.middleware.ts
| | └─ models/
| | | └─ authEmail.ts
| | | └─ currentSearched.ts
| | | └─ minCost.ts
| | | └─ minCostOtherValues.ts
| | | └─ minCostValue.ts
| | | └─ minPath.ts
| | | └─ minPathOtherValues.ts
| | | └─ minPathValue.ts
| | | └─ minTime.ts
| | | └─ minTimeOtherValues.ts
| | | └─ minTimeValue.ts
| | | └─ stationBookMark.ts
| | | └─ stationFromTo.ts
| | | └─ token.ts
| | | └─ user.ts
| | └─ routes/
| | | └─ auth.routes.ts
| | | └─ bookmark.routes.ts
| | | └─ path.routes.ts
| | | └─ searchHistory.routes.ts
| | | └─ user.routes.ts
| | └─ templates/
| | | └─ signUpEmailAuth.html
| | | └─ userComplaint.html
| | └─ utils/
| | | └─ jsonResponse/
| | | | └─ fail.ts
| | | | └─ success.ts
| | | └─ type/

```

```
| | | └─ auth.ts
| | | └─ error.ts
| | | └─ responseType.ts
| | | └─ searchPath.ts
| | └─ validation/
| |   └─ auth.ts
| |   └─ station.ts
| |   └─ emailAuth.ts
| |   └─ math.ts
| |   └─ optimizedPath.ts
| |   └─ searchHistory.ts
| |   └─ token.ts
| └─ app.ts
└─ .env
└─ .eslintrc
└─ .gitignore
└─ ormconfig.ts
└─ package.json
└─ README.md
└─ tsconfig.js
└─ yarn.lock
└─ yarn-error.log
└─ Dockerfile
```

〈 표 11 프로젝트 백엔드 파일 구조 〉

.husky는 husky 구동을 위한 설정이 들어있는 디렉토리입니다. prettier와 eslint를 통해 컨벤션을 정하고 프로젝트를 하고 있기는 하지만 commit을 할 때 검사를 할 수 없어서 올바른지 않은 스타일의 코드가 올라갈 수 있습니다. 따라서 husky를 사용해 pre-commit 단계에서 eslint를 활용해 코딩 스타일은 한번 검증하고, 수정한 다음 commit을 합니다.

dist 디렉토리는 ts에서 js로 변환된 파일이 들어있는 디렉토리 입니다. node는 js의 실행환경이기 때문에 TS로 프로젝트를 하고 실제 구동시에는 변환된 js파일들로 구동을 해야 합니다.

src는 실제 프로젝트의 코드가 들어가 있는 디렉토리입니다. config는 third-party module들의 설정파일이 들어가는 디렉토리입니다. controllers는 말 그대로 controller가 들어가 있는 routes에는 router가 models에는 DB model이 들어가 있습니다. middlesWares에는 특정 요청에 대해 라우터가 로직을 수행하기 전 거치는 middleware가 포함되 있습니다. teamplates에는 html파일이 들어가 있습니다. 본 프로젝트는 Client-side rendering을 사용하기 때문에 원칙적으로는 html파일이 서버에 존재하지 않지만 메일의 내용을 html로 구성하기 위해 만들었습니다. utils는 기타 함수가 포함된 디렉토리입니다. controller에서 사용하는 함수들과 TS에서 타입 명시를 위해 별도로 정의한 interface가 이 위치에 존재합니다.

app.ts는 엔트리 포인트입니다.

.env는 소스코드 상에서 노출이 되면 안되는 정보를 담고 있습니다. DB관련 정보, JWT secret, 사용자에게 이메일을 보낼때 사용하는 이메일 정보 등이 .env에 존재합니다. .env내 부 정보는 노출되면 안되기 때문에 레파지토리에 올라가지 않습니다.

.eslintrc는 eslint설정파일입니다.

.gitignore는 git에 commit할 때 무시할 디렉토리, 파일 목록입니다.

ormconfig.ts는 typeorm 설정파일입니다.

Dockerfile은 backend container를 만들기 위한 image를 만들때 사용하는 파일입니다.

```

frontend/
├── .husky/
│   ├── /
│   ├── .gitignore
│   ├── husky.sh
│   └── pre-commit
├── config/
├── node_modules/
├── build/
├── public/
│   ├── img/
│   │   ├── bookmark.svg
│   │   ├── logo.svg
│   │   └── unbookmark.svg
│   ├── favicon.ico
│   ├── index.html
│   ├── manifest.json
│   └── robots.txt
├── src/
│   ├── components/
│   │   ├── header/
│   │   │   └── HeaderNav.js
│   │   ├── map/
│   │   │   └── SubwayMap.js
│   │   ├── modal/
│   │   │   ├── searchHistoryModal/
│   │   │   │   └── SearchHistoryModal.js
│   │   │   ├── searchResultModal/
│   │   │   │   ├── SearchResultData.js
│   │   │   │   └── SearchResultModal.js
│   │   │   ├── CommonModal.js
│   │   │   ├── ModalCloseButton.js
│   │   │   └── Portal.js
│   │   └── search/
│   │       ├── InputOperaionTarget.js
│   │       ├── ResetButton.js
│   │       ├── SearchButton.js
│   │       ├── SearchCotainer.js
│   │       ├── SelectOperationTarget.js
│   │       ├── SelectStopoverButton.js
│   │       └── SubmitButton.js

```



```

├── styles/
│   ├── Authorization.js
│   ├── GlobalStyle.js
│   ├── ResultMessage.js
│   ├── SearchPathBtn.js
│   └── SelectTargetButton.js
├── user/
│   ├── Logout.js
│   ├── PathTable.js
│   └── ValidateUserAccount.js
├── lib/
│   ├── authenticateData.js
│   ├── customAxios.js
│   ├── dataPath.js
│   ├── localStorage.js
│   ├── makeRequest.js
│   ├── subwayData.js
│   ├── validateStation.js
│   └── validateUserInfo.js
├── pages/
│   ├── Bookmark.js
│   ├── EmailVerification.js
│   ├── EmailReauthorization.js
│   ├── FindPathPage.js
│   ├── SignIn.js
│   ├── SignUp.js
│   ├── UserProfile.js
│   ├── LostAndFoundLocation.js
│   ├── UserComplaint.js
│   └── StorageLocation.js
├── stores/
│   ├── indexStore.js
│   ├── login.js
│   ├── modalOpenStore.js
│   └── searchTargetStore.js
├── App.js
├── index.css
├── index.js
├── reportWebVitals.js
├── .env.development
├── .env.production
├── .eslintrc
├── .gitignore
├── .prettierrc
├── package.json
├── README.md
└── Dockerfile

```

< 표 12 프로젝트 전체 파일 구조 >

.husky는 husky 구동을 위한 설정이 들어있는 디렉토리입니다. prettier와 eslint를 통해 컨벤션을 정하고 프로젝트를 하고 있기는 하지만 commit을 할 때 검사를 할 수 없어서 올바른 스타일의 코드가 올라갈 수 있습니다. 따라서 husky를 사용해 pre-commit 단계에서

eslint를 활용해 코딩 스타일은 한번 검증하고, 수정한 다음 commit을 합니다.

config, scripts파일에는 babel, webpack, react에 대한 설정파일이 들어가 있습니다. CRA(Create React App)을 사용해 리액트 라이브러리를 사용하고 있기 때문에 원래대로라면 이 파일들이 노출되지 않지만 전역상태관리툴로 mobx를 사용하기 위해서 eject 했습니다.

public 디렉토리는 static file들이 들어가는 디렉토리 입니다.

src 디렉토리는 실제 구현 코드가 들어가는 위치입니다. components에는 페이지 하나에 해당하는 컴포넌트의 하위 컴포넌트들이 들어가 있습니다. lib 디렉토리는 기능 구현을 위해 사용된 기타 함수가 들어가 있습니다. pages에는 url path에 따라 보여주어야 할 화면에 해당하는 컴포넌트가 들어가 있습니다. stores에는 mobx파일이 들어가 있습니다.

.env.development는 개발에 사용될 환경변수 입니다. .env.production은 실제 배포시 사용할 환경변수입니다.

.eslintrc와 .prettierrc는 각각 eslint와 prettier의 설정파일 입니다.

