

캡스톤 기말 보고서

자리잇(지하철 좌석 공유 & 하차 알림 앱)

OMG 팀

2302554 박정현

2302592 정예슬

2302603 최수빈

2302605 최원희

2302608 한연수

목차

0	로고	3
0.1	OMG 로고	3
0.2	자리잇 로고	3
0.2.1	로고 디자인 방향성	3
0.2.2	주요 디자인 요소 분석	3
0.2.3	제작 방식 및 도구	3
0.2.4	기대 효과	4
1	프로젝트 계획서	4
1.1	개요	4
1.2	필요성 및 목적	4
1.3	주요기능	4
1.3.1	지원 노선 범위	4
1.3.2	혼잡도 시간대 분석 및 추천	4
1.3.3	개인 맞춤형 탑승 제안	4
1.3.3.1	세부 추진 과제	5
1.3.3.2	기대 효과	5
1.3.4	자리 공유를 위한(탑승역 입력 설정 후 하차역 입력)	5
1.3.5	머신러닝 Machine Learing /Deep Learning /딥러닝 기반 예측 시스템 구축	6
1.4	프로그램 개발 및 추진 방법	6
1.5	기대효과	7
1.6	기타	7
2	요구사항 명세서	7
2.1	혼잡도 시간대 분석 및 추천	7
2.2	실시간 칸별 혼잡도 예측	8
2.3	개인 맞춤형 탑승 제안	8
2.4	탑승역 설정 후 하차역 시간별	9
2.5	개발하면서 추가할 가능성이 있는 기능들	9
2.6	머신러닝/딥러닝 기반 예측 시스템 구축	10
2.7	회원 관리	10
2.7.1	회원관리(User Requirements)	10
2.7.2	주요 모듈 요구사항	11
2.7.2.1	실시간 칸별 혼잡도 예측	11
2.7.2.2	개인 맞춤형 탑승 제안	11
2.7.2.3	머신러닝/딥러닝 기반 예측 시스템	12
2.7.2.4	탑승역 설정 후 하차역 시간별 UI(색상) 변화	12
2.7.3	추가 고려 요구사항	12
3	분석	12
3.1	경쟁사 분석	12
3.1.1	또타 지하철, T map 대중교통, 지하철 종결자, 이번 정거장	12

3.2	지하철 혼잡도 생성 방법-----	16
3.2.1	PUZZLE 의 지하철 혼잡도 -----	16
3.2.2	혼잡한 지하철 구간 체크-----	18
3.2.3	열차의 시간대별 혼잡도-----	19
3.2.4	실시간 간별 혼잡도 예측-----	20
3.2.5	혼잡도 단계 구분-----	20
3.3	지하철 혼잡도 분석-----	20
3.3.1	서울교통공사 데이터 수집 및 전처리 -----	20
3.3.2	서울교통공사 노선별 지하철역 정보-----	21
3.3.3	서울교통공사 지하철 혼잡도 정보-----	21
3.3.4	진입 역 기준 칸 혼잡도-----	22
3.3.5	지하철 혼잡도 안내 구성 내용-----	22
3.3.5.1	열차의 시간대별 혼잡도 -----	23
3.3.5.1.1	지하철 열차 혼잡도 -----	23
3.3.6	실시간 간별 혼잡도 예측-----	24
3.3.6.1	혼잡도 단계 구분 -----	24
3.4	데이터 분석 기반 예측 시스템-----	25
3.5	데이터 수집 및 전처리-----	25
3.5.1	데이터 수집 -----	25
3.5.2	데이터 전처리 -----	25
3.6	예측 모델 개발-----	26
3.6.1	머신러닝 및 딥러닝 기법 적용 -----	26
3.6.2	최적 모델 선정-----	26
3.7	모델 학습 및 성능 평가-----	26
3.7.1	학습 과정 -----	26
3.7.2	성능 평가 지표 -----	26
3.8	데이터베이스 연동-----	26
3.9	결론 및 기대효과-----	26
4	설계-----	27
4.1	Architecture-----	27
4.2	Data 설계 : ER diagram-----	28
4.3	스키마-----	28
4.4	데이터베이스 상세 설계-----	28
4.4.1	회원 데이터베이스 -----	28
4.4.2	모듈 데이터베이스 -----	29
4.4.3	지하철 호선 데이터베이스-----	30
4.5	화면 설계-----	33
4.5.1	UI 설계 -----	33
4.6	LOGIC 설계-----	33
4.6.1	회원관리 -----	33

4.7	Interface 설계 : API 데이터 -----	36
4.7.1	SK OPEN API 이용절차-----	36
4.8	데이터 저장 방식 및 위치 설계-----	36
4.8.1	서비스 관련 데이터 -----	36
4.8.2	회원 정보 관련 데이터-----	36
4.8.3	각 데이터 저장 공간-----	37
4.8.4	추천 시나리오별 정리-----	37
4.8.5	전체 구조 요약-----	37
4.8.6	요약 정리-----	37
4.9	앱 색상 설계-----	37
5	코딩-----	38
5.1	GitHub -----	38
5.2	회원가입 실행화면-----	38
5.3	홈 화면 실행화면-----	38
5.4	모듈 초기 스케치-----	38
5.5	회원 가입 화면-----	39
5.6	메인 화면-----	41
5.7	홈 화면-----	42
5.8	시간표 화면-----	42
5.9	칸별 혼잡도 화면-----	43
5.10	좌석 선택 화면-----	44
5.11	하차 알림 설정 화면-----	46
6	테스팅-----	47
6.1	Test Case -----	47
6.1.1	테스트 개요-----	48
6.1.2	테스트 대상 -----	48
6.1.3	테스트 환경 -----	48
6.1.4	테스트 방법 -----	48
6.1.5	테스트 기준 -----	48
6.2	회원관리-----	48
6.3	전화번호 인증-----	48
6.4	로그인-----	49
6.5	회원정보 조회-----	49
6.6	비밀번호 변경-----	49
6.7	비밀번호 재설정-----	50
6.8	회원 탈퇴-----	50
6.9	통합 테스트 시나리오 표-----	50
7	회의록-----	51

0 로고

0.1 OMG 로고



0.2 자리잇 로고



0.2.1 로고 디자인 방향성

- 지하철과 관련된 서비스임을 시각적으로 즉시 전달
- 미니멀하고 IT 스러운 인상으로 모바일 앱에 어울리는 디자인
- 브랜드 정체성을 전달할 수 있도록 로고 아이콘과 텍스트를 결합

0.2.2 주요 디자인 요소 분석

- 지하철 아이콘 : 좌측에 위치한 지하철 이미지로, 앱의 주제를 즉각적으로 전달
- 텍스트 'JARIIT' : 얇고 세련된 산세리프체 사용
- 색상 : 코랄 계열의 친화적이고 따뜻한 이미지, 딥 그레이로 안정감과 신뢰감을 부여
- 전체 스타일 : 심플, 테크, 스마트한 느낌의 플랫 & 미니멀 디자인

0.2.3 제작 방식 및 도구

- AI 기반 로고 생성기([브랜드 마크 링크](#))+사용자 커스터마이징

- 키워드: Platform, Subway, Station,, Empty seat, Congestion, Route

0.2.4 기대 효과

- 시각적으로 지하철 관련 서비스임을 명확히 전달
- 스마트하고 신뢰감 있는 브랜드 이미지 형성

1 프로젝트 계획서

1.1 개요

출·퇴근 시간대에는 환승이 편리한 차량 구간에 승객이 집중되면서 칸별 혼잡도 차이가 크게 나타난다. 승객들은 보다 한산한 칸이나 좌석을 찾기 위해 이동하지만, 실제로는 사전에 정보를 확인할 방법이 없어 운에 의존해야 하는 경우가 많다. 이처럼 혼잡도 칸과 여유 있는 칸을 예측하기 어려운 점은 지하철 이용 편의성의 주요 불편 요소로 지적된다. 서울교통공사가 객실 혼잡도 안내 장치를 도입한 것처럼, 본 앱은 이러한 문제를 해결하기 위해 실시간 혼잡도 및 좌석 정보를 제공하는 것을 주요 목표로 한다.

1.2 필요성 및 목적

- 지하철은 출퇴근 시간대에 혼잡도가 극심해 좌석 부족 및 시간적, 거리적인 지하철 이용 문제가 발생하고 있다.
- 특히 하차 시점이 불분명한 승객으로 인해 좌석을 필요로 하는 사람들의 이용이 어렵고, 실시간 정보 부족으로 혼잡도나 좌석 상황을 예측하기 힘들다.
- 이를 개선하기 위해, 혼잡도 분석 및 하차 정보 공유가 가능한 앱이 필요하다.

1.3 주요 기능

1.3.1 지원 노선 범위

- 1호선 ~ 9 호선
 - 수도권 핵심 구간을 연결하는 1호선부터 9 호선까지의 전 노선을 포함.
- 신분당선
 - 강남에서 수원 방면으로 이어지는 신분당선을 지원하여, 주요 환승 및 이동 편의를 제공.
- 공항철도
 - 서울역에서 인천국제공항까지 연결되는 공항철도를 포함하여 공항 접근성을 강화.

1.3.2 혼잡도 시간대 분석 및 추천

- 서울교통공사 데이터를 활용해 출퇴근 시간대 혼잡 패턴 분석
- 덜 붐비는 시간대 추천으로 쾌적한 이용 유도

1.3.3 개인 맞춤형 탑승 제안

- 주요 업무
 - 이용 패턴 기반 추천: 사용자 이동 이력 학습 → 최적 시간·칸 자동 추천
 - 편의 기능 제공: 하차 알림, 좌석 추천 기능 포함
 - 지속적 개선: 사용자 피드백 수집 및 모델 성능 반영
 - UI/UX 설계: 개인화 추천 인터페이스, 알림 기능 UX 최적화
- 역할 및 의의
 - 개인 맞춤형 탑승 제안 기능은 단순 정보 제공을 넘어, 사용자의 실제 이동 습관에 기반한 맞춤형 서비스를 제공한다. (ex. 평소 7호선을 오전

9 시에 탑승하는 사용자는 ‘이번 주에는 8 시 50 분이 덜 혼잡합니다’라는 개인 최적화 알림을 받을 수 있음)

- 사용자는 더 효율적이고 쾌적한 이동 경험을 누리며 서비스는 차별화된 사용자 만족도를 확보할 수 있다.

1.3.3.1 세부 추진 과제

- 피드백 수집 체계 고도화
 - 추천 정확성 및 유용성 평가 → 사용자 의견을 모델 개선에 반영
- 알림 시스템 구체화
 - 푸시 알림, 앱 내 알림 등 다중 채널 설계 → 사용자 선택권 확대
- 모델 성능 비교·검증
 - 초기 버전 VS 개선 버전 성능 분석 → 추천 정확도 및 신뢰성 확보

1.3.3.2 기대 효과

- 사용자 만족도 제고: 맞춤형 알림 제공으로 체감 효용 극대화
- 혼잡 완화 기여: 분산 탑승 유도 → 지하철 혼잡도 개선
- 서비스 차별화: 단순 교통 정보 앱을 넘어선 스마트 모빌리티 플랫폼으로의 도약

1.3.4 자리 공유를 위한(탑승역 입력 설정 후 하차역 입력)

- 개발 배경 및 기능의 필요성
 - 지하철 이용 시 승객들은 단순히 열차에 탑승하는 것뿐만 아니라, 언제 하차할지·좌석이 언제 비게 될지를 알고 싶어 하는 경우가 많다.
 - 특히 출퇴근 시간대에는 대부분의 승객이 서서 이동해야 하며, 좌석이 비는 순간을 놓치지 않는 것이 중요한 편의 요소가 된다.
 - 현재까지는 해당 승객이 자리에서 움직일 때까지 기다려야 하므로 불편과 불확실성이 존재했다.
 - 따라서, 사용자가 앱에 하차역을 설정하면 자동으로 잔여 정차역 수를 계산하여 색상 변화로 시각화하는 기능은 다른 승객들이 직관적으로 “이 좌석이 언제쯤 비게 될지”를 파악할 수 있게 해준다.
- 기능 목적
 - 이 기능은 단순한 사용자 개인용 알림을 넘어서, 서 있는 승객이 그 좌석 사용자의 하차 시점을 예측할 수 있도록 돋는 기능이다. 좌석에 앉은 사람이 어느 역에서 내릴 예정인지를 앱을 통해 입력하면, 시스템이 자동으로 하차까지 남은 역 수를 측정하여 색상으로 나타내주게 된다.
 - 직접 말을 걸지 않아도, 또는 그 사람이 움직이지 않아도, 색상 정보만으로 해당 좌석 사용자의 하차가 가까워지고 있다는 사실을 알 수 있게 된다.
- 기술 구현 방식:
 - 탑승/하차역 입력
 1. 사용자가 앱에서 출발역과 도착역을 설정.
 2. 앱은 해당 노선의 전체 정차역 리스트를 불러오고, 시작점과 도착점을 기준으로 경로를 산출.
 - 남은 정차역 수 계산

- 열차 위치 API(실시간 도착 정보)와 사용자가 입력한 하차역을 비교.
현재 열차가 지나간 역부터 하차역까지의 남은 역 수를 계산.
- 색상 단계별 변화
 - 잔여 정차역 수에 따라 좌석 색상이 자동 변경.
 - 색상 예시
 1. 10 역 이상 남음 → 초록(여유)
 2. 6~9 역 남음 → 노랑(관심)
 3. 3~5 역 남음 → 주황(준비)
 4. 1~2 역 남음 → 빨강(임박)

1.3.5 머신러닝 Machine Learning /Deep Learning 기반 예측 시스템 구축(추후 개발)

- 공공데이터(승객 수, 열차 위치 등)를 활용한 혼잡도·좌석 점유 예측 모델 개발
- 머신러닝/딥러닝 적용으로 예측 정확도 향상
- 무엇을 예측하려는 것일까?

1.4 프로그램 개발 및 추진 방법

- ① 데이터 수집 및 분석
 - 서울교통공사 API를 통해 실시간 혼잡도, 칸별 인원, 시간대별 이용량 등 수집
 - 노선·역·요일별 이용 패턴 분석으로 데이터 기반 설계 지원
- ② 예측 모델 개발
 - 혼잡도 및 탑승 칸 위치 예측 모델 설계 및 개발
 - 데이터 전처리, 모델 학습 및 성능 최적화 반복 수행
- ③ 앱 개발 및 기능 구현
 - 목적지 기반 좌석 알림 및 추천 기능 구현
 - 좌석 상태 시각화 (예: 초록→주황→빨강)로 자연스러운 착석 유도
 - 예측 결과 시각화 UI 포함 (앱 내 표시 그래픽 등)
- ④ 사용자 테스트 및 고도화
 - 베타 테스트를 통해 피드백 수집 및 개선
 - 예측 정확도 향상, UI/UX 개선, 안정성 확보
 - 실사용을 고려한 기능 고도화 및 운영 체계 마련
- ⑤ Man-Month
 1. 계산 절차
 - 인원 수: 5 명
 - 프로젝트 기간: 2025년 3월 2일 ~ 2025년 11월 30일
 - 1인당 주간 투입 시간: 15 시간
 - 1개월 = 4.33 주 기준 사용
 - Man-Month = 160 시간 기준 사용
 2. 전체 주차 계산
 - 2025년 3월 2일(월) ~ 2025년 11월 30일(일)까지는 총 39 주 + 6 일
39.9 주로 계산
 3. 총 투입 시간

- 5 명 × 15 시간/주 × 39.9 주 = 2,992.5 시간 3

4. Man-Month

- 2,992.5 시간 ÷ 160 시간 = 18.70 Man-Month

5. 최종 결과:

- 총 Man-Month = 약 18.7 MM
- 본 프로젝트는 2025년 3월 2일부터 11월 30일까지 약 39.9 주간 진행되었으며, 총 5명이 주당 15시간씩 투입되었습니다.
- 이에 따라 총 투입 시간은 2,992.5 시간이며, 이를 기준으로 산출한 총 Man-Month 는 약 18.7 MM입니다.