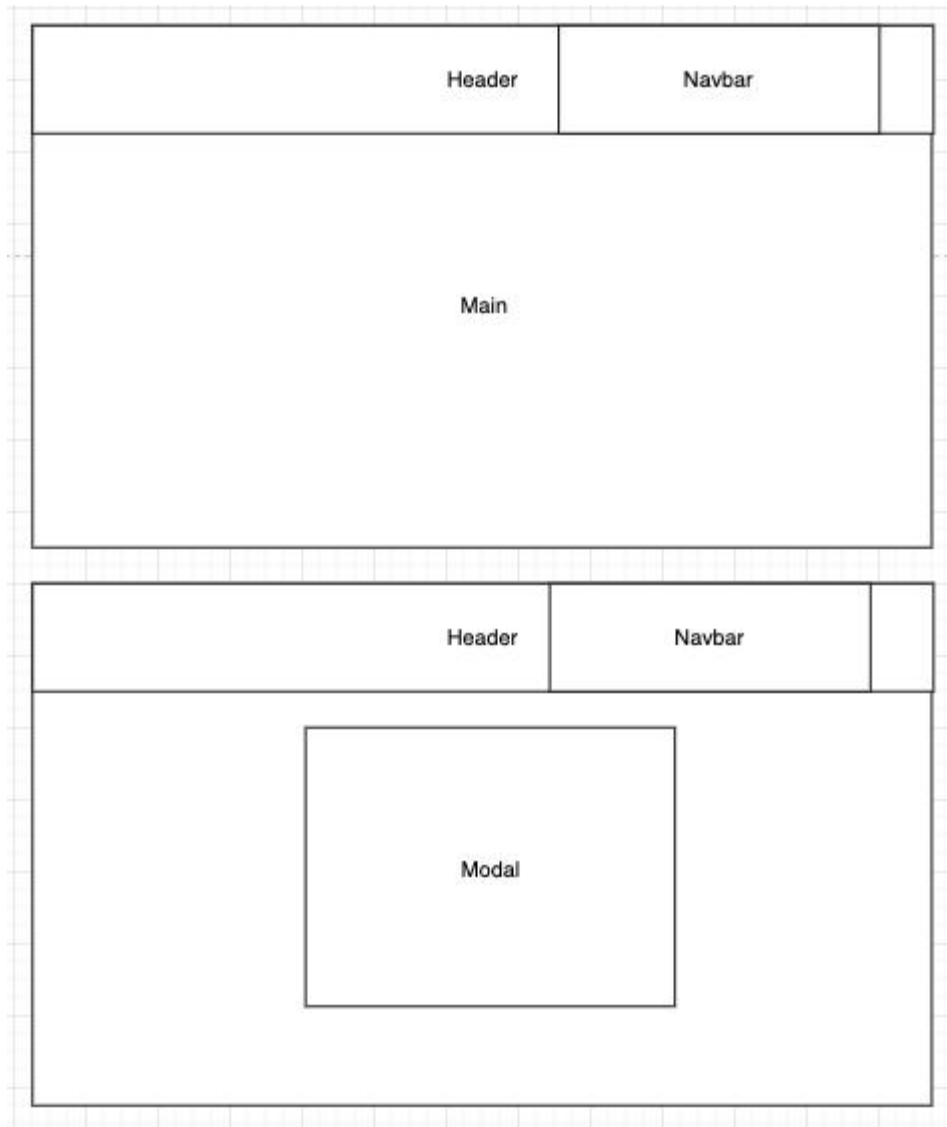
.gitignore는 git에 commit할때 무시할 파일과 디렉토리 리스트입니다.

Dockerfile은 frontend container를 만들기 위한 image파일을 만들때 사용할 파일입니다.

## 4.7. 화면 설계

### 4.7.1. 전체화면설계

전체 화면의 대략적인 설계는 다음과 같습니다. 모든 페이지에 navigation bar가 포함된 header가 공통적으로 들어갑니다. 페이지에 따라 달라지는 부분은 main부분 입니다. Modal창의 경우 화면 정 중장에 띄워지게 됩니다.



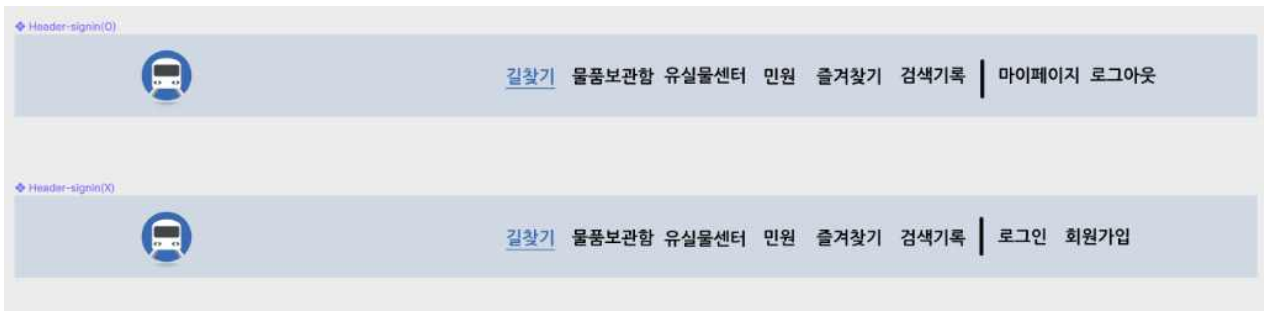
< 그림 6 전체화면설계도 >

위에서 설계한 전체 화면 설계도를 바탕으로 각 기능에 해당하는 실제 디자인을 다음과 같이 구상했습니다.

## 4.7.2 Figma를 사용한 실제 구현 화면

### (i). 헤더

모든 페이지에 공통적으로 들어가는 헤더는 다음과 같은 디자인을 가지고 있습니다. 로그인 일 하지 않을 상태라면 로그인, 회원가입 버튼이 나오고 로그인을 했다면 마이페이지, 로그아웃 버튼이 나옵니다.



< 그림 7 페이지 헤더 >

### (ii). 메인페이지

메인 페이지는 경로를 검색할 수 있는 페이지입니다. 최단거리, 최소시간, 최소비용 중 하나의 옵션을 선택하고 출발역, 도착역을 명시한 뒤 검색 버튼을 누르면 검색 결과 모달창이 나오게 됩니다.



< 그림 8 메인화면 >

(ii-1). 메인페이지(경유지 설정 선택시)



< 그림 9 메인화면-경유지 선택 >

(ii-2) 메인페이지(지하철 노선도에서 역 선택시 모달창 디자인)



< 그림10 메인화면-노선도에서 역 선택 >

### (ii-3) 메인페이지(유저가 입력한 정보에 대한 에러메시지 위치)

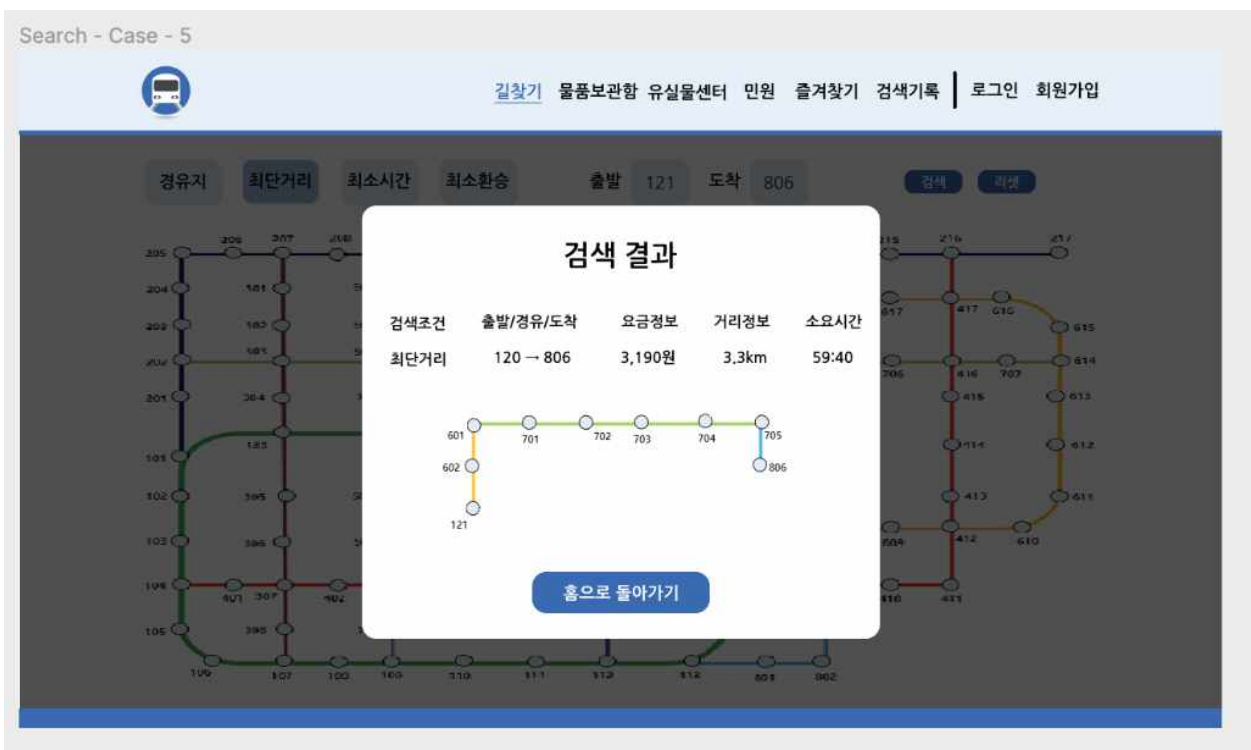


< 그림 11 메인화면-에러메시지 위치 >

에러메시지1은 최단거리, 최소시간, 최소비용 옵션을 하나를 선택하지 않았을때 뜨는 에러메시지 위치입니다.