1.5 기대효과

1. 승객 편의성 향상

- 하차역 입력으로 빠른 좌석 실시간 예측 및 착석 기회 제공
- 혼잡도 기반 최적 탑승 칸 및 환승 위치 안내
- 실시간 좌석 정보 제공으로 빠른 자리 확보 가능

2. 지하철 운영 효율성 증대

- 좌석 회전율 향상 및 승객 분산으로 공간 활용 최적화
- 데이터 기반 승객 흐름 분석을 통한 운영 개선(배차 간격, 열차 추가 등)

3. 사회적 효과

- 출퇴근 스트레스 완화, 편안한 이동 환경 제공
- 대중교통 활성화로 탄소 배출 감소 및 환경 보호 기여

4. 기술 확장성

- AI 기반 사용자 분석으로 좌석 이용 패턴 최적화 및 실생활 적용 가능

5. 비상 상황 대응

- 자연/혼잡 시 실시간 좌석 정보 공유 및 대체 경로 제공
- 앱 알림 기능을 통해 사고 및 운행 정보 신속 전달

1.6 기타

1. 기능 보완 및 확장

- 행선지 표시 및 좌석 배치도 제공으로 자리 양보 유도 및 하차 정보 공유
- 실시간 알림 및 좌석 추천으로 목적지 기반 좌석 이용 최적화
- 자리 예약 및 대기 알림 기능으로 서 있는 승객의 편의 향상

2. 보안 및 안정성

- 개인정보 보호: 행선지 정보 익명 처리, 수집 데이터 주기적 삭제
- 법적 준수: 개인정보 보호법, 위치정보법 등 관련 법률 철저히 준수
- 안정성 확보: 서버 확장성 고려, 긴급 상황 대응 및 오류 방지 시스템 구축

3. 지하철 시스템 연계

- 서울교통공사 등과 협업하여 실시간 열차·좌석 정보 정확도 강화

2 요구사항 명세서

2.1 혼잡도 시간대 분석 및 추천

- C-01 시간대별 승객 수 데이터 수집
- C-02 요일별 혼잡도 반영

- C-03 시간대별 평균 혼잡도 시각화
- C-04 혼잡도 예측 후 추천 제공
- C-05 과거+실시간 데이터 결합으로 정확도 개선

ID	분류	요구사항	세부설명
C-01	혼잡도 분석	데이터 수집	시간대별 지하철 승객 수 데이터를 수집.
C-02	혼잡도 분석	혼잡도 수치화	요일(평일/주말)에 따른 혼잡도 변화를 반영.
C-03	혼잡도 분석	혼잡도 시각화	특정 시간대별 평균 혼잡도를 시각화하여 제공.
C-04	혼잡도 분석	예측 추천	이용자가 원하는 시간대를 입력하면 혼잡도를 예측해 추천.
C-05	혼잡도 분석	정확도 개선	과거 데이터와 실시간 데이터를 결합해 정확도를 개선.

- 시스템은 서울교통공사에서 제공하는 데이터를 기반으로 시간대별 지하철 승객 수를 수집해야 합니다.
- 이렇게 수집된 데이터를 요일(평일/주말) 기준으로 구분하여 혼잡도를 수치화하고, 특정 시간대별 평균 혼잡도를 시각적으로 제공되어야 합니다.
- 또한 사용자가 원하는 시간대를 입력하면 해당 시간대의 혼잡도를 예측하여 상대적으로 쾌적한 시간대를 추천할 수 있어야 합니다.
- 더 나아가 과거 데이터와 실시간 데이터를 결합해 예측 정확도를 지속적으로 개선할 수 있습니다.

2.2 실시간 칸별 혼잡도 예측

ID	분류	요구사항	세부설명
R-01	혼잡도 예측	데이터 수집	열차 위치 및 탑승 인원 데이터를 실시간으로 수집.
R-02	혼잡도 예측	혼잡도 수치화	칸별 혼잡도를 기준으로 잡은 숫자 형태로 수치화.
R-03	혼잡도 예측	혼잡도 시각화	혼잡도 결과를 기준 색상으로 시각화.
R-04	혼잡도 예측	혼잡도 갱신	혼잡도 데이터를 일정 주기(예: 30 초)마다 갱신.
R-05	혼잡도 예측	예측값 유지	서버 장애 시 최근 예측값을 유지하도록 함.

- 앱은 열차 위치 및 탑승 인원 데이터를 실시간으로 수집하여 칸별 혼잡도를 수치화해야 합니다.
- 이렇게 계산된 혼잡도는 색상으로 구분하여 시각적으로 제공되며, 일정 주기(예: 30 초)마다 자동 갱신되어야 합니다.

- 만약 서버 장애가 발생할 경우 최근 예측값을 유지하여 사용자 경험이 끊기지 않도록 보장해야 합니다.

2.3 개인 맞춤형 탑승 제안

ID	분류	요구사항	세부설명
P-01	혼잡도 분석	데이터 수집	사용자의 출발역, 하차역, 시간 정보를 입력.
P-02	혼잡도 분석	서비스 제공	사용자의 과거 탑승 패턴 데이터를 기반으로 제공.
P-03	혼잡도 분석	서비스 추천	혼잡도가 낮은 칸/시간대를 자동으로 추천.
P-04	혼잡도 분석	서비스 제공	장애인, 노약자, 임산부 등의 배려석 근접 칸 추천 기능을 제공. (추후 추가 예정)
P-05	혼잡도 분석	서비스 제공	사용자가 선호하는 좌석/칸을 설정하면 우선 추천.

- 사용자는 앱 내에서 출발역, 하차역, 시간 정보를 입력할 수 있어야 합니다.
- 시스템은 사용자의 과거 탑승 패턴 데이터를 기반으로 혼잡도가 낮은 칸이나 시간대를 추천합니다.
- 또한 장애인, 노약자, 임산부 등 교통 약자를 위해 배려석이 있는 칸을 우선적으로 추천할 수 있는 기능도 제공합니다(추후 확장 기능).
- 사용자가 특정 좌석이나 칸에 대한 선호를 설정하면, 시스템은 이를 반영하여 우선적으로 추천할 수 있어야 합니다.

2.4 빈자리 공유를 위한… 탑승역 설정 후 하차역 시간별

ID	분류	요구사항	세부설명
U-01	UI 변화	데이터 수집	사용자가 탑승역과 하차역을 설정할 수 있도록 함.
U-02	UI 변화	혼잡도 제공	이동 경로에 따른 혼잡도를 색상으로 표시.
U-03	UI 변화	혼잡도 구분	색상은 혼잡도 단계별로 직관적으로 구분.
U-04	UI 변화	데이터 반영	사용자가 시간대를 변경하면 UI도 즉시 업데이트됨.
U-05	UI 변화	색상 조정	예측 정확도를 고려해 색상 변화의 기준값을 동적으로 조정.

- 사용자는 탑승역과 하차역을 설정할 수 있어야 하며, 시스템은 이동 경로에 따른 혼잡도를 색상으로 표시해야 합니다.
- 색상은 혼잡도의 단계별 기준에 따라 직관적으로 구분되며, 사용자가 시간대를 변경하면 UI도 즉시 업데이트됩니다.
- 또한 예측 정확도를 고려하여 색상 변화의 기준값은 동적으로 조정될 수 있어야 합니다.

2.5 개발하면서 추가할 가능성 있는 기능들

ID	요구사항	설명
UR-1	최근 이용 노선 자동 저장 기능 제공	사용자가 자주 이용하는 노선을 자동 저장하여 이후 빠르게 재사용할 수 있어야 함.
UR-2	시스템 로그 및 오류 추적 기능 제공	관리자 또는 운영자는 시스템에서 발생한 오류나 사용자 로그를 확인할 수 있어야 함.
UR-3	사용자 피드백 수집 기능 제공	사용자가 앱 사용 중 문제점 또는 개선 의견을 제출할 수 있어야 함.
UR-4	칸별 혼잡도 시각화 기능 제공	지하철 칸별 혼잡도를 색상 또는 그래픽으로 시각화하여 제공해야 함.
UR-5	쾌적한 칸 추천 기능 제공	사용자에게 비교적 덜 불비는 지하철 칸을 추천해주는 기능이 있어야 함.

- 추후 개발 과정에서 확장 가능한 기능도 고려해야 합니다.
- 예를 들어, 사용자가 자주 이용하는 노선을 자동으로 저장해 빠르게 재사용할 수 있는 기능, 운영자가 시스템 로그 및 오류를 추적할 수 있는 기능, 사용자 피드백을 수집해 개선에 반영할 수 있는 기능 등이 있습니다.
- 또한 칸별 혼잡도를 그래픽 또는 색상으로 더욱 직관적으로 시각화하고, 이를 기반으로 쾌적한 칸을 추천하는 기능 역시 확장 가능합니다.

2.6 머신러닝/딥러닝 기반 예측 시스템 구축(추후 추가 예정)

ID	분류	요구사항	세부설명
M-01	혼잡도 예측	기술 적용	혼잡도 예측 모델로 머신러닝(Random Forest 등)과 딥러닝(LSTM 등)을 적용.
M-02	혼잡도 예측	성능 평가	데이터셋을 학습용/검증용/테스트용으로 분리하여 모델 성능을 평가.
M-03	혼잡도 예측	정확도 산출	예측 정확도를 지표(MAE, RMSE 등)로 산출.
M-04	혼잡도 예측	기술 적용	성능 비교 후 가장 효율적인 모델을 운영 서버에 적용.
M-05	혼잡도 예측	데이터 재학습	새로운 데이터가 누적되면 모델을 주기적으로 재학습.

- 혼잡도 예측의 정확성을 높이기 위해 머신러닝(Random Forest 등)과 딥러닝(LSTM 등) 알고리즘을 적용한 모델을 구축할 수 있습니다.
- 데이터셋은 학습용, 검증용, 테스트용으로 분리해 성능을 평가하며, MAE·RMSE 등의 지표를 활용해 정확도를 산출합니다.

- 여러 모델 간 성능 비교를 통해 최적의 모델을 운영 서버에 적용하며, 새로운 데이터가 누적되면 주기적으로 재학습을 수행해 모델 성능을 유지합니다.

2.7 회원관리

2.7.1 회원관리(User Requirements)

- UR-01 회원가입: 이메일/소셜 계정 기반 회원가입
- UR-02 로그인/인증: 이메일·비밀번호, 소셜 로그인, 본인 인증
- UR-03 회원 정보 조회: 이메일, 닉네임 등 조회
- UR-04 회원 정보 수정: 일부 개인정보 수정 가능
- UR-05 회원 탈퇴: 계정 삭제 및 개인정보 완전 삭제

ID	분류	요구사항	세부설명
UR-01	회원관리	회원가입 기능 제공	사용자는 이메일 또는 소셜 계정을 이용해 회원가입을 할 수 있어야 함.
UR-02	회원관리	로그인 및 인증 기능 제공	사용자는 이메일/비밀번호 또는 소셜 계정으로 로그인할 수 있어야 하며, 인증 절차를 통해 본인 확인이 가능해야 함.
UR-03	회원관리	회원 정보 조회 기능 제공	사용자는 자신의 등록 정보(이메일, 닉네임 등)를 조회할 수 있어야 함.
UR-04	회원관리	회원 정보 수정 기능 제공	사용자는 닉네임 등 일부 개인정보를 수정할 수 있어야 함.
UR-05	회원관리	회원 탈퇴 기능 제공	사용자는 언제든지 자신의 계정을 탈퇴할 수 있어야 하며, 탈퇴 시 개인정보는 삭제되어야 함.

- 앱은 기본적으로 회원 관리 기능을 제공해야 합니다.
- 사용자는 이메일 또는 소셜 계정을 통해 회원가입을 할 수 있으며, 로그인 시 이메일/비밀번호 또는 소셜 계정 인증 방식을 통해 본인 확인이 가능해야 합니다.
- 회원은 자신의 등록된 개인정보(이메일, 닉네임 등)를 언제든지 조회할 수 있고, 일부 개인정보(예: 닉네임)는 수정할 수 있어야 합니다.
- 또한 원활 경우 계정을 탈퇴할 수 있으며, 탈퇴 시 개인정보는 모두 삭제되어야 합니다.

2.7.2 주요 모듈 요구사항

2.7.2.1 실시간 칸별 혼잡도 예측

- R-01 열차 위치/탑승 인원 데이터 수집
- R-02 칸별 혼잡도 수치화
- R-03 혼잡도 색상 시각화
- R-04 일정 주기(30 초)마다 갱신
- R-05 장애 시 최근 예측값 유지

2.7.2.2 개인 맞춤형 탑승 제안

- P-01 출발·하차역·시간 정보 입력
- P-02 과거 탑승 패턴 기반 서비스 제공
- P-03 혼잡도 낮은 칸/시간 자동 추천
- P-04 배려석 근접 칸 추천 (추후 추가)
- P-05 선호 좌석/칸 우선 추천

2.7.2.3 머신러닝/딥러닝 기반 예측 시스템(추후 추가)

- M-01 RF/LSTM 등 예측 모델 적용
- M-02 데이터셋 분리 후 모델 성능 평가
- M-03 정확도 지표(MAE, RMSE 등) 산출
- M-04 성능 비교 후 최적 모델 적용
- M-05 신규 데이터 기반 주기적 재학습

2.7.2.4 탑승역 설정 후 하차역 시간별 UI(색상) 변화

- U-01 출발/하차역 설정
- U-02 경로 혼잡도 색상 표시
- U-03 단계별 색상 구분
- U-04 시간 변경 시 UI 즉시 반영

2.7.3 추가 고려 요구사항

- UR-1 최근 이용 노선 자동 저장
- UR-2 시스템 로그 및 오류 추적
- UR-3 사용자 피드백 수집
- UR-4 칸별 혼잡도 Heatmap 시각화
- UR-5 쾌적한 칸 추천 기능 제공

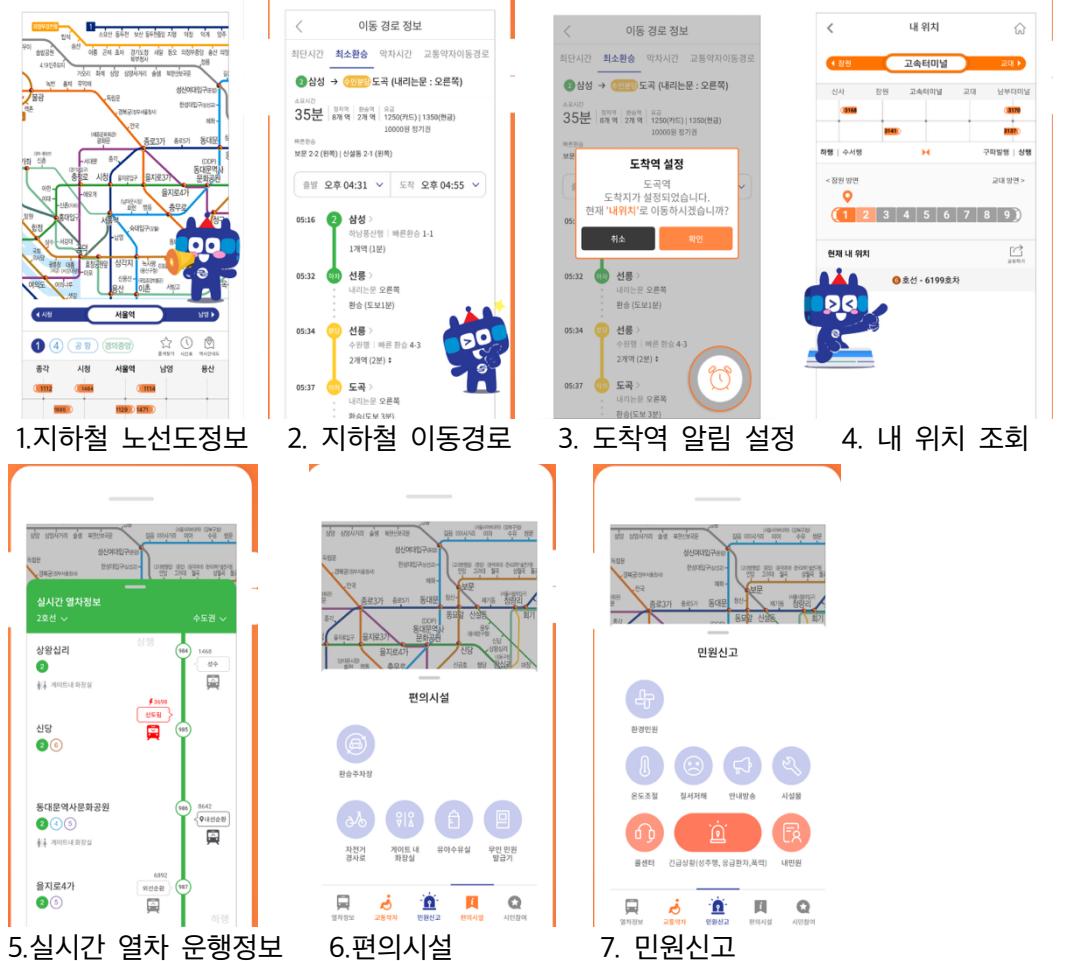
3 분석

3.1 경쟁사 분석

3.1.1 또타 지하철, T map 대중교통, 지하철 종결자, 이번 정거장

항목	내용
앱명	또타 지하철(TOTA)
운영사	한국철도공사(KORAIL)
플랫폼	Android / iOS
출시일	2022.11
다운로드 수	10 만+
주요 기능	- 철도 승차권 예매

	<ul style="list-style-type: none"> - 실시간 열차 위치 - 하차 알림 - 목적지 도착 알림 - 알림 음성 설정 - 지하철 노선 기반 위치 추적
자리잇과 공통점	<ul style="list-style-type: none"> - 하차알림 제공 - 실시간 열차 위치 기반 기능 일부 포함 - 지하철 노선 기반 서비스 제공



항목	내용
앱명	T map 대중교통(티맵)
운영사	티맵모빌리티(T map Mobility)

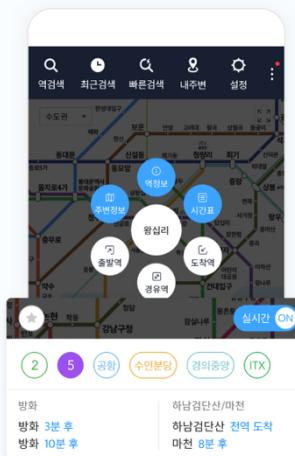
플랫폼	Android / iOS
출시일	2010.10
다운로드 수	1,000 만+
주요 기능	<ul style="list-style-type: none"> - 내비게이션 - 실시간 교통 정보 제공 - 대중교통 경로 안내 - 택시 호출(T map 택시) - AI 기반 경로 추천 - 열차별, 시간대별, 경로별 혼잡도를 분석
자리잇과 공통점	<ul style="list-style-type: none"> - 대중교통 사용자 편의를 위한 기능 제공 - 실시간 정보 활용 - 목적지 기반 서비스 구조



여유·보통·주의·혼잡 등 4 단계로 혼잡도 정보

항목	내용
앱명	지하철 종결자 (Smarter Subway) 
운영사	도플소프트 (Doppel Soft)
플랫폼	Android / iOS
출시일	2012년
다운로드 수	100 만+

주요 기능	<ul style="list-style-type: none"> - 전국 주요 도시 지하철 실시간 정보 제공 - 하차 알림 기능 (Wi-Fi 기반 위치 인식) - 최적 경로 안내 (최소 시간, 최소 환승) <ul style="list-style-type: none"> - 지하철 노선도 제공 - 주변 장소 검색 (화장실, 버스 정류장 등) - 다국어 지원 (영어, 중국어, 일본어 등)
자리잇과 공통점	<ul style="list-style-type: none"> - 하차 알림 기능 제공 - 지하철 실시간 정보 기반 - 지하철 이용 편의성 강화 목표



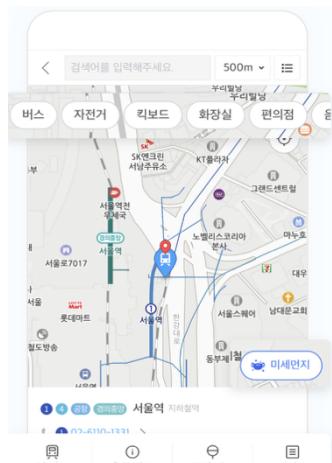
1. 실시간 지하철 도착 정보



2. 최적화 경로 검색



3. 하차 알림 기능



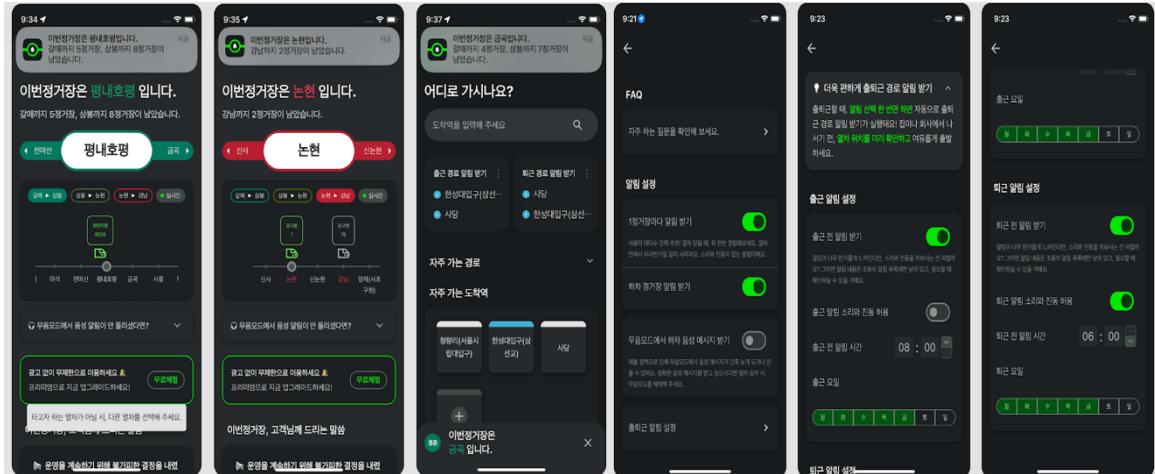
4. 주변 정보 / 도보 검색



5. 급행정보

항목	내용
앱명	이번 정거장

운영사	서울시립대학교 공공빅데이터 학술동아리 'POD'
플랫폼	Android / iOS
출시일	2014년
다운로드 수	50만+
주요 기능	<ul style="list-style-type: none"> - 하차 알림 기능 (지하철, 버스 모두 지원) - 목적지 알림 음성 지원 (TTS) - 노선 기반 자동 위치 추적 - 목적지 설정 시 진동/알림 기능 제공 - 백그라운드에서도 작동하는 하차 알림 - 약자나 수면 승객을 위한 전용 모드
자리잇과 공통점	<ul style="list-style-type: none"> - 하차 알림 시스템 제공 - 지하철 사용자 편의성 향상 목적 - 사용자 위치 기반 정보 제공



1. 이번정거장의 하차알림시스템

2. 알림 설정 기능(출근알림,퇴근알림)

3.2 지하철 혼잡도 생성 방법

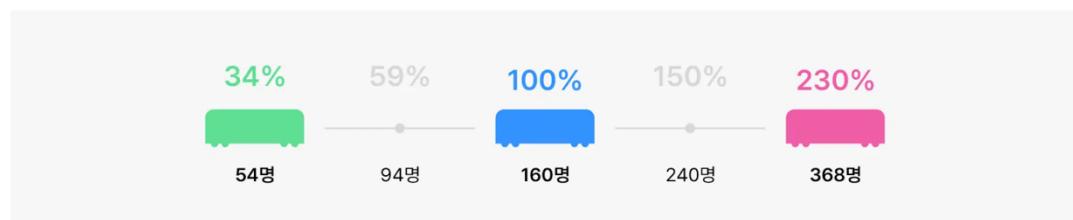
- 역사 및 선로 주변 기지국 또는 지하철 객차(칸) 내 AP 와의 모바일 통신 이력을 토대로 이용객들의 승하차 및 혼잡도 데이터를 생성

3.2.1 PUZZLE 의 지하철 혼잡도(예시)

- ① 추정: 지하철역 인근 기지국과의 모바일 통신 정보를 기반으로 탑승역으로 추정

- ② 지하철 이동 경로 추정
 - 가) 지하철을 타고 이동하면서 주변 기지국과 통신한 정보를 기반으로 이동 경로를 추정
 - 나) 지하철 객차(칸) 내 AP 와의 모바일 통신 정보를 기반으로 몇 번 칸에 탑승했는지 추정
- ③ 하차 추정: 지하철 경로 상 이동이 종료된 시점의 지하철 역에서 하차한 것으로 추정
- ④ 혼잡도 생성: 1~3 으로 추정된 지하철 탑승 정보를 기반으로 지하철 역사, 칸, 시간대별 탑승인원을 추정하여 열차/칸 혼잡도를 생성
- 지하철 혼잡도 기준
 - 지하철 혼잡도는 국토교통부의 지하철 혼잡도 기준을 따르고 있으며, 한 칸의 승객 수가 160 명일 때를 100%로 환산하여 제공

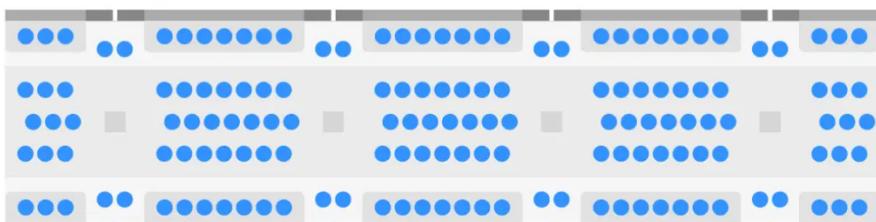
지하철 혼잡도 기준 *지하철 한 칸을 기준으로 합니다.



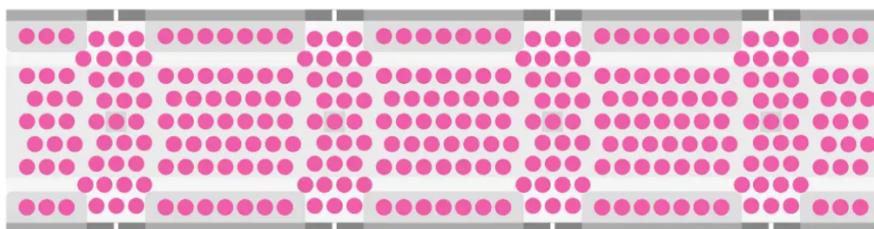
34% 좌석에 모두 앉은 상태



100% 통로에 3줄, 각 출입문에 2명씩 서 있는 상태



230% 통로에 5줄, 각 출입문에 30-40명씩 서 있는 상태



- 서울교통공사 기준

- 서울교통공사 기준: 혼잡도 4 단계 & 인원 수
- 서울 지하철 1량(칸)의 정원은 약 160 명

	색상 구분	혼잡도(%)	예상 탑승 인원 (160 명 기준)	체감 설명
여유	파란색	0% ~ 34%	0 ~ 약 54 명	좌석 여유 많고, 서 있는 사람 거의 없음
보통	초록색	35% ~ 99%	약 55 명 ~ 159 명	입석은 있지만 널널함, 이동 가능
주의	노란색	100% ~ 149%	약 160 명 ~ 239 명	입석 승객 많고, 승객끼리 접촉
혼잡	빨간색	150% 이상	240 명 이상	승객 간 밀착 심하고, 이동 및 스마트폰 사용 어려움

- 지하철 혼잡도 안내 페이지(서울교통공사 기준 등)에는 주로 다음과 같은 구성 내용이 포함.
- 혼잡도 정보를 이용자에게 명확하고 실용적으로 전달하기 위한 요소들로 이루어져 있음.

3.2.2 혼잡한 지하철 구간 체크

3.2.2.1 지하철 혼잡도

- 수도권 1~9 호선 및 신분당선의 지하철 혼잡도 데이터를 기반으로 출퇴근길 혼잡 구간을 선별

● 출근 시간대 (08:00~08:30)



- 출근 시간대는 서울 외곽에서 중심으로 진입하는 부분이 주로 혼잡한 양상을 보임
- 특히 업무 공간이 밀집된 강남권으로 향하는 방향에서 혼잡한 구간이 많이 나타남(고속터미널역, 잠실역, 방배역, 청계산입구역 등)

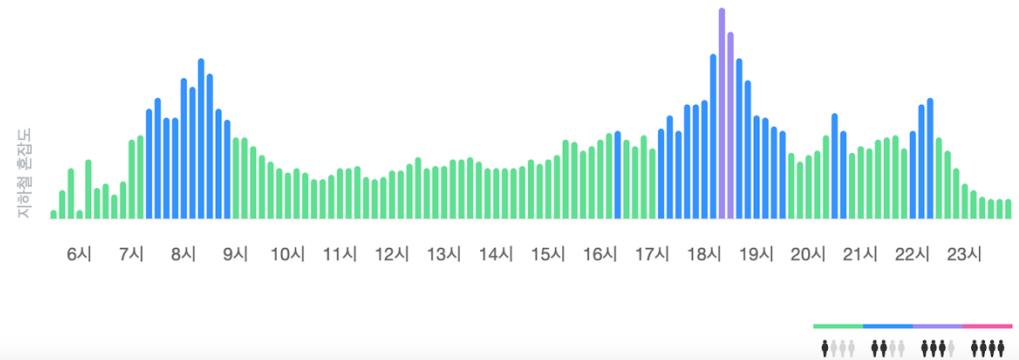
● 퇴근 시간대 (18:00~18:30)



- 퇴근 시간대는 반대로 서울 중심에서 외곽으로 이동하는 부분이 혼잡한 모습을 띠고 있음

3.2.3 열차의 시간대별 혼잡도

3.2.3.1 지하철 열차 혼잡도



- [2호선 왕십리역]으로 진입하는 열차의 시간대별 혼잡도

3.2.4 실시간 칸별 혼잡도 예측

- 칸별 혼잡도를 실시간 분석해 비교적 여유 있는 칸을 추천
- 서울교통공사 API 연동을 통한 정확한 정보 제공

3.2.5 혼잡도 단계 구분 (서울교통공사 기준)

단계	색상	혼잡도(%)	예상 탑승 인원 (1량=160명)	체감 설명
여유	파랑	0% ~ 34%	0 ~ 약 54명	좌석 여유 많고 서 있는 사람 거의 없음
보통	초록	35% ~ 99%	55명 ~ 159명	입석 승객 있지만 이동 가능
주의	노랑	100% ~ 149%	160명 ~ 239명	입석 많고 승객 간 접촉 발생
혼잡	빨강	150% 이상	240명 이상	승객 밀착 심함, 이동·스마트폰 사용 어려움

3.3 지하철 혼잡도 분석

3.3.1 서울교통공사 데이터 수집 및 전처리

- [서울시 지하철 혼잡도 통계](#)
- 전체적으로 … 혼잡도 증가 , 1 호선은 많이 증가…//….....