에러메시지2는 출발역, 도착역, 경유지(경유시사용 선택시)을 명시하지 않았을때 뜨는 에러메시지 위치입니다.

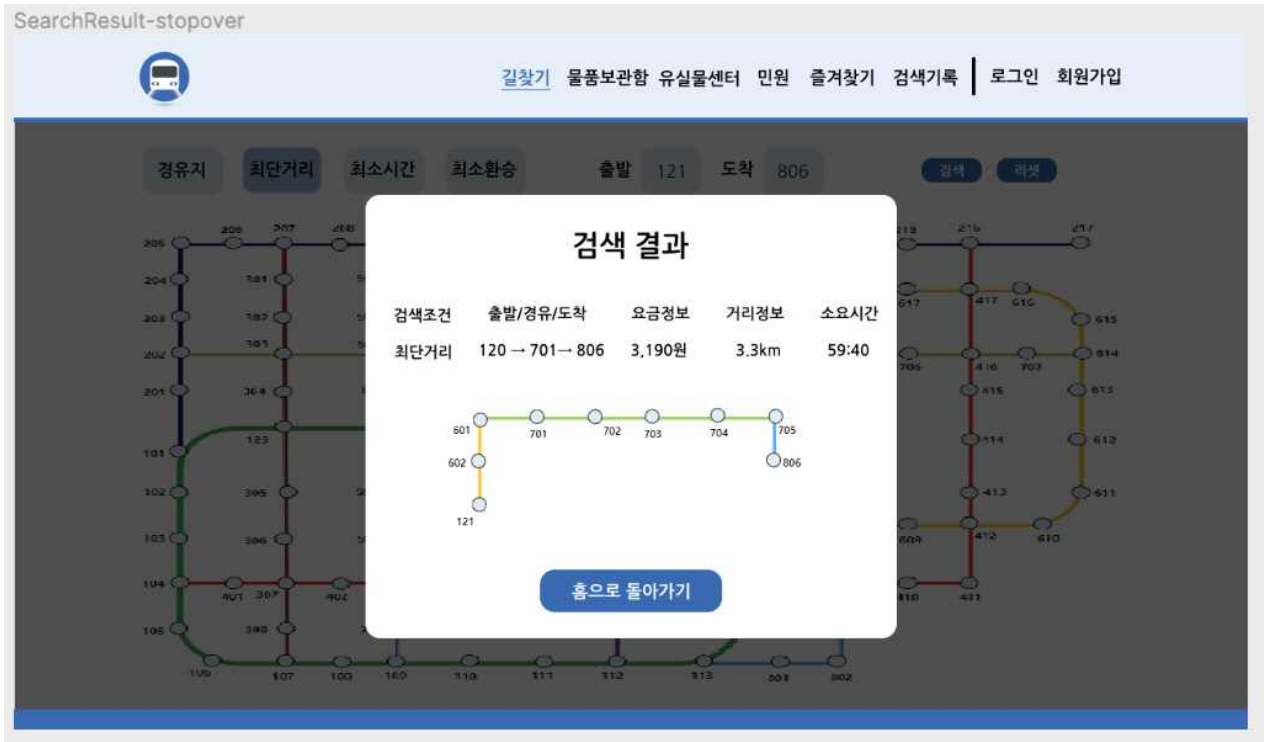
### (iii) 검색결과



< 그림 12 검색결과-경유지 설정X >

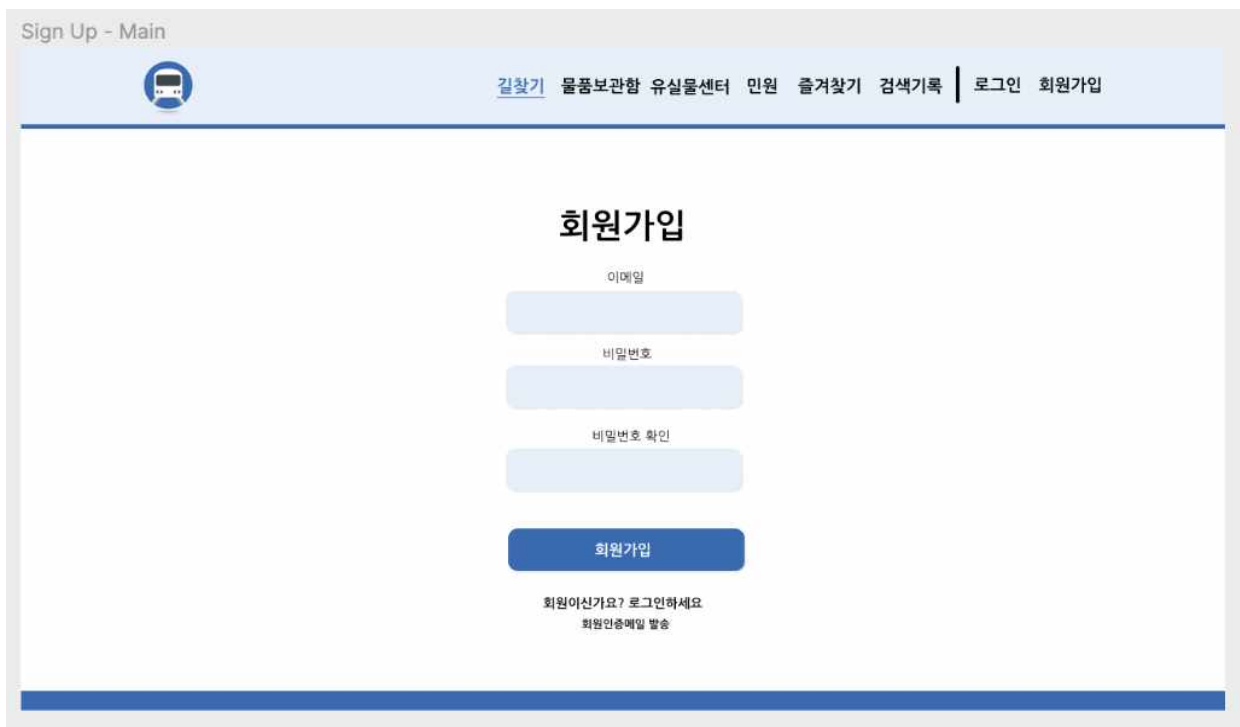
검색 결과는 모달창으로 구현했습니다. “홈으로 돌아가기” 버튼을 누르면 모달창이 꺼집니다. 검색 결과 모달창에는 최단거리, 최소시간, 최소환승 중 선택한 검색조건, 입력한 출발점, 경유지(경유지를 선택했을 경우), 도착점, 선택한 검색조건에 해당하는 최소비용과 그에 상응하는 다른 비용이 나옵니다.