A	B	C	D	E	F	G
시점	서울교통공사					
평균	1호선	2호선	3호선	4호선	5호선	
2021	126.8	84.0	149.4	140.6	150.8	132.2
2023	136.2	109.1	144.0	140.1	166.2	127.2

- 통계개요

* 통계명: 지하철혼잡도

* 통계종류: 서울지역에서 운행하는 지하철의 혼잡도정도를 제공하는 일반 · 조사통계

* 작성목적 : 지하철 이용인원의 혼잡 정도를 파악하여 편리하고 쾌적한 지하철 환경 조성을 위한 각종 정책수립과 연구 · 분석 등을 위한 기초 자료를 제공하는 것을 목적으로 함

* 조사체계: 서울시 도시철도과

* 공표주기: 정기(매년, 12 월 기준)

* 공표범위: 지역 - 해당기관 및 자치구

내용 - 서울지역에서 운행중인(서울메트로, 도시철도공사) 지하철의 혼잡정도 등

○ 용어설명

* 혼잡도: 열차 1량당 정원대비 승차인원으로, 승차인과 좌석수가 일치할 경우를 혼잡도 34%로 산정

○ 기타

* 2004년부터 훌수년도에만 혼잡도조사 실시

○ 출처: 서울시 도시철도과

3.3.2 서울교통공사 노선별 지하철역 정보

서울교통공사_노선별 지하철역 정보

전철역코드	전철역명	전철명명(영문)	호선	외부코드	전철명명(중문)	전철명명(일문)
1722	서정리	Seojeong-ri	01호선	P163	西井里	ソジョンニ
1724	평택	Pyeongtaek	01호선	P165	平澤	ピョンテク
1701	구로	Guro	01호선	141	九老	クロ
1002	남영	Namyeong	01호선	134	南營	ナミヨン
0159	동묘앞	Dongmyo	01호선	127	东廟	トンミョアブ
1906	의정부	Uijeongbu	01호선	110	议政府	ウイジョンブ
1905	회룡	Hoeryong	01호선	111	回龙	フェリヨン
1902	도봉	Dobong	01호선	114	道峰	トボン
1801	개봉	Gaepong	01호선	143	开峰	ケボン
1918	전곡	Jeongok	01호선	100-2	全谷	チョンゴク
1901	방학	Banghak	01호선	115	放鹤	バンハク
1916	소요산	Soyasan	01호선	100	逍遙山	ソヨサン
1913	동두천중앙	Dongducheon jun... ...guk	01호선	103	东豆川中央	トンドゥチョン·チュンアン
1816	간석	Ganseok	01호선	155	间石	カンソク
1814	소사	Sosa	01호선	147	素砂	ソサ
1812	인천	Incheon	01호선	161	仁川	インチョン
1805	송내	Songnae	01호선	150	松内	ソンネ
1712	화서	Hwaseo	01호선	P154	华西	ファソ

- 서울교통공사에서 제공하는 1~8 호선, 9 호선 2~3 단계(언주~중앙보훈병원) 노선별 지하철역을 제공하는 서비스

3.3.3 서울교통공사 지하철혼잡도 정보

서울교통공사_지하철혼잡도 정보

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	
1	연번	요일구분	호선	역번호	출발역	상하구분	5시30분	6시30분	7시30분	8시30분	9시30분	10시30분	11시30분	12시30분	13시30분	14시30분	15시30분	16시30분	17시30분	18시30분	19시30분	20시30분	21시30분	
2	1	평일	1	158	청량리	상선	7.2	6.9	4.5	8.3	10.4	18.6	14.6	12.4	16	15.9	17.7	17.2	14.7	17.2	17.2	22.3	18.1	18.4
3	2	평일	1	157	제기동	상선	7.6	8.7	6.5	8.7	12.9	21.4	18.1	16.7	24.6	15.1	21.6	22.4	16.1	19.8	20.6	25.7	22.2	21.5
4	3	평일	1	156	신사동	상선	6.7	11.2	7.2	9.6	14.8	23.3	20.5	20	28.6	19.3	26.2	25.2	19.7	23.6	26.2	31.3	27	25.9
5	4	평일	1	159	동대문	상선	6.3	11.8	7.4	12.2	17.7	27	22.3	20	24.4	21.9	26.4	29	20.4	25	26.5	32.1	26.3	26.9
6	5	평일	1	158	용산	상선	7.4	11.7	8.3	14	21.7	36.3	24	27.2	26.5	23	29	32.6	30.1	33.6	35.2	34.6	32.5	32.5
7	6	평일	1	154	동대문	마천	5.1	11.2	8.1	14.6	23.1	32.5	22.5	20.9	27	29.7	30.3	24.4	30.2	33.4	34.6	32.5	32.5	32.5
8	7	평일	1	153	종로3가	상선	5.6	12.3	10.7	19.4	29.7	42.6	30.9	34	36.7	32.7	37.9	32.5	28.9	30.5	42.6	33.8	30.4	34.2
9	8	평일	1	152	종각	상선	6.4	11.4	10.4	19.6	29	37.7	30.8	34.2	37.4	30.2	34.4	35.1	26.7	28.8	36	32.8	28.8	31.1
10	9	평일	1	151	시청	상선	7	13.9	17.1	30.6	49.4	62.3	56.6	53	49	31.9	43.1	36	29.3	30.7	40.2	34.8	27.5	31
11	10	평일	1	150	서대문	상선	7.3	20.6	19.8	37.9	65.9	73.1	67.7	58.2	35.8	46.5	37.1	31.2	32.3	42.6	36.3	29	30.8	
12	11	평일	1	158	청량리	마천	20.3	18.3	27.4	43.8	67.4	64.6	61.1	50.1	34.2	32.3	25.5	19.1	26.5	24.9	24.7	25	26.3	18.7
13	12	평일	1	157	제기동	마천	20.6	19.3	28.4	44.1	69.1	71.7	64	51.2	35.1	32.3	26.3	29.1	24	28.1	30.9	23.5	30.5	22.2
14	13	평일	1	156	신사동	마천	21.6	20.5	32.1	42.4	63.2	77.1	66.1	57.1	37	33.1	26.8	27.8	26.2	37.1	32.2	32.4	22.2	
15	14	평일	1	159	동대문	마천	14.4	17.3	22.7	43.9	85.6	74.6	55.8	45.4	36.3	36.3	29.6	27.1	31.4	35.6	30.7	37.9	34.5	
16	15	평일	1	155	동대문	마천	13.5	17.6	22	43.8	70.6	80	83.9	70.6	40.8	37.1	29.9	29.4	26.8	31.1	35.3	30.7	38.3	26
17	16	평일	1	154	종로3가	마천	13	16.3	21.2	42.1	65	75.7	77.4	66.2	37.9	35.1	29.1	28.7	27.7	35.9	33.1	30.8	40.3	31.2
18	17	평일	1	153	종로3가	마천	11.8	11.4	20.6	34.8	60.5	75.3	56.1	35.3	29	29.9	22.8	23.4	25.6	27.7	28.9	27.1	35.7	27.7
19	18	평일	1	152	종각	마천	9.8	10.6	17.3	28.4	45.4	50.9	44.9	38.7	26.4	24.9	19.5	21.3	24.2	29	27	27.3	30	28.8
20	19	평일	1	151	시청	마천	9.2	9.5	16.8	22.5	36.8	35.9	26.8	30.4	21.7	22.6	18.2	20.4	21.8	28.6	25	30.2	33.6	29.3
21	20	평일	1	150	서대문	마천	10.9	15.9	15.9	24.1	38.1	45.6	24.4	29.5	21.6	21.6	22.9	29.6	26.6	31.1	35.4	33.6	33.6	
22	21	평일	1	85	목동	마천	8.5	16.1	12.7	38.6	53.6	35	28.1	21.6	28.8	28.4	22.7	31.1	30.2	28.6	32	32	32	
23	22	평일	2	209	한강대	마천	10.7	12.8	13.3	37.9	40.4	52.6	32.6	26.1	23	29.1	24.7	25	31.6	32.2	28.6	35		
24	23	평일	2	208	동작대	마천	14.3	22.3	21	57.2	64.6	84.3	60.4	41.4	39	32.1	32.3	41.2	33.9	32.5	43.4	40.5	36.7	43.8
25	24	평일	2	207	상암동	마천	16.2	23.6	22.4	61.4	69	89.1	67.5	43.1	40.9	33.3	33.2	42.5	34.5	33.2	44.2	41.3	36.8	44.3
26	25	평일	2	206	신사동	마천	18.4	26.2	21.5	62.2	76.1	93	71.8	45.4	40.7	33.4	37.7	42.4	33.4	34.8	42.2	44.4	35.9	43
27	26	평일	2	205	동작구역	마천	14.8	30.6	22	61.2	82.6	94	77.2	45.2	43.1	31.5	29.9	37.1	33.3	32.3	39.7	38.6	31.1	
28	27	평일	2	204	풀마로4	마천	14.9	27.9	22	48	75.4	84.8	76.7	42.3	39.9	31.6	26.9	35.2	32.6	31.3	36	35.7	32.5	34.5
29	28	평일	2	203	동작구역	마천	14.6	25.2	19.8	38.1	50.9	58.7	57	47.1	33	28.1	29.9	22.9	37.7	33.1	31.8	34.5	37.5	
30	29	평일	2	202	마포구	마천	21.3	19.8	19.8	38.7	57	47.1	33	28.1	29.9	22.9	37.7	33.1	31.8	34.5	37.7			
31	30	평일	2	98	시청	마천	17.9	13	26.5	41.7	43.1	54.2	37.5	26.3	29.1	23.1	33.8	35.8	30.7	36.4	40.9	35.7	36.8	
32	31	평일	2	243	충로	마천	10.2	17.3	12.1	23.2	35.6	39	31.2	25.5	25.4	25.5	23.3	31.9	35.1	30.4	35.2	39.7	34.6	36
33	32	평일	2	242	아현	마천	12	16.2	12.6	22.8	36.7	39.7	33.3	25.9	25.9	22.4	30.6	37.1	30.3	34.6	39.6	35.3	34.9	
34	33	평일	2	241	이촌	마천	13	16.3	12.9	23.3	34.9	39.6	31.2	22	23.1	23.1	19.5	29	34.4	27.3	34.3	33.3	35.9	
35	34	평일	2	240	신촌(재래)	마천	14.3	16.1	13.9	21.3	35.1	37.7	28.6	19.1	19.9	19.7	19.5	27	31.7	27.3	32.9	38.1	33.3	37.2
36	35	평일	2	239	신촌(구입)	마천	11.2	21.4	18.1	26.1	40.5	53.5	33.1	22.7	17.5	16.6	15.9	24.7	24	27.2	33.1	32.6	33.1	33.1
37	36	평일	2	238	한강대	마천	12.5	32.2	29	35.6	44.9	51.8	31.6	22.1	17.7	26.5	27.1	27.1	29.6	30.6	31	34		
38	37	평일	2	187	문래	마천	18.7	30.5	30.9	35.9	74.9	77.1	50.1	32.5	22.1	19.7	28.8	28.4	27.7	26.3	34.6	30.6	30.7	
39	38	평일	2	236	영등포구	마천	22.6	34.5	37.7	42.7	78.4	82.8	55.6	36.9	26	22.3	22.3	24.6	31.2	30	27.6	35.8	31.3	32.6
40	39	평일	2	235	문래	마천	24.1	35.8	39.6	44.3	76.9	76.8	48.9	35	23.7	25.5	21.9	24.5	31	30.5	27.8	36.2	31.3	33.1
41	40	평일	2	234	신도림	마천	24.1	27.4	33.1	44.7	50.2	68.4	68.6	50.3	32.7	30.2	24.5	34.5	32.5	29.9	38.6	33.9	29.7	
42	41	평일	2	233	마포구	마천	31.2	28.1	39.4	49.4	58.2	71.1	76.9	64.1	42.3	42.9	33.2	31.4	34.1	32.1	30.5	39.2	33.1	28.8
43	42	평일	2	232	구로디지털	마천	38.9	31.3	40.4	52.3	57.8	63.5	66.3	61.1	36.9	42.4	32.9	29.7	32.4	32.2	28	36.7	30.5	26.2
44	43	평일	2	231	신내방	마천	48.1	36.2	41.6	55.6	65.3	69.9	72.9	71.9	39.2	43.7	38.1	30.8	32.8	33.5	28.5	37	32	26.3
45	44	평일	2	316	서대문	마천	42.4	44.6	41.1	51.1	64.7	67.2	66.6	66.6	41.0	48.0	44.4	41.1	41.1	41.1	41.1	41.1	41.1	

- 서울교통공사 1~8 호선 30 분 단위 평균 혼잡도로 30 분간 지나는 열차들의 평균 혼잡도(정원대비 승차인원으로, 승차인과 좌석수가 일치할 경우를 혼잡도 34%로 산정, 단위: %).
- 서울교통공사 혼잡도 데이터는 요일구분(평일, 토요일, 일요일), 호선, 역번호, 역명, 상하선구분, 30 분단위 별 혼잡도 데이터로 구성 (2024년부터 분기별 제공됩니다.)

3.3.4 진입 역 기준 칸 혼잡도

TMAP 대중교통 API

칸 혼잡도 결과

검색한 역으로 진입하는 대표 열차의 칸별 혼잡도입니다.



3.3.5 지하철 혼잡도 안내 구성 내용

- 혼잡도 정의 및 기준 설명
 - 혼잡도가 무엇인지 정의
 - 혼잡도 산정 기준: 정원 대비 탑승 인원의 비율
 - 혼잡도 측정 단위: 칸 단위(1량), 또는 차량 전체
- 혼잡도 단계 및 구간 설명
 - 혼잡도 단계 및 구간 설명
 - 혼잡도 등급 (예: 파랑/초록/노랑/빨강 4 단계)
 - 각 단계별 혼잡도 비율(%) 및 예상 탑승 인원 수
 - 단계별 체감 상황 예시 (예: "스마트폰 사용 가능", "승객 간 밀착 심함")

여유	파랑	0~34%	~54 명	여유롭고 앉을 수 있음
보통	초록	35~99%	~159 명	보통, 입석 있으나 편함
주의	노랑	100~149%	~239 명	혼잡, 밀착 시작
혼잡	빨강	150% 이상	240 명↑	매우 혼잡, 이동 어려움

(위와 동일)

- 실시간 혼잡도 확인 방법 안내
 - 역 전광판, 열차 내 디스플레이 안내 활용 방법
- 혼잡 시간대 안내(시간별 평균 혼잡도)
 - 주요 노선의 출퇴근 시간대(예: 79 시, 1719 시) 혼잡 현황
 - 노선별/역별 평균 혼잡도 데이터 시각화 (그래프, 히트맵 등)
 - 예: 2 호선 강남역 → 평일 오전 8 시대 = 170% 이상
- 이용자 행동 유도 및 팁
 - 덜 혼잡한 칸(예: 맨 앞/뒤칸) 이용 권장
 - 피크 시간대 이용 자제 권고
 - 안전을 위한 대기 위치 조정 안내
- 혼잡도 측정 기술 안내(선택적 포함)
 - 차량 중량센서, CCTV AI 분석, Wi-Fi 데이터 활용 등
 - 측정 방식의 신뢰성과 정확도 설명
- Ex) 예시 화면 구성(홈페이지 또는 앱)
 - 노선도 상 각 칸별 색상 표시
 - 혼잡도 % 및 색상별 설명
 - 시간대별 평균 혼잡도 차트
 - 혼잡 칸 회피를 위한 탑승 위치 추천
- 사용자의 지하철 이용 패턴
 - 시간, 장소 기반 패턴
 - 출퇴근 시간대 반복: 매일 오전 8 시경 A 역에서 승차, 오후 6 시경 B 역에서 하차
 - 특정 역 반복 이용: 주로 A 역 ↔ B 역만 왕복
 - 다중 환승 패턴: C 역에서 항상 환승 후 특정 노선 이용
- 예시 흐름 (혼잡도 표시 기능 기준)
 - 사용자가 역 선택 → 백엔드 서버로 요청
 - 서버에서 혼잡도 API 연동 or DB 조회 → JSON 반환
 - 앱은 Retrofit으로 데이터 수신 → ViewModel에서 LiveData 변환
 - UI는 Compose 또는 XML로 색상 표시 (파랑/초록/노랑/빨강)
 - 특정 기준 초과 시 → FCM으로 사용자에게 푸시 알림

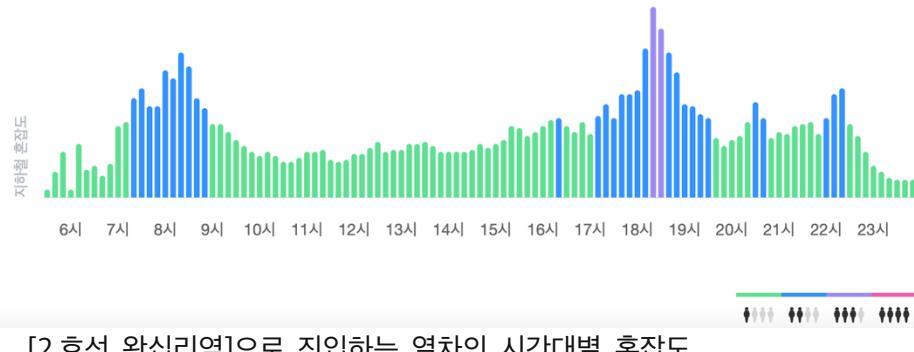
3.3.5.1 열차의 시간대별 혼잡도

3.3.5.1.1 지하철 열차 혼잡도

[2호선 왕십리역]으로 진입하는 열차의 시간대별 혼잡도

2025.05.25. ~ 2025.08.24., 상왕십리역 방면, 화요일

열차방향 바꾸기 ↪



▪ [2 호선 왕십리역]으로 진입하는 열차의 시간대별 혼잡도

3.3.6 실시간 칸별 혼잡도 예측

- 혼잡도를 실시간 분석해 비교적 여유 있는 칸을 추천
- 서울교통공사 API 연동을 통한 정확한 정보 제공

3.3.6.1 혼잡도 단계 구분 (서울교통공사 기준)

단계	색상	혼잡도(%)	예상 탑승 인원 (1량=160 명)	체감 설명
여유	파랑	0% ~ 34%	0 ~ 약 54 명	좌석 여유 많고 서 있는 사람 거의 없음
보통	초록	35% ~ 99%	55 명 ~ 159 명	입석 승객 있지만 이동 가능
주의	노랑	100% ~ 149%	160 명 ~ 239 명	입석 많고 승객 간 접촉 발생
혼잡	빨강	150% 이상	240 명 이상	승객 밀착 심함, 이동·스마트폰 사용 어려움

1) 칸별 혼잡도 시스템 도입 배경

서울시는 승객의 탑승 환경을 개선하기 위해 2호선 신형 전동차에 ‘객실 혼잡도 안내 시스템’을 도입하였다. 이 시스템은 차량에 설치된 하중 센서를 통해 각 칸의 무게를 실시간 측정하여 탑승 인원을 추정하고, 이를 기반으로 혼잡도를 제공한다.

2) 혼잡도 산정 방식 및 안내

칸별 하중 데이터를 정원 대비 비율로 계산해 여유(79% 이하), 보통(80~129%), 혼잡(130% 이상)의 세 단계로 구분된다. 산정된 혼잡도는 열차 내부 LCD 화면에 표시되며, 향후 앱이나 외부 플랫폼에서도 확인할 수 있도록 확장될 가능성이 있다.

3) 안전 문제와 시스템의 의미

2016~2018년 사이 도시철도 범죄 통계를 보면, 2호선의 사건 비율이 높고, 특히 성범죄 비중이 큰 것으로 나타났다. 이러한 배경에서 칸별 혼잡도 안내 시스템은 단순 편의성을 넘어 승객 안전 확보 측면에서도 중요한 역할을 한다.

3.4 데이터 분석 기반 예측 시스템(Data Analysis-based Prediction System)

- 현대의 대중교통 시스템은 승객 수요 증가와 교통 혼잡 문제로 인해 효율적인 운영이 점차 중요해지고 있다.
- 특히 지하철은 대표적인 대중교통 수단으로, 특정 시간대의 혼잡도(Congestion Level)와 좌석 점유율(Seat Occupancy Rate)에 대한 정확한 예측은 승객의 편의성과 교통 관리의 효율성을 동시에 높일 수 있는 핵심 요소이다.
- 본 모듈은 공공데이터(Public Data)를 기반으로 한 예측 시스템을 구축하여, 단순한 과거 데이터 분석에 머무르지 않고 “미래의 혼잡도를 예측하고 이를 서비스에 활용”하는 것을 목표로 한다.

3.5 데이터 수집 및 전처리

3.5.1 데이터 수집 (Data Collection)

- 본 모듈에서는 공공데이터 포털(Open Data Portal) 및 서울교통공사와 같은 기관에서 제공하는 자료를 활용한다.
 - 승객 수 데이터 (Passenger Count Data): 시간대별 승하차 인원 및 구간별 탑승률.
 - 열차 위치 데이터 (Train Location Data): 열차의 현재 위치, 운행 간격, 정차 시간 등.

3.5.2 데이터 전처리 (Data Preprocessing)

- 수집된 데이터는 원본 그대로 사용할 수 없으므로 전처리 과정을 거친다.
 - 결측치(Missing Value) 처리
 - 이상치(Outlier) 제거
 - 시간 단위 정규화(Time Normalization)
 - 학습용(Training)과 검증용(Validation), 테스트용(Test) 데이터셋 분리
- 데이터의 품질(Data Quality)은 모델 성능에 직접적인 영향을 미치기 때문에 전처리는 필수적인 단계이다.

3.6 예측 모델 개발(Prediction Model Development)

3.6.1 머신러닝 및 딥러닝 기법 적용(Machine Learning & Deep Learning Application)

- 예측 모델 개발 과정에서는 여러 알고리즘을 적용하고 비교한다.

- 랜덤 포레스트(Random Forest, RF): 다수의 의사결정나무(Decision Trees)를 조합하는 앙상블(Ensemble) 학습 기법으로, 변수 해석력이 뛰어나며 안정적인 예측이 가능하다.
- 장단기 메모리 신경망(Long Short-Term Memory, LSTM): 시계열 데이터(Time Series Data) 분석에 특화된 딥러닝 모델로, 시간적 패턴과 연속성을 학습할 수 있어 혼잡도 변화 예측에 적합하다.

3.6.2 최적 모델 선정 (Best Model Selection)

- 여러 모델을 학습 및 검증하여, 성능 지표를 기반으로 최적의 모델을 선정한다.
- 모델 선정 기준은 예측 정확도(Accuracy)뿐만 아니라, 서비스 적용 가능성(Practical Applicability)과 처리 속도(Processing Speed)도 포함된다.

3.7 모델 학습 및 성능 평가(Model Training and Evaluation)

3.7.1 학습 과정(Training Process)

- 데이터셋을 훈련용과 검증용으로 분할하여 모델을 학습시킨다. 이후 테스트 데이터셋으로 성능을 검증한다.

3.7.2 성능 평가 지표 (Evaluation Metrics)

- 모델의 성능은 다음과 같은 지표를 통해 정량적으로 측정한다.
 - MAE (Mean Absolute Error, 평균 절대 오차): 예측값과 실제값의 절대적 차이의 평균.
 - RMSE (Root Mean Square Error, 평균 제곱근 오차): 큰 오차에 더 민감하게 반응하는 지표로, 모델의 안정성을 평가할 수 있다.
- 이를 통해 각 모델의 장단점을 파악하고, 본 프로젝트의 목적에 가장 적합한 모델을 선택한다.

3.8 데이터베이스 연동 (Database Integration)

- 본 모듈의 예측 결과는 데이터베이스(Database)에 저장되어, 다른 모듈에서 활용될 수 있도록 한다.
 - 예측 테이블(Prediction Table): 혼잡도 및 좌석 점유율 예측값을 구조적으로 저장하는 데이터베이스 테이블.
 - 연동 목적: 추천 시스템(Recommendation System), 알림 시스템(Notification System) 등 다른 모듈이 실시간으로 데이터를 불러와 서비스에 적용할 수 있도록 지원한다.
- 이를 통해 예측 모델이 단순히 분석 결과에 그치지 않고, 실제 서비스와 연계되는 엔드투엔드(End-to-End) 시스템을 구현할 수 있다.

3.9 결론 및 기대 효과

- 본 모듈은 단순 데이터 분석 단계를 넘어, 데이터 기반 의사결정(Data-driven Decision Making)을 실현하는 핵심 요소로 기능한다.
- 미래의 혼잡도 예측 결과는 사용자(User)에게 보다 효율적인 탑승 시간 및 객차 선택을 제공할 수 있으며, 이는 곧 앱(App) 전체 서비스 품질(Service Quality) 향상으로 이어진다.

- 또한, 데이터베이스 연동을 통해 예측 결과가 다른 모듈에 제공됨으로써, 개인 맞춤형 추천(Personalized Recommendation), 하차 알림(Exit Notification) 등의 고도화된 서비스가 가능해진다.
- 궁극적으로 본 모듈은 사용자 편의성(User Convenience) 증대와 교통 혼잡 완화(Social Congestion Relief)라는 사회적 가치 창출에 기여할 것으로 기대된다.

4 설계

4.1 Architecture

1. System Architecture

가) 개발 환경

항목	내용
IDE	Android Studio (Giraffe 이상 추천)
언어	Java
OS	Android (API Level 26 이상 권장)
빌드 시스템	Gradle
SDK	Android SDK
버전관리	GitHub
협업 툴	GitHub

나) Framework

목적	기술 스택
UI	안드로이드 탑재
네트워킹	Http, Json
위치 기반 서비스	서울교통공사 API
알림 기능	Firebase Cloud Messaging (FCM), WorkManager
백엔드 통신	REST API
지도 및 위치 시각화	Google Maps API
데이터 저장	Android Studio (Room), Firebase
혼잡도 분석 (선택)	열린공공데이터, 자체 알고리즘
좌석 및 역 지정	자체 개발 시스템

2. Application Software Architecture

가) 아키텍처 패턴: MVVM(Model-View-ViewModel)

- ① View
 - Android UI (Activity, Fragment, Compose)
 - 사용자 입력 수신 및 ViewModel과 바인딩
- ② ViewModel UI
 - 로직 처리
 - LiveData/StateFlow로 View에 상태 전달
 - Repository 호출

③ Model

- 데이터 모델 정의 (User, SubwayStation, CongestionData 등)
- Repository 가 API/DB 와 통신

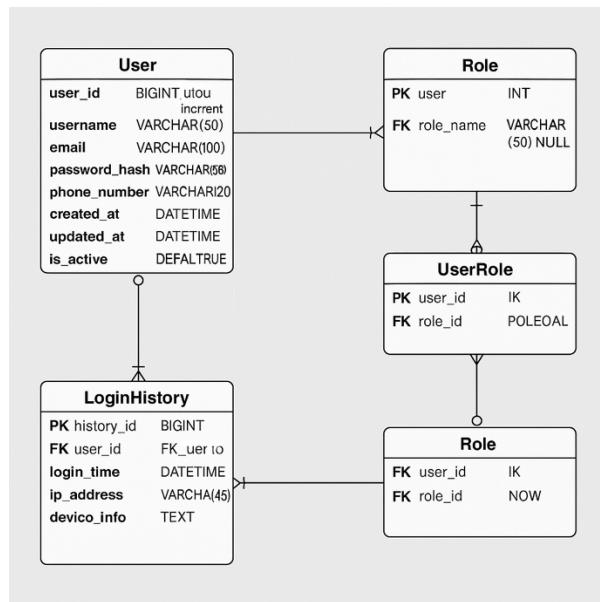
④ Repository

- 서버 API 호출 (Retrofit)
- 로컬 데이터 처리 (Room)
- ViewModel 에 데이터 전달

⑤ DataSource

- RemoteDataSource: 서버 API 통신
- LocalDataSource: Room 또는 SharedPreferences

4.2 Data 설계: ER diagram



4.3 스키마

4.4 데이터베이스 상세 설계

4.4.1 회원 데이터베이스(User DB)

- 사용자 정보를 저장하는 데이터베이스

필드명	타입	설명
user_id	INT AUTO_INCREMENT PRIMARY KEY	PK, 자동 증가 ID
username	VARCHAR(50)	사용자 이름
email	VARCHAR(100)	이메일 주소
password	VARCHAR(255)	암호화된 비밀번호
phone_number	VARCHAR(20)	전화번호 (선택)
created_at	DATETIME DEFAULT CURRENT_TIMESTAMP	가입일 (타임스탬프)

CRUD

-- CREATE

```
INSERT INTO Users (user_id, username, email, password, phone_number, created_at)
VALUES (1, 'jeong', 'jeong@email.com', 'pw1234', '01012345678', CURRENT_TIMESTAMP);
```

-- READ

```
SELECT * FROM Users WHERE user_id = 1;
```

-- UPDATE

```
UPDATE Users
SET username = 'newjeong', phone_number = '01098765432'
WHERE user_id = 1;
```

-- DELETE

```
DELETE FROM Users WHERE user_id = 1;
```

4.4.2 모듈 데이터베이스(Module DB)

- 하차 알림, 좌석 지정, 알림 설정 등 사용자 맞춤 기능 설정 데이터를 저장

필드명	타입	설명
module_id	INT AUTO_INCREMENT PRIMARY KEY	PK, 자동 증가 ID
user_id	INTEGER	FK → User.user_id
seat_number	VARCHAR(10)	사용자가 지정한 좌석 번호 (선택)
destination	VARCHAR(100)	설정된 하차역
alarm_enabled	BOOLEAN DEFAULT TRUE	하차 알림 설정 여부
created_at	DATETIME DEFAULT CURRENT_TIMESTAMP	설정한 날짜

CRUD

-- CREATE

```
INSERT INTO User_Module (module_id, user_id, seat_number, destination, alarm_enabled)
VALUES (101, 1, '3A', '강남역', 1);
```

-- READ

```
SELECT * FROM User_Module WHERE user_id = 1;
```

-- UPDATE

```
UPDATE User_Module
SET seat_number = '4B', destination = '신림역', alarm_enabled = 0
WHERE module_id = 101;
```

-- DELETE

```
DELETE FROM User_Module WHERE module_id = 101;
```

4.4.3 지하철 호선 데이터베이스(Subway DB)

역, 호선, 혼잡도 등 지하철 관련 정보를 저장

(1) 지하철 노선 정보 테이블

필드명	타입	설명
line_id	INT AUTO_INCREMENT PRIMARY KEY	PK, 자동 증가
line_name	VARCHAR(50)	예: 1호선, 2호선 등
color_code	VARCHAR(10)	지하철 호선 색상 (선택)

CRUD

-- CREATE

```
INSERT INTO Subway_Line (line_id, line_name, color_code)
VALUES (2, '2호선', '#00A862');
```