### (iii-1) 검색결과(경유지 설정시)



< 그림 13 검색 결과-경유지 설정O >

### (iv). 회원가입

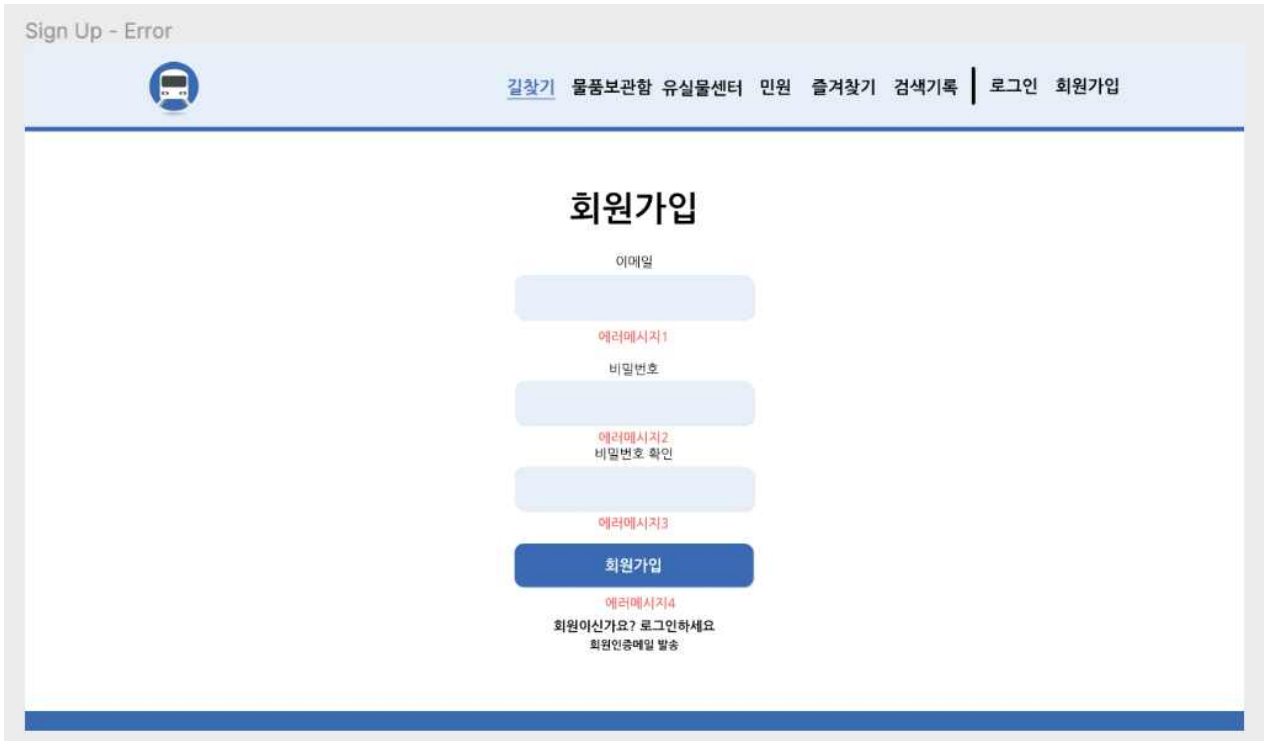


< 그림 14 회원가입창 >



회원가입 창에서는 유저가 이메일, 비밀번호, 비밀번호확인 창을 모두 입력해야 합니다. 여기서는 정규식을 활용해 1차적으로 이메일 형식이 맞는지, 비밀번호가 조건(8자 이상, 알파벳, 숫자, 특수문자 모두 포함)에 만족하는지를 검사합니다. “회원가입인가요? 로그인하세요”를 클릭하면 로그인 창으로 이동하고 “회원인증메일 발송”을 클릭하면 회원인증 메일을 재발송하는 창으로 이동합니다.

#### (iv-1) 회원가입(에러메시지 위치)



< 그림 15 회원가입창-에러메시지 위치 >

에러메시지1은 이메일 정규식과 맞지 않을 경우, 공백으로 둘 경우 뜨게됩니다.

에러메시지2는 비밀번호 정규식을 만족하지 않을 경우 뜨게 됩니다.

에러메시지3은 비밀번호와 비밀번호확인이 일치하지 않을 경우, 비밀번호 정규식을 만족하지 않을 경우 뜨게 됩니다.

에러메시지4는 클라이언트에서 확인할 수 없는 정보에 대해 에러메시지를 띄웁니다. 예를 들어 이미 사용중인 이메일일 경우 등이 있습니다.



## (v). 로그인 창

< 그림 16 로그인창 >

로그인 창에서는 유저가 이메일, 비밀번호창을 모두 입력해야 합니다. 여기서는 정규식을 활용해 1차적으로 이메일 형식이 맞는지, 비밀번호가 조건(8자 이상, 알파벳, 숫자, 특수문자 모두 포함)에 만족하는지를 검사합니다. “회원이 아니신가요? 가입하세요”를 클릭하면 회원가입 창으로 이동하고 “회원인증 메일 발송”을 클릭하면 회원인증 메일을 재발송하는 창으로 이동합니다.

### (V-1) 로그인창(에러메시지 위치)

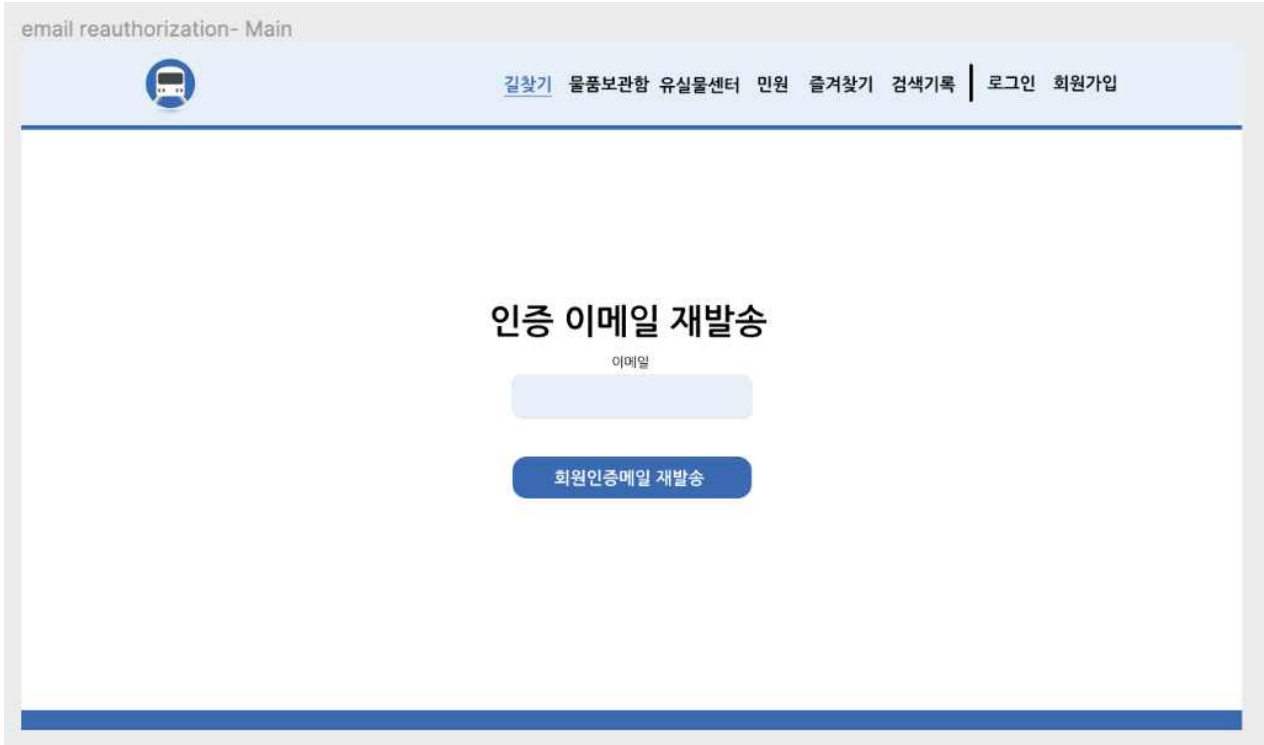
< 그림 17 로그인창-에러메시지 위치 >

에러메시지1은 이메일 정규식과 맞지 않을 경우, 공백으로 둘 경우 뜨게됩니다.

에러메시지2는 비밀번호 정규식을 만족하지 않을 경우 뜨게 됩니다.

에러메시지3은 클라이언트에서 확인할 수 없는 정보에 대해 에러메시지를 띄웁니다. 예를 들어 가입하지 않은 계정, 회원가입을 했으나 이메일인증을 하지 않은 경우 등이 있습니다.

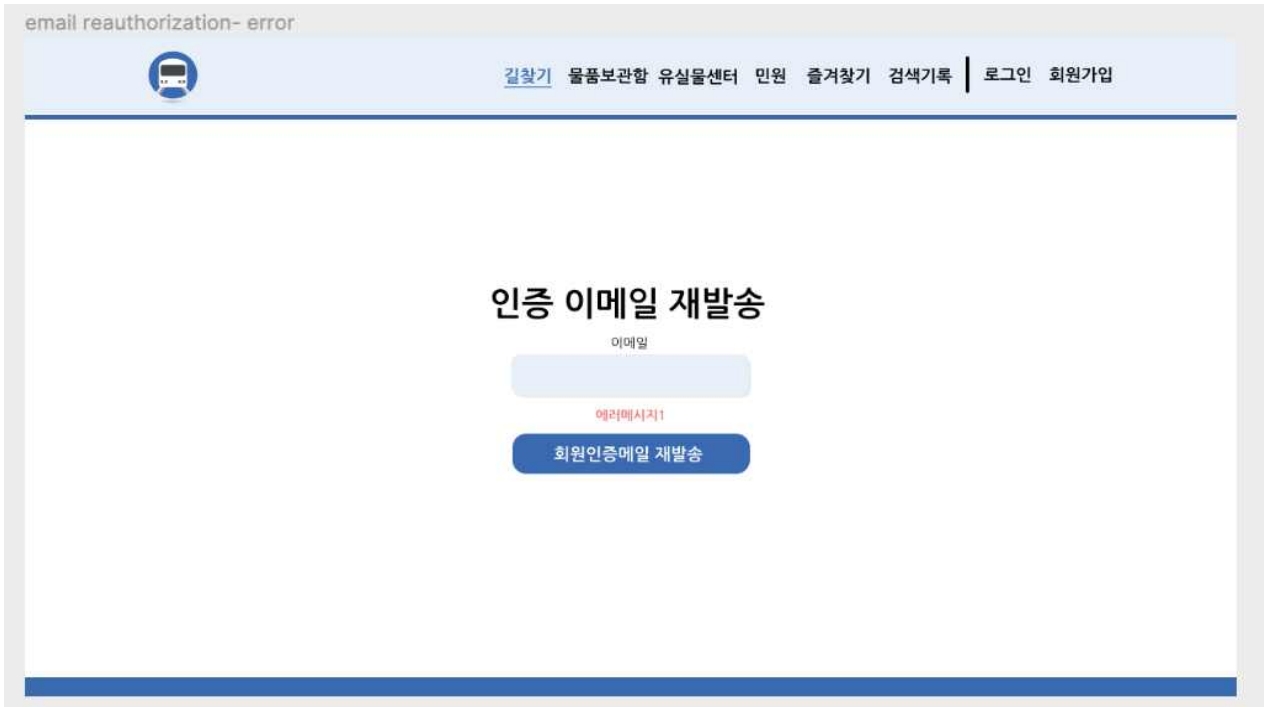
## (Vi). 인증 이메일 재발송 창



< 그림 18 인증 이메일 재발송 창 >

인증이메일은 발송 직후부터 1시간동안만 유효합니다. 따라서 만일 유저가 회원인증메일을 1시간 이내에 확인하지 않았을 경우 인증 이메일 재발송 창을 통해 이메일 인증을 다시 받아야 합니다. 이메일을 입력하고 “회원인증메일 재발송“을 클릭하면 인증 이메일이 재발송 됩니다.

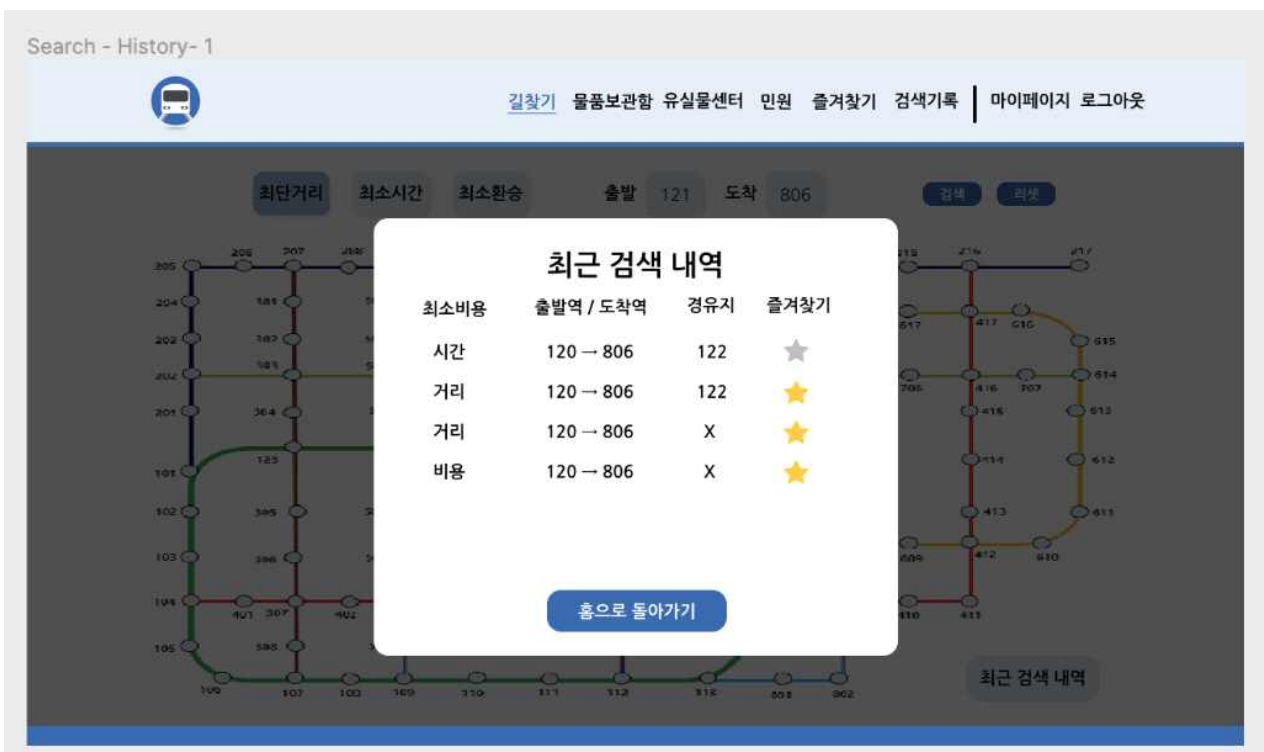
## (Vi-1). 인증 이메일 재발송 창(에러 메시지 위치)



< 그림 19 인증 이메일 재발송 창-에러 메시지 위치 >

에러메시지1은 이메일 정규식과 맞지 않을 경우, 공백으로 둘 경우, 서버에서 데이터에 오류가 있다고 판단한 경우 뜨게 됩니다. 서버에서 오류가 있다고 판단하는 경우는 이미 인증이 정상적으로 진행된 경우, 회원가입을 진행하지 않은 경우가 여기에 해당합니다.

## (Vii). 최근 검색 내역 모달창



< 그림 20 최근 검색 내역 모달창 >

로그인을 한 뒤 최근 검색 내역을 조회할 수 있습니다. 최근 검색 내역은 가장 최근에 검색한 7개의 내역이 나오게 됩니다. 각 검색 내역은 최소비용, 출발역, 도착역, 경유지, 즐겨

찾기 유무를 보여줍니다. 경유지가 없을 경우 해당 항목은 'X'표시로 나오며 즐겨찾기에 추가가 되어있다면 노란색 별을, 아니라면 회색 별로 나옵니다. 검색 내역 중 원하는 것을 클릭하면 최근 검색 내역 모달창이 꺼지고 검색 결과 창으로 이동해 클릭한 검색 내역에 해당하는 검색 결과를 보여줍니다.

### (Viii). 유실물센터 창

L&F - Main

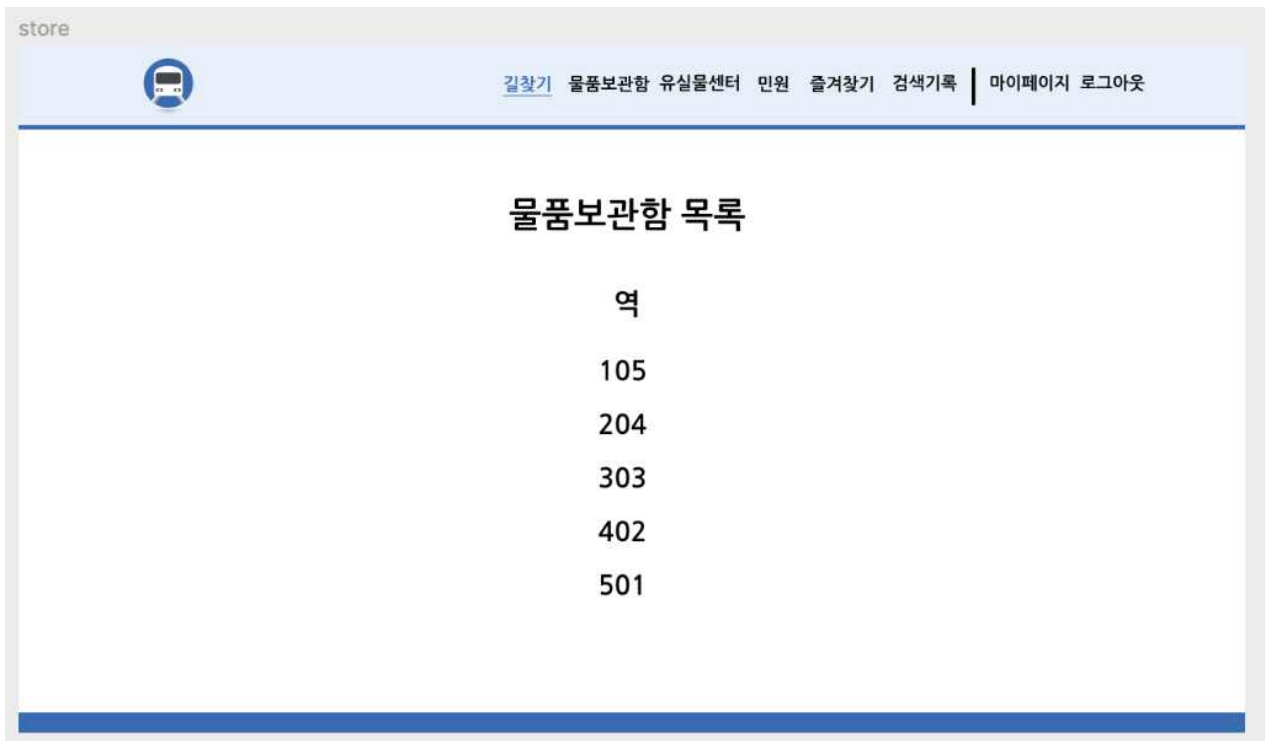

[길찾기](#)
[물품보관함](#)
[유실물센터](#)
[민원](#)
[즐거찾기](#)
[검색기록](#)
|
 [마이페이지](#)
[로그아웃](#)

역	전화번호	운영시간
105	02-478-9545	09:00-18:00
204	031-330-8674	10:00-18:00
303	02-2784-1278	10:00-19:00
402	032-598-7365	12:00-21:00
501	02-864-1579	09:00-18:00

#### < 그림 21 유실물센터 조회창 >

유실물센터 위치를 조회하는 창입니다. 이 창에서는 유실물 센터의 위치, 전화번호, 운영시간을 보여줍니다.