-- READ

```
SELECT * FROM Subway_Line WHERE line_id = 2;
```

-- UPDATE

```
UPDATE Subway_Line
SET line_name = '2호선 (내선)', color_code = '#00B070'
WHERE line_id = 2;
```

-- DELETE

```
DELETE FROM Subway_Line WHERE line_id = 2;
```

(2) 지하철 역 정보 테이블

필드명	타입	설명
station_id	INT AUTO_INCREMENT PRIMARY KEY	PK
station_name	VARCHAR(100)	역 이름
line_id	INTEGER	FK → SubwayLine.line_id
congestion	VARCHAR(100)	혼잡도 (0~100 퍼센트)

-- CREATE

```
INSERT INTO Subway_Station (station_id, station_name, line_id, congestion)
VALUES (101, '강남역', 2, '혼잡');
```

-- READ

```
SELECT * FROM Subway_Station WHERE station_name = '강남역';
```

-- UPDATE

```

UPDATE Subway_Station
SET congestion = '여유'
WHERE station_id = 101;

-- DELETE
DELETE FROM Subway_Station WHERE station_id = 101;

```

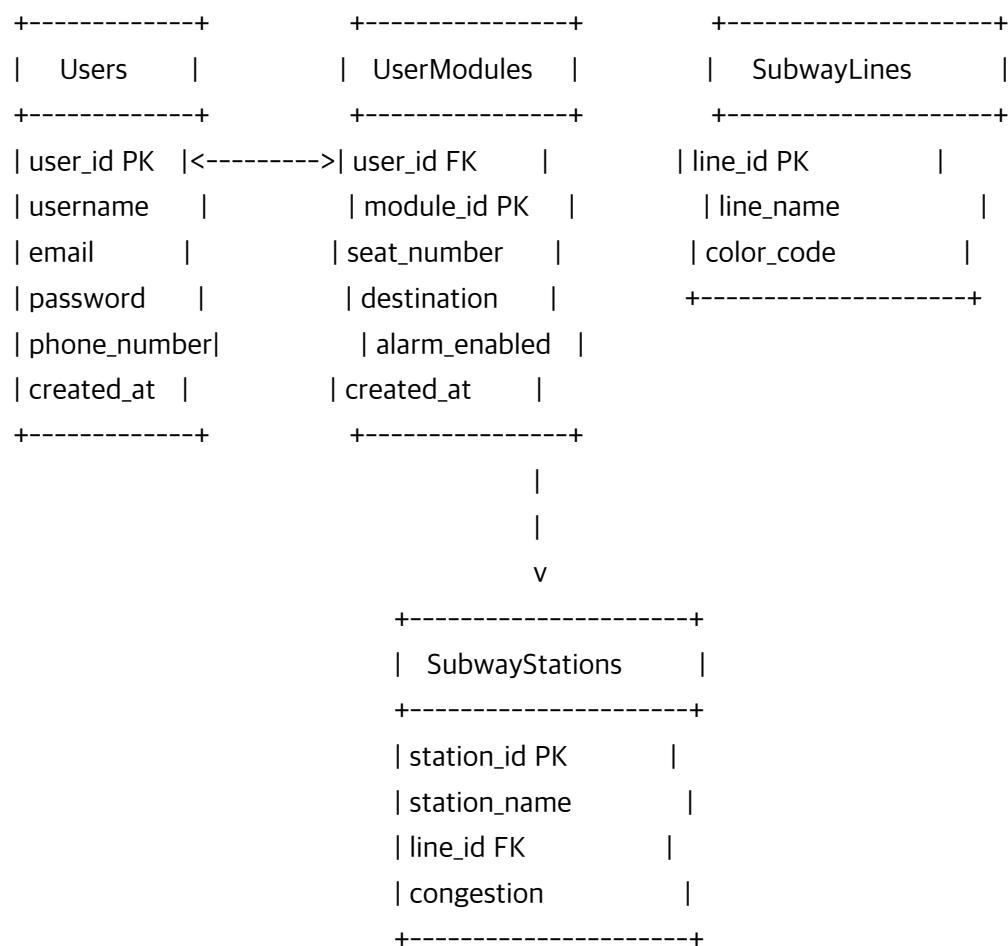
- ERD 관계 요약

- User ↔ UserModule: 1:N
- SubwayLine ↔ SubwayStation: 1:N

- 확장 아이디어

- 하차 알림 로그 테이블: 알림 기록을 남기고 싶다면 추가 가능.
- 좌석 예약 기능을 세분화하고 싶다면 SeatReservation 테이블 별도 생성.
- 실시간 혼잡도를 외부 API에서 가져오는 구조로 할 수도 있음.

- ER 다이어 그램 설명



- 관계 요약

테이블 관계	설명
Users ↔ UserModules	1:N (한 유저는 여러 모듈 설정 가능)

SubwayLines ↔ SubwayStations	1:N (한 노선에 여러 역 존재)
------------------------------	---------------------

4.5 화면 설계

4.5.1 UI 설계(현재 시간대 + 혼잡도 표시 화면)

1. 화면 이름: 혼잡도 현황 화면

2. 화면 구성

가) 상단영역

- 현재 시간 (예: 오전 8:45)
- 현재 선택한 노선 & 역 이름 (예: 2호선 강남역)

나) 중앙영역

- 칸별 혼잡도 표시 (막대 그래프 or 칸 아이콘 색상)
 - 여유, 보통, 주의, 혼잡
- 추천 칸 강조 표시 (예: "3 번째 칸이 가장 여유 있음")

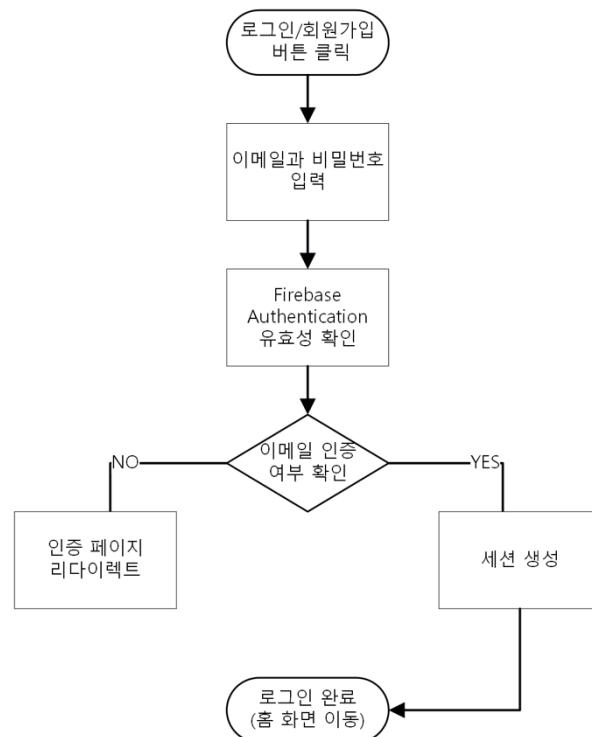
다) 하단영역

- "다른 시간대 보기" 버튼 → 예측 화면 이동
- "다른 역 선택" 버튼

4.6 LOGIC 설계

4.6.1 회원관리

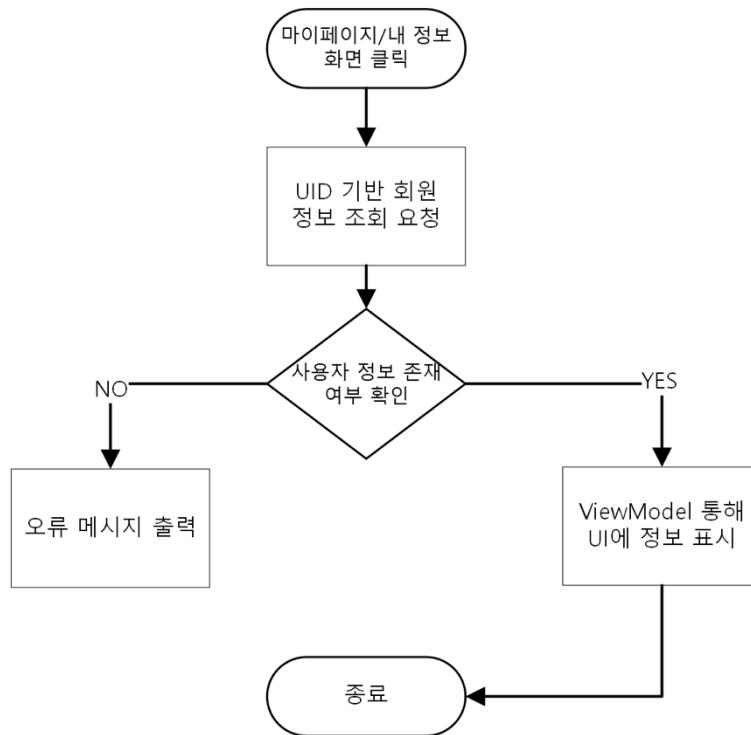
A. 로그인 및 신규 회원 등록



1. 사용자가 이메일과 비밀번호를 입력하고 로그인 또는 회원가입 버튼을 클릭한다.
2. Firebase Authentication을 통해 해당 이메일과 비밀번호의 유효성을 확인한다.
3. 회원가입의 경우, 이메일 인증 메일을 발송하고 인증 완료 시 최종 계정이 생성된다.
4. 로그인 시 이메일 인증 여부를 확인한 후 세션을 생성하고 홈 화면으로 이동한다.

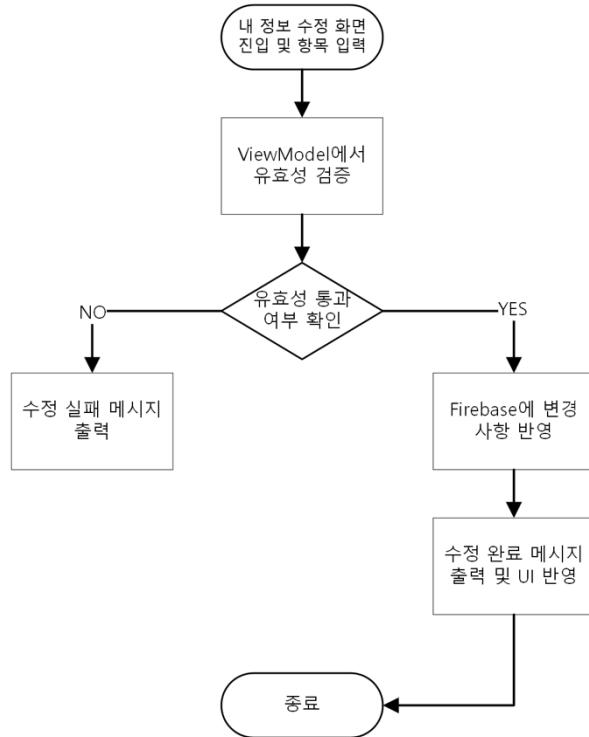
5. 인증되지 않은 이메일로 로그인 요청 시 경고 메시지를 출력한다.

B. 회원 정보 조회



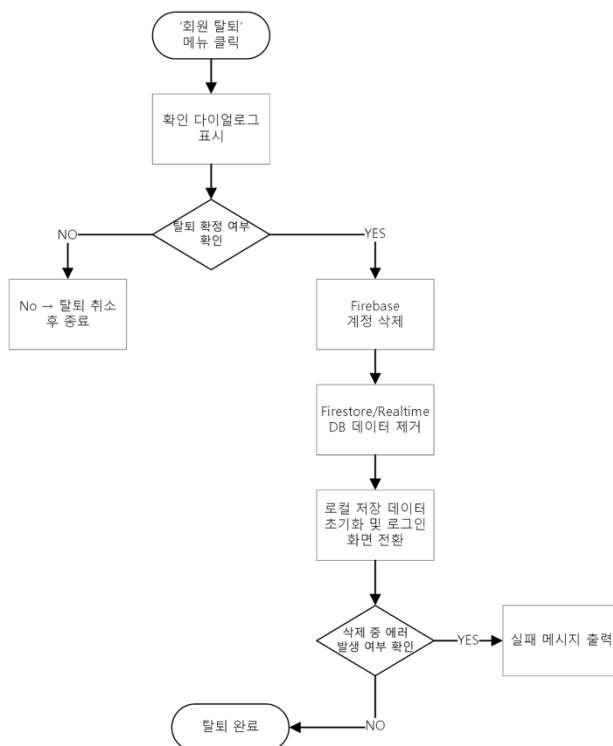
1. 사용자가 마이페이지 또는 내 정보 화면을 클릭한다.
2. 현재 로그인된 사용자의 UID를 기반으로 Firebase Firestore 또는 Realtime Database에서 회원 정보를 불러온다.
3. 조회된 정보는 ViewModel을 통해 UI에 전달되어 사용자에게 표시된다.
4. 사용자 정보가 존재하지 않을 경우 오류 메시지를 출력한다.

C. 회원 정보 수정



1. 사용자가 내 정보 수정 화면에서 닉네임, 비밀번호 등 변경할 항목을 입력한다.
2. 변경 요청이 제출되면 ViewModel에서 입력값의 유효성을 검증한다.
3. 유효한 경우, Firebase Authentication 및 Firestore에 변경 사항을 반영한다.
4. 성공 시 수정 완료 메시지를 출력하고 수정된 내용을 화면에 반영한다.
5. 오류 발생 시 수정 실패 메시지를 출력한다.

D. 회원 탈퇴



1. 사용자가 내 정보 수정 화면에서 닉네임, 비밀번호 등 변경할 항목을 입력한다.

2. 변경 요청이 제출되면 ViewModel에서 입력값의 유효성을 검증한다.
3. 유효한 경우, Firebase Authentication 및 Firestore에 변경 사항을 반영한다.
4. 성공 시 수정 완료 메시지를 출력하고 수정된 내용을 화면에 반영한다.
5. 오류 발생 시 수정 실패 메시지를 출력한다.

4.7 Interface 설계: API 데이터

4.7.1 SK OPEN API 이용절차

- 회원가입/로그인

1. 로그인 화면에서 T 아이디로 로그인을 클릭한 후 기존 T 아이디를 이용해 가입하거나, 회원가입 > T 아이디로 가입을 클릭해 T 아이디를 신규 생성하고 회원으로 가입할 수 있습니다.
- 개인 회원은 SK open API 기능 및 데이터를 개인 용도로 사용하며 별도 세금 계산서 발행이 필요하지 않은 경우 적절한 회원 유형입니다.
- 사업자 회원은 사업체에서 고객 서비스를 개발하는 용도로 SK open API 기능 및 데이터를 사용하려는 경우 적합한 회원 유형으로, 사용 요금에 대해 세금 계산서를 발행할 수 있으며, 필요한 경우 계정 관리 업무와 관련해 서비스 지원을 요청할 수 있습니다.
2. SK open API 로그인 화면에서 [회원가입] 버튼을 클릭하세요.
3. 회원가입 화면에서 [개인회원] 또는 [사업자]를 선택하여 가입을 진행하세요.

- 앱 만들기

1. 홈페이지 우측 상단의 대시포드 클릭 후 화면 좌측의 상세 메뉴 영역에서 [앱]을 클릭하세요.
2. [앱 만들기] 버튼을 클릭하세요.
3. 앱 만들기 팝업에서 앱 이름을 입력 후 확인을 클릭하세요.
4. 만들어진 앱을 확인합니다. 앱 선택 시 앱 상세 정보를 확인할 수 있습니다.

- 상품 구매

1. 홈 화면의 상품 영역에서 상품의 카테고리를 선택 후 상세정보를 확인할 상품을 클릭하세요.
 2. 좌측 상세 메뉴 영역에서 각 메뉴를 클릭해 상품 정보를 확인한 후 [API 사용 요금]을 클릭하세요.
 3. 사용할 요금제의 [사용하기]를 클릭 후 사용 신청 화면에서 상세 정보를 입력하거나 설정하세요. 약관에 동의한 후 [사용 신청하기]를 클릭하세요.
 4. 사용 신청 완료 창에서 상품 정보와 결제 정보를 확인하세요.
- 서비스 이용

- 홈 > 대시보드 > 요금 조회 > 사용상품관리 화면에서 구매하신 상품 확인이 가능 합니다.