## (IX). 물품보관함 창



< 그림 22 유실물센터 조회창 >

물품보관함 목록을 조회하는 창입니다. 이 창에서는 물품보관함이 위치한 역을 조회할 수 있습니다.

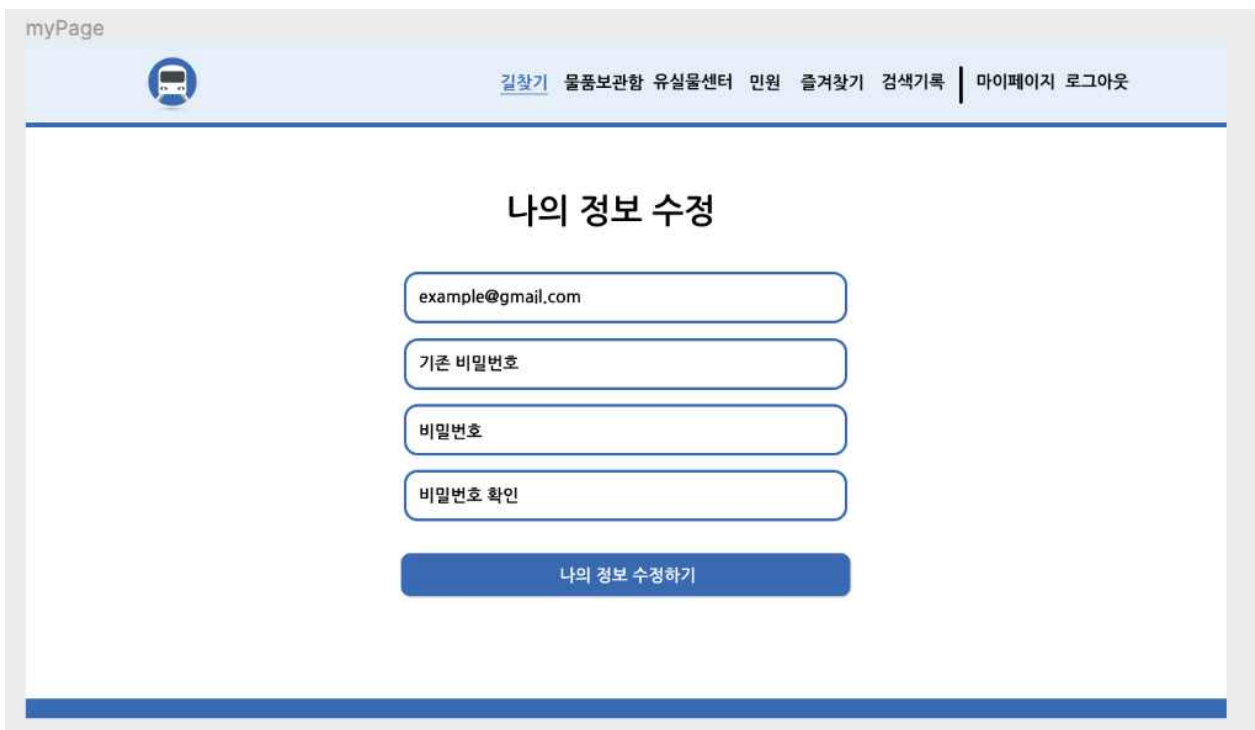
## (IX). 즐겨찾기 창




< 그림 23 즐겨찾기 창 >

로그인을 하면 유저가 만든 즐겨찾기 목록을 볼 수 있습니다. 즐겨찾기 목록은 최대 10개 까지 지정할 수 있습니다. 즐겨찾기 목록은 검색기록 모달창과 같이 최소비용, 출발역, 도착역, 경유지, 즐겨찾기를 설정할 수 있습니다. 즐겨찾기 목록을 해제하고 싶다면 별을 누르면 됩니다. 또 한 즐겨찾기 목록중 하나를 클릭하면 그에 해당하는 검색 결과 모달창이 띄워집니다.

## (X). 마이페이지 창



myPage

 [길찾기](#) [물품보관함](#) [유실물센터](#) [민원](#) [즐거찾기](#) [검색기록](#) | [마이페이지](#) [로그아웃](#)

### 나의 정보 수정

### < 그림 24 마이페이지 창 >

마이페이지 창에서는 개인 정보를 수정할 수 있습니다. 이메일을 수정하면 인증 메일을 다시 받아야 하며 비밀번호 역시 바꿀 수 있습니다. 이메일만 바꾸고 싶은 경우에는 이메일만 입력을 하면 되고 비밀번호를 바꾸고 싶다면 기존비밀번호, 비밀번호, 비밀번호 확인을 입력해야 합니다.

## (X-1). 마이페이지 창(에러메시지 위치)

myPage

길찾기 | 물품보관함 | 유실물센터 | 민원 | 즐겨찾기 | 검색기록 | 마이페이지 | 로그아웃

### 나의 정보 수정

example@gmail.com  
에러메시지1

기존 비밀번호  
에러메시지2

비밀번호  
에러메시지3

비밀번호 확인  
에러메시지4

나의 정보 수정하기  
에러메시지5

< 그림 25 마이페이지 창-에러메시지 위치 >

에러메시지1: 이메일이 정규식을 만족하지 않을 경우, 공백일 경우 뜨게 됩니다

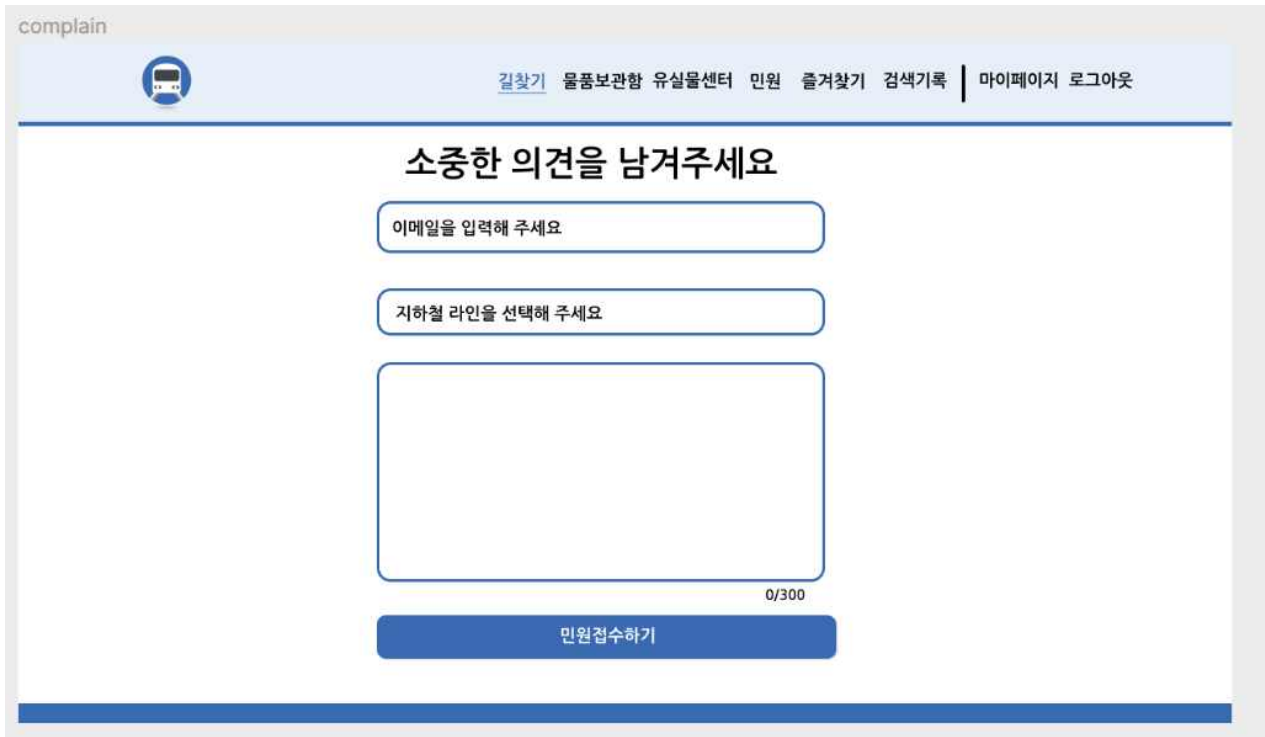
에러메시지2: 기존 비밀번호가 정규식을 만족하지 않을 경우, 공백일 경우 뜨게 됩니다.

에러메시지3: 비밀번호가 정규식을 만족하지 않을 경우, 공백일 경우 뜨게 됩니다.

에러메시지4: 비밀번호확인이 정규식을 만족하지 않을 경우, 공백일 경우, 비밀번호와 일치하지 않을 경우 뜨게 됩니다.

에러메시지5: 클라이언트에서 확인하지 못하는 에러가 발생할 경우 이 위치에 에러메시지가 뜨게 됩니다. 예를 들어 이미 사용중인 이메일 등이 있습니다.

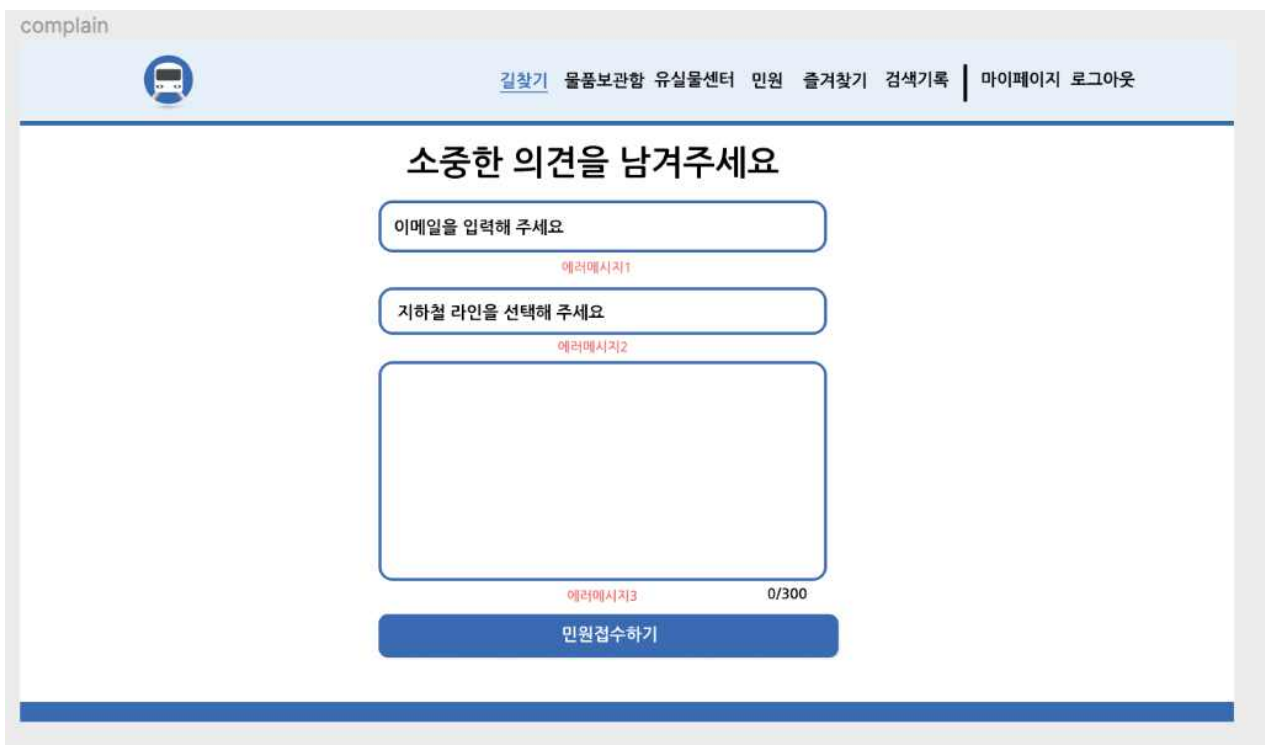
## (Xi). 민원창



< 그림 26 민원창 >

유저가 본인의 이메일을 입력하고 지하철 라인을 선택한 뒤 민원 내용을 최대 300글자 까지 입력하면 민원이 접수되고 접수한 내용이 입력한 이메일로 발송됩니다.

### (Xi-1) 민원창(에러메시지 위치)



< 그림 27 민원창-에러메시지 위치 >

에러메시지1: 이메일이 정규식을 만족하지 않거나 공백일 경우 뜨게 됩니다.





min\_time\_other\_values: 경로에 해당하는 최단시간과 상응하는 거리, 가격을 담고있습니다.

min\_time: 최단시간에 대응되는 경로를 담고 있습니다.

user: 유저에 대한 정보를 담고있습니다.

current\_searched: 유저가 최근에 검색한 기록을 담고 있습니다.

station\_bookmark: 유저가 추가한 즐겨찾기 목록을 담고 있습니다.

auth\_email: 회원가입을 새로 한 유저가 이메일 인증을 할 때 사용되는 랜덤 키값을 담고 있습니다.

token: 유저가 로그인 상태를 유지할 때 accessToken이 만료되면 accessToken 재발급을 위해 refreshToken을 담고 있습니다.

lost\_and\_found: 분실물센터에 대한 정보를 담고 있습니다.

storage\_box: 물품보관함에 대한 정보를 담고 있습니다.

## 5. 설계 계획

### 5.1. 일정표

<개발 일정>

번호	내용	9월				10월				11월				12월	기간 (주)
		1	2	3	4	1	2	3	4	1	2	3	4	1	
1	프로젝트 준비														2
2	UI/UX 디자인														1
3	client 개발														9
4	server 개발														9
5	테스트														2
6	시연 준비														2

< 표 13 개발 일정표 >

### 5.2. 업무 분담

이름	역할
김윤기	팀장, 서버, 프론트, 인프라 담당
박소민	디자인, UI구현 담당
김명비	문서담당
안선영	문서담당

< 표 14 팀원 업무 분담 표 >

## 6. 설계 평가

### 사용자 평가

평가항목	내 용
사용성	어플리케이션을 사용하면서 쉽게 조작이 가능한가?
경제성	어플리케이션으로 인해 지하철 이용이 원활해지고 간편해 졌는가?
정확성	결과가 제대로 나오는가?

< 표 15 사용자평가표 >

**사용성:** 본 서비스는 웹을 기반으로 하고 있기 때문에 높은 접근성을 갖고 있습니다. 또한 navigation bar에서 모든 기능에 접근할 수 있고 사용자가 입력한 정보가 유효하지 않을 경우 해당 위치에 경고 문구를 넣는 식으로 디자인을 해서 사용자가 더 빠르게 원하는 정보를 얻을 수 있게 했습니다. 따라서 사용성은 우수하다고 판단됩니다.

**경제성:** 위의 설문조사를 통해 사용자가 요구하는 경유지를 포함한 경로찾기, 물품함 위치, 민원 등을 추가해 편의를 대폭 늘렸습니다. 또 한 지하철 자체 내부의 큰 변화가 없다면 대규모 업데이트를 할 일이 없을 것이라 판단되므로 개발이 되고 나면 서버 유지보수 정도만 필요할 것이라 예상됩니다. 따라서 경제성 역시 우수하다고 판단됩니다.

**정확성:** permutation을 구현해 하나의 정거장에서 다른 정거장으로 가는 모든 경우의 수를 구하였고 플로이드와샬을 사용해 최단비용을, 플로이드와샬을 하면서 backtracking을 활용해 실제 경로를 구했습니다. 또한 여러 테스트케이스를 만들어 정확도를 체크했고, 체크했던 테스트 케이스를 모두 통과했습니다. 따라서 정확성이 높습니다.

### 자체 평가 (독립적 평가)

평가항목	내 용
윤리성	어플리케이션의 비양심적 사용을 막을 수 있는가?
정확성	고객의 정보가 제대로 저장되는가? 경로검색이 제대로 되는가?
UI	프로젝트 설계 시 계획한 UI와 비교하여 개발 후 UI와 같은가?
보안 오류	회원의 정보가 다른 곳으로 유출되지 않는가?

< 표 16 자체평가표 >

**윤리성:** 회원가입시 실제 존재하는 이메일인지를 검증하기 때문에 본인이 직접 가입을 해야 개인정보와 관련된 부분을 사용할 수 있게 했습니다. 또 한 지하철 사용을 돕는 서비스인 만큼 서비스가 악의적인 의도로 사용되지 않을 것이라 판단됩니다. 따라서 윤리성은 준수하다 판단됩니다.