4.8 데이터 저장 방식 및 위치 설계

4.8.1 서비스 관련 데이터

- 예: 지하철 노선, 실시간 혼잡도, 하차 알림 설정, 역 정보 등
- 주로 읽기 위주, 즉 사용자들이 조회만 많이 하고 자주 바뀌지 않거나 외부 서버에서 받아오는 데이터

4.8.2 회원 정보 관련 데이터

- 예: 로그인 정보, 사용자 하차 역 설정, 즐겨찾는 노선, 좌석 지정 기록 등
- 읽고 쓰기 모두 필요하고, 사용자마다 다른 개인화된 정보

4.8.3 각 데이터 저장 공간

- 서비스 관련 데이터 → 외부 서버(API) 또는 로컬 캐시
 - 외부 API (ex. 공공데이터포털 API)
 - 로컬 캐시는 Room(로컬 앱), Shared Preferences
 - 실시간 정보는 서버에서 받아와야 최신 상태 유지 가능
 - 자주 바뀌지 않으면 앱 안에 저장해도 됨
- 회원 정보(로그인, 즐겨찾기 등) → 로컬 DB or Firebase
 - Firebase Realtime Database / Firestore
 - Room (SQLite 기반)
 - Firebase 는 사용자별 데이터 관리에 강함
 - Room 은 기기 내부 저장용으로 간단한 구조에 적합

4.8.4 추천 시나리오별 정리

- 서비스 데이터(혼잡도, 지하철 정보)
 - 혼잡도: 외부 API에서 받아오는 게 정확하고 최신임 (예: 서울시 지하철 API)
 - 지하철 역/노선 정보: 앱 내 Room DB에 한 번 저장해두면 빠르게 불러올 수 있음
 - 처음 앱 설치 시 한 번 받아서 Room에 저장해두고, 주기적으로 업데이트
- 회원 정보(로그인, 하차 알림, 즐겨찾기 등)
 - 로그인: Firebase Authentication 쓰면 구글/이메일 로그인 쉽게 가능
 - 하차 알림 설정, 즐겨찾기: Firebase Firestore 또는 Firebase Realtime Database 추천 (사용자 ID 별로 데이터 구분하고 저장 가능)
 - Firebase 는 클라우드 기반이라 여러 기기에서 정보 동기화가 가능

4.8.5 전체 구조 요약

- 사용자 스마트폰
 - Room DB (지하철 역 정보, 하차 설정 로컬 저장)
 - Shared Preferences (단순 설정값 저장)
 - Firebase (회원 정보, 즐겨찾기, 로그인)
- 외부 서버
 - 공공 API (지하철 혼잡도, 실시간 도착 정보 등)

4.8.6 요약 정리

- 실시간 지하철 정보 → 외부 API 데이터가 계속 변함
 - 지하철 역/노선 정보 → Room (로컬 DB), 빠르게 불러오기, 오프라인 지원
 - 사용자 로그인 → Firebase Authentication, 쉬운 구현, 보안 강함
 - 즐겨찾기/하차 알림 → Firebase Firestore, 사용자별 데이터 관리 용이

4.9 앱 색상 설계



colorPrimary colorPrimaryVariant colorAccent grayLight grayDark

- colorPrimary: #0D3B66 (딥 네이비)
- colorPrimaryVariant: #14477A (세컨 네이비)
- colorAccent: #1E6091 (강조용 블루)
- grayLight: #D3D3D3(밝은 회색)
- grayDark: #696969 (어두운 회색)

5 코딩

5.1 gitHub

- <https://github.com/choisuvin/zariit.git>

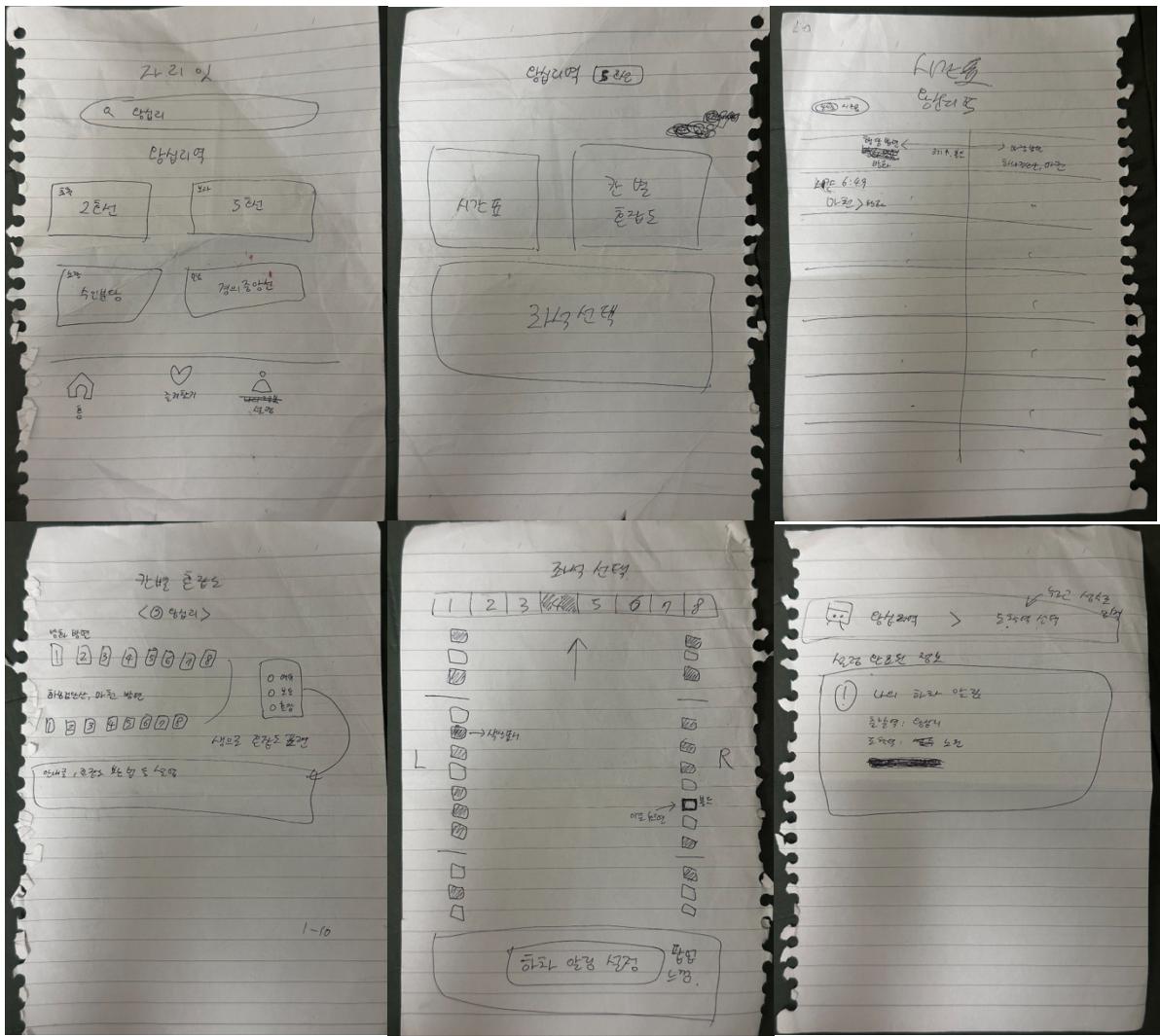
5.2 회원가입 실행화면

- https://youtube.com/shorts/7_uW4ZLGMVo?si=W_Nhs3At3JzYFM7q

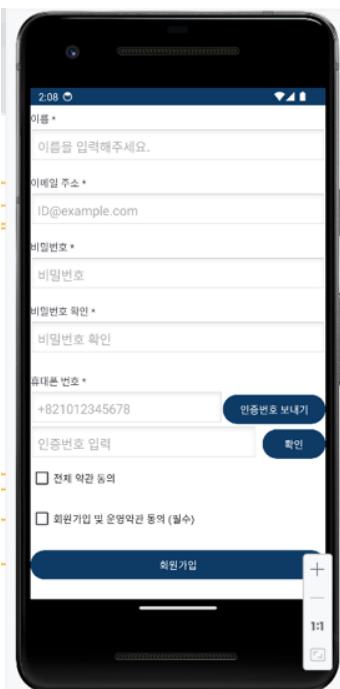
5.3 홈 화면 실행화면

- <https://youtube.com/shorts/W83iuBLOuL4?feature=share>

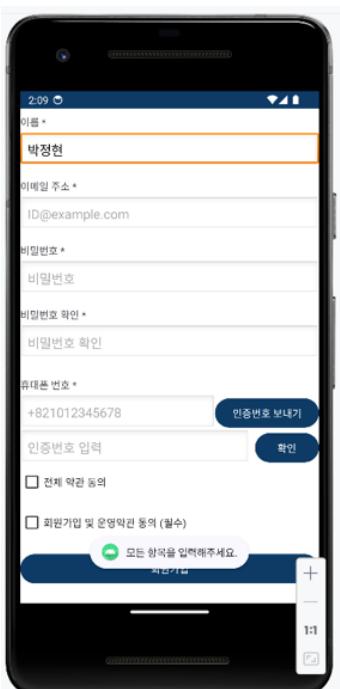
5.4 모듈 초기 스케치



5.5 회원 가입 화면

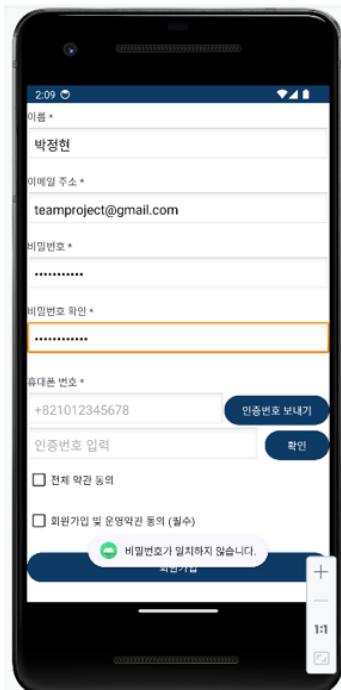


회원가입 버튼을 누르면 신규 가입 화면이 열리며, 사용자는 이름, 이메일 주소, 비밀번호, 휴대폰 번호를 입력하고 휴대폰 인증을 완료해야 한다. 또한 필수 약관 동의를 진행해야 회원가입이 완료된다.

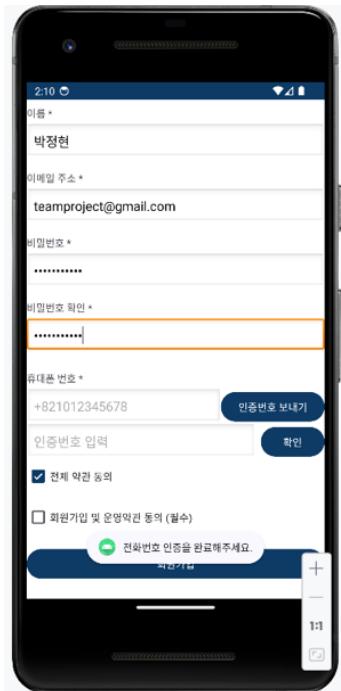


모든 입력 항목을 완료하지 않은 상태에서 회원가입을 시도할 경우, 화면 하단에 ‘모든 항목을

‘입력해주세요.’라는 안내 메시지가 표시되어 필수 정보 누락을 알린다.



비밀번호와 비밀번호 확인 값이 서로 일치하지 않을 경우, ‘비밀번호가 일치하지 않습니다.’라는 안내 메시지가 표시되어 사용자에게 오류를 알려준다.



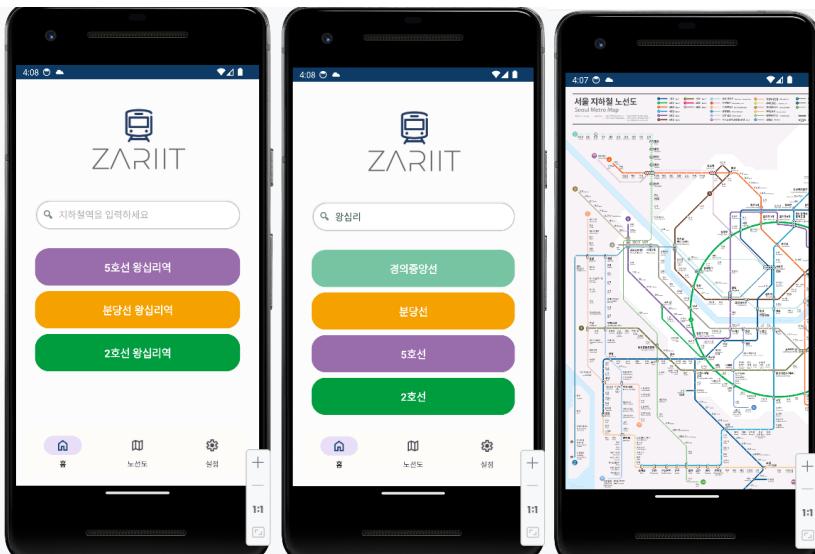
휴대폰 번호를 입력 후 인증 절차를 완료해야 하며, 인증을 진행하지 않은 상태에서 회원가입을 시도할 경우 ‘전화번호 인증을 완료해주세요.’라는 안내 메시지가 표시된다.

5.6 메인 화면



사용자는 이메일과 비밀번호를 입력하여 로그인할 수 있으며, 또는 구글 및 카카오 계정을 통한 간편 로그인 기능도 지원한다.

5.7 홈 화면

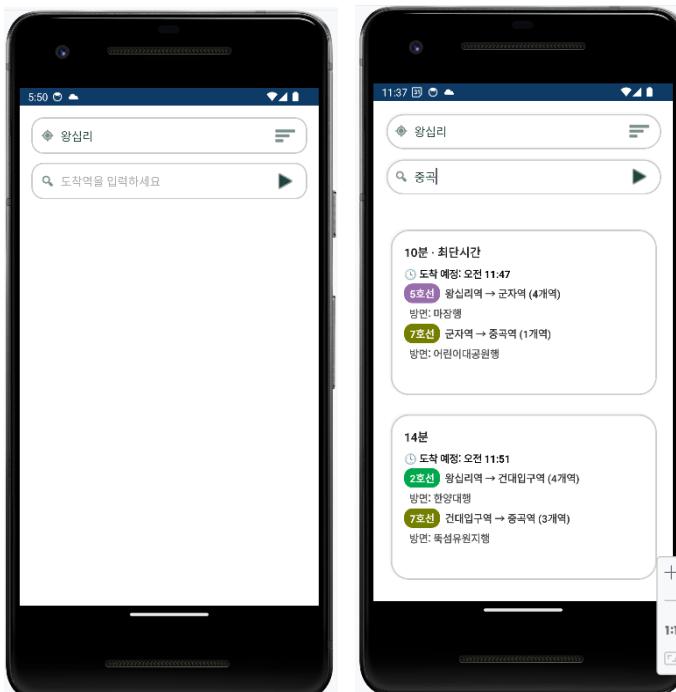


검색창에 지하철역을 입력하면 해당 역의 호선을 선택할 수 있으며, 검색창 아래에는 사용자가 자주 조회한 역과 호선 정보가 자동으로 표시된다. 또한 하단 메뉴에서 ‘노선도’를 선택할 경우 수도권 지하철 전체 노선도가 제공된다.