**정확성:** 위 사용자 평가의 정확성에서 서술했듯이 경로검색 부분은 여러 개의 테스트케이스를

만들어 검증을 했고 테스트케이스를 모두 통과했습니다. 고객 정보의 경우 보다 정확한 정보를 받기 위해 클라이언트, 서버에서 이중으로 검증을 하고, 중복이 생기면 안되는 데이터, 또는 NULL이면 안되는 데이터에 대해서는 컬럼에 제약을 걸고, 중복여부를 체크해 유저의 정보를 정확히 저장하고 있습니다. 따라서 정확성은 높다고 생각합니다.

**UI :** 헤더에 위치한 네비게이션바에 사용자가 사용 가능한 기능에 대한 링크를 모두 넣어 놓았기 때문에 유저는 기능을 한눈에 파악할 수 있습니다. 또한 유저가 로그인을 했을 때와, 하지 않았을 때 네비게이션바에 보여지는 일부 링크를 다르게 해서 로그인 유무를 명확히 했습니다 (로그인을 안하면 로그인, 회원가입이 로그인을 하면 마이페이지, 로그아웃이 나옵니다). 더하여 유저 로그인이 필요한 기능을 유저가 로그인 하지 않고 접근하려 할 경우 로그인 창으로 리다이렉트하는 기능을 만들어 올바른 서비스 사용을 유도하고 있습니다. 따라서 UI는 우수하다고 생각합니다.

**보안 오류:** 세션-쿠키 방식을 사용해 로그인을 구현하면 쿠키가 탈취되었을 때 유저의 정보가 유출된다는 문제가 발생합니다. 따라서 본 서비스는 이를 막기 위해 JWT 방식으로 로그인을 구현했고 accessToken을 탈취당하지 않게 하기 위해 localStorage에 저장했습니다. 또 한 refreshToken은 쿠키에 저장했고 js로 접근하지 못하게 HTTPOnly 옵션을 사용했습니다. 또 한 쿠키가 탈취당하는 것을 막기 위해 HTTPS를 사용할 것입니다.

그럼에도 불구하고 XSS공격을 조금이라도 더 막기 위해 이메일, 비밀번호를 클라이언트에서 요청과 함께 보낼 때 클라이언트에서 한번, 백엔드에서 한번 2중으로 검사를 하고 있습니다.

서버에 접근하는 오리진을 오직 본 서비스의 클라이언트만으로 하기 위해 CORS를 허용할 때 Access-Control-Allow-Origin헤더를 \*가 아닌 클라이언트 오리진으로 하였습니다.

마지막으로 좀 더 보안을 완벽하게 하기 위해 배포단계에서 Helmet을 사용해 HTTP 헤더를 설정해 웹 취약성으로 보호하고 추가로 hpp모듈을 통해 http parameter pollution을 막을 것입니다. 따라서 보안오류가 발생할 수 있는 여지를 최대한 줄였습니다.

## 7. 참고자료

### 7.1. 국내 도서/ 논문

### 7.2. 해외 도서/ 논문

### 7.3. 기타 참고자료

-서울교통공사: <http://www.seoulmetro.co.kr/kr/cyberStation.do>

-사이버스테이션: <https://www.humetro.busan.kr/homepage/cyberstation/map.do>

-인천교통공사: <https://www.ictr.or.kr/main/railway/guidance/distance.jsp>

-카카오지하철 IOS 애플리케이션

-플로이드와샬 알고리즘:

<https://techbless.github.io/2020/11/11/C-%ED%94%8C%EB%A1%9C%EC%9D%B4%EB%93%9C-%EC%95%8C%EA%B3%A0%EB%A6%AC%EC%A6%98-%EA%B5%AC%ED%98%84%EA%B3%BC-%ED%94%8C%EB%A1%9C%EC%9D%B4%EB%93%9C-%EC%99%80%EC%83%AC-%EA%B2%BD%EB%A1%9C-%EC%B6%9C%EB%A0%A5-%EB%B0%A9%EB%B2%95>

# ● 설계/프로젝트 진행보고서(상세설계보고서)에 대한 채점기준 (Rubrics)

설계/프로젝트 진행보고서(상세설계보고서)를 평가하기 위한 총합적 채점 기준(Holistic Rubrics)					
평가항목	수행 수준				
	매우 우수 5	우수 4	보통 3	개선 가능 2	개선 필요 1
• 프로젝트 개발 환경과 플랫폼 등의 주요 도구 및 기술을 체계적으로 선정하고 설명하였습니다.	• 프로젝트 개발 환경과 타겟 플랫폼 등의 주요 도구 및 기술을 어느 정도 잘 선정하고 설명하였습니다.	• 프로젝트 개발 환경과 타겟 플랫폼 등의 주요 도구 및 기술을 어느 정도 잘 선정하고 설명하였습니다.	• 프로젝트 개발 환경과 타겟 플랫폼 등의 주요 도구 및 기술을 보통 정도로 선정하고 설명하였습니다.	• 프로젝트 개발 환경과 타겟 플랫폼 등의 주요 도구 및 기술을 약간 부실하게 선정하고 설명하였습니다.	• 프로젝트 개발 환경과 타겟 플랫폼 등의 주요 도구 및 기술을 제대로 선정하지 못하고, 설명도 매우 부실하다.
• 설계 내용의 충실성	• 설계내용이 매우 훌륭하게 충실히 기술되었습니다.	• 설계내용이 어느 정도 잘 충실히 기술되었습니다.	• 설계내용이 보통 정도로 기술되었습니다.	• 설계내용이 조금 부실하게 기술되었습니다.	• 설계내용이 많이 빠진 채로 기술되었습니다.
• 현실적 설계 제한 요건의 반영	• 경제성, 윤리성 등의 현실적 설계 제한 요건이 설계에 매우 잘 고려되고 반영되었습니다.	• 경제성, 윤리성 등의 현실적 설계 제한 요건이 설계에 어느 정도 잘 고려되고 반영되었습니다.	• 경제성, 윤리성 등의 현실적 설계 제한 요건이 설계에 보통 정도로 반영되었습니다.	• 경제성, 윤리성 등의 현실적 설계 제한 요건이 설계에 조금 부실하게 반영되었습니다.	• 경제성, 윤리성 등의 현실적 설계 제한 요건이 설계에 전혀 반영되지 않았다.
• 사용자 요구사항의 반영	• 사용자 요구사항이 설계에 매우 훌륭하게 반영되었습니다.	• 사용자 요구사항이 설계에 어느 정도 잘 반영되었습니다.	• 사용자 요구사항이 설계에 보통 정도로 반영되었습니다.	• 사용자 요구사항이 설계에 조금 부실하게 반영되었습니다.	• 사용자 요구사항이 설계에 전혀 반영되지 않았다.
• 보고서 서술 양식의 체계성	• 보고서 서술 양식(목차, 참고자료 등)이 매우 잘 갖추어졌다.	• 보고서 서술 양식(목차, 참고자료 등)이 어느 정도 잘 갖추어졌다.	• 보고서 서술 양식(목차, 참고자료 등)이 보통으로 갖추어졌다.	• 보고서 서술 양식(목차, 참고자료 등)의 체계가 조금 부족하다.	• 보고서 서술 양식(목차, 참고자료 등)의 체계가 거의 갖추어지지 않았다.

● 팀별 보고서의 자기 평가 채점표

- ◆ 기여도 평가 : 한 팀이 얻을 수 있는 5점 척도 평가의 총 점수는 (팀원 수 \* 3)+3점이다. 따라서 팀원들 점수의 합이 (팀원 수 \* 3)+3점을 넘어서는 안 됩니다. 이 총점을 각 팀원에게 5점 척도 (1~5점)의 점수로 나누어준 후, 가중치와 5점 척도 평가결과를 곱하여 개인별 기여도 평가점수를 산출합니다.
- ◆ 보고서 평가 : 채점기준표의 해당 평가항목 수행수준에 자기 평가한 내용을 표시(○ 또는 ✓)합니다. 그 다음에 이를 바탕으로 총합적 수행수준의 5점 척도(1~5점)의 평가결과를 정하고, 가중치와 5점 척도 평가결과를 곱하여 보고서 평가점수를 산출합니다.
- ◆ 총점 : 기여도 평가점수와 보고서 평가점수를 합하여 산출합니다.

보고서 제목	상세설계 보고서		
자기 평가하는 팀	4조참치	평가일	2021년 11월 15일

팀원 이름	기여도 평가			보고서 평가			총점 (G = C+F)	최종 교수 평가 점수	비고
	가 중 치 (A)	5점 척도 평가 결과 (B)	기여도 평가 점수 (C = AxB)	가 중 치 (D)	5점 척도 평가 결과 (E)	보고서 평가 점수 (F = DxE)			
김윤기	1	5	5	1	5	5	10		
김명비	1	3	3				8		
박소민	1	4	4				9		
안선영	1	3	3				8		
팀 합산 점수		15							