5.8 시간표 화면



지하철 역과 호선을 선택하면 해당 역과 호선 정보가 화면 상단에 표시되며, 그 아래에 시간표, 칸별 혼잡도, 좌석 선택 기능을 이용할 수 있는 버튼이 제공된다. 화면 하단에는 사용자의 이전 탑승 경로가 최신 순으로 최대 두 개까지 표시되며, 이를 선택하면 도착역을 다시 입력하지 않아도 이전 경로 그대로 좌석 선택 화면으로 바로 이동할 수 있다.



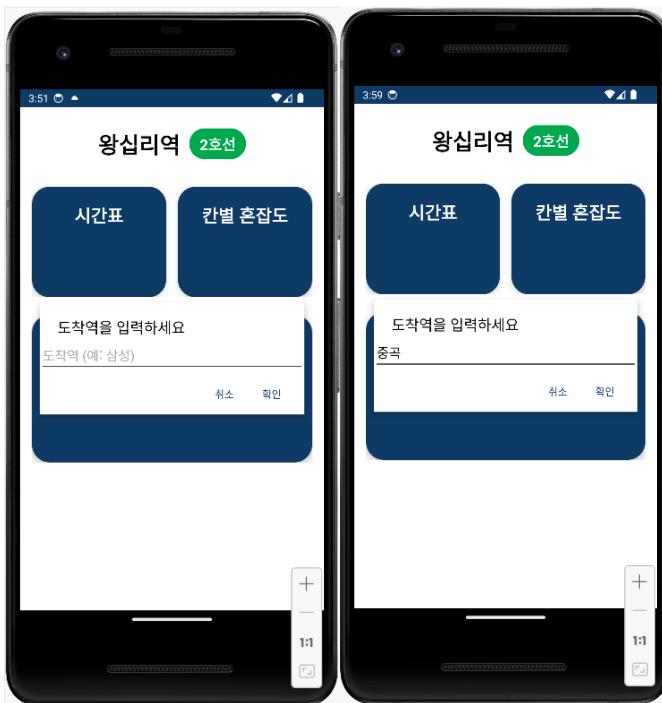
시간표: 사용자가 입력한 도착역 기준으로, 출발역부터 도착역까지 실제 소요 시간과 예상 도착 시간을 보여준다. 모든 경로는 소요 시간이 짧은 순서대로 추천되며, 초기 화면에서 설정한 출발역은 자동 입력된다. 단, 사용자는 필요 시 해당 값을 삭제 후 새로 입력할 수도 있다.

5.9 칸별 혼잡도 화면

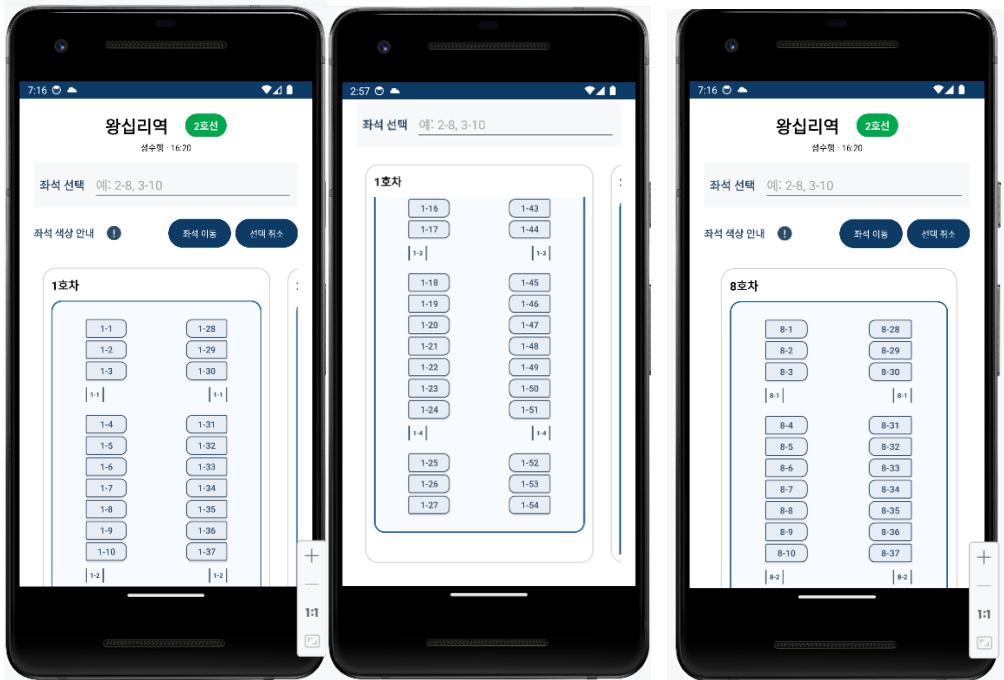


칸별 혼잡도: 상행과 하행을 구분하여, 각 칸의 혼잡 정도를 색상으로 시각화해 보여준다. 혼잡도 데이터는 실시간 SK Open API 지하철 혼잡도 API와 TMAP 대중교통 API를 기반으로 제공된다. 또한 칸별 정보가 제공되지 않는 구간은 임시로 회색 표시로 나타낸다.

5.10 좌석 선택 화면

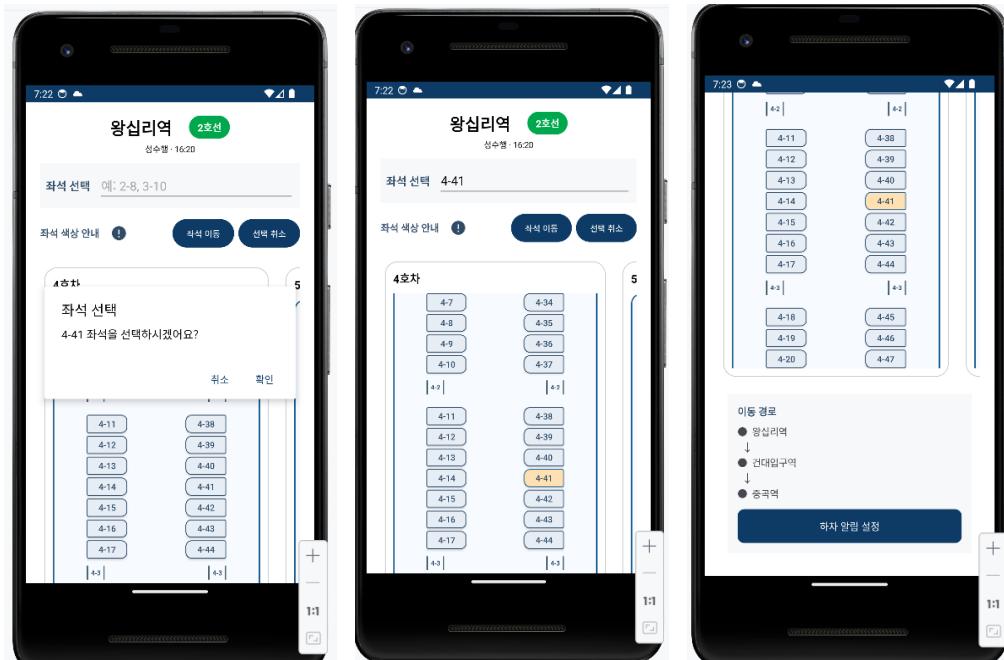


좌석 선택: 사용자가 ‘좌석 선택’ 버튼을 클릭하면 먼저 도착역 입력을 요청하는 팝업이 나타난다. 도착역을 입력한 후에는, 해당 구간의 좌석 배치와 선택 가능한 좌석 정보를 확인할 수 있는 좌석 선택 화면이 표시된다.



사용자가 도착지를 입력하면, 해당 구간의 운행 방향(상행/하행)과 함께 열차 도착 시각(예: ○시 ○분 차량)이 역 정보 하단에 표시된다.

열차는 1호차부터 8호차까지 구성되어 있으며, 각 차량의 좌석은 양쪽 3-7-7-7-3 배열로, 총 54개의 좌석으로 설정되어 있다.



사용자가 좌석을 선택하면, “좌석을 선택하시겠습니까?”라는 확인 팝업이 표시된다. ‘확인’을 누르면 선택한 좌석에 색상이 표시되며, 화면 하단에는 이동 경로(출발지 → 환승역 → 도착지)가 자동으로 나타난다.

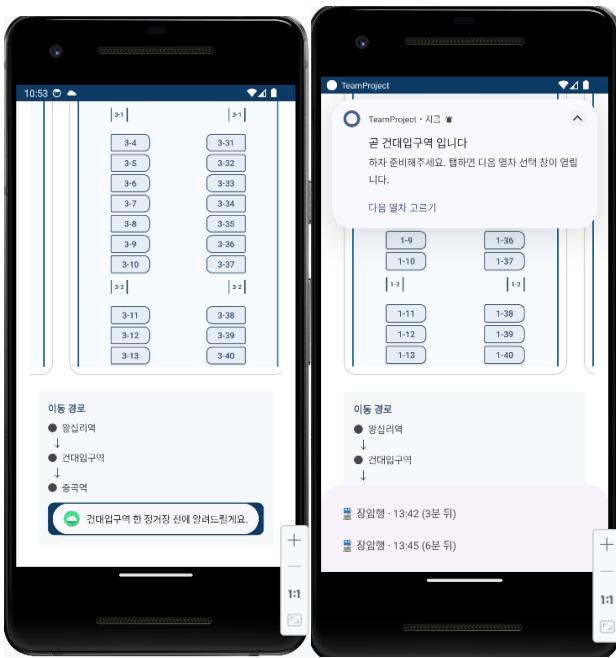


좌석 색상은 도착까지 남은 역의 수에 따라 아래와 같이 표시된다.

남은 역 수	색상	의미
10 역 이상	초록	여유
6~9 역	노랑	관심
3~5 역	주황	준비
1~2 역	빨강	임박

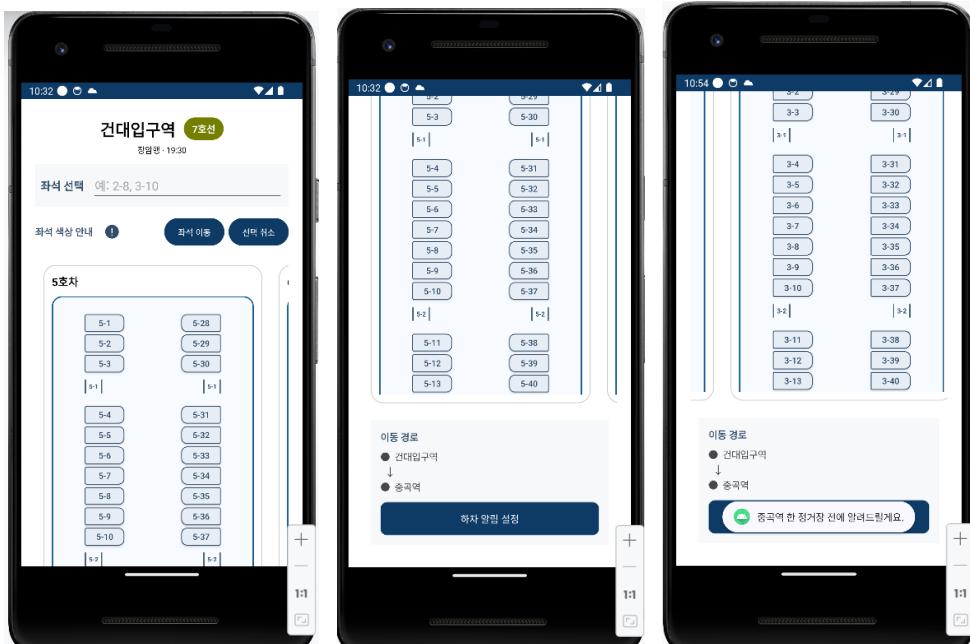
이동 경로에는 사용자가 앱 접속 시 입력한 출발역과 좌석 선택 전 입력한 도착역이 함께 표시되며, 환승이 필요한 경우 자동으로 환승역까지 경로가 설정된다.

5.11 하차 알림 설정 화면



사용자가 ‘하차 알림 설정’ 버튼을 누르면, 앱은 환승역 이전 역(전거장)에서 “곧 하차해야 합니다”라는 사전 알림을 보내준다.

또한 하차 직전에는 “곧 도착 예정입니다”라는 도착 알림이 뜨며, 화면 하단에는 사용자가 환승하려는 다음 차량의 도착 정보가 함께 표시된다.



사용자가 탑승할 차량(몇 분 뒤 도착 예정 열차)을 선택하면, 해당 차량 기준으로 환승역 이후 도착역까지의 좌석 선택 화면이 새롭게 표시된다. 이 화면에서도 이전과 동일하게 좌석을 선택할 수 있으며, 선택 후 ‘하차 알림 설정’을 누르면 해당 구간의 하차역 한 정거장 전에서 “곧 하차해야 합니다”라는 알림을 받을 수 있다.

6 테스팅

6.1 Test Case

6.1.1 테스트 개요

- 자리잇 앱의 회원 관리 기능(가입, 로그인, 탈퇴 등)이 정상적으로 작동하는지 확인하고, 사용자 인증 및 정보 처리 과정에서의 정합성과 안정성을 검증하는 데 목적을 둠.

6.1.2 테스트 대상

- 전화번호 기반 회원가입 및 인증
- 이메일/비밀번호 기반 로그인
- 회원 정보 조회 및 수정
- 비밀번호 변경 및 재설정
- 회원 탈퇴

6.1.3 테스트 환경

- 앱 플랫폼: Android
- 백엔드 서비스: Firebase
- 테스트 도구: Android Studio Emulator / 실제 디바이스

6.1.4 테스트 방법

- 기능 기반 테스트 (기능 흐름을 시나리오로 검증)
- Firebase API 호출 여부 및 응답 확인
- UI 및 UX 흐름 정상 동작 확인

6.1.5 테스트 기준

- 각 기능 수행 시 정상 응답 및 처리 결과 확인
- 예외 상황에서도 오류 메시지 표시 및 안정적인 복구
- 사용자 데이터가 Firebase에 정확히 저장 및 삭제

6.2 회원관리

Test Case ID	TC_001
테스트명	회원가입 (전화번호 인증)
입력값	이메일, 전화번호, 비밀번호
테스트 절차	정보 입력 → 인증요청 클릭 → 인증번호 입력 → Firebase 계정 생성
예상 결과	인증번호 확인 후 가입 완료
비고	전화번호 인증 실패 시 오류 메시지 표시

6.3 전화번호 인증

Test Case ID	TC_002
테스트명	전화번호 인증
입력값	전화번호, 인증번호

테스트 절차	인증번호 전송 → 인증번호 입력 → Firebase 인증 확인
예상 결과	인증 성공 시 계정 활성화
비고	인증번호 만료/오입력 케이스 필요

6.4 로그인

Test Case ID	TC_003
테스트명	로그인 (이메일/비밀번호)
입력값	이메일, 비밀번호
테스트 절차	로그인 정보 입력 → 로그인 클릭 → Firebase 인증 요청
예상 결과	로그인 성공 시 홈 이동, 오류 시 메시지 출력
비고	이메일 인증 여부 확인 필요

6.5 회원정보 조회

Test Case ID	TC_004
테스트명	회원정보 조회
입력값	없음 (로그인 상태)
테스트 절차	'내 정보' 페이지 접근 → Firebase에서 사용자 정보 조회
예상 결과	사용자 이메일, 전화번호 표시
비고	로그인 유지 필요

6.6 비밀번호 변경

Test Case ID	TC_005
테스트명	비밀번호 변경
입력값	기존 비밀번호, 새 비밀번호
테스트 절차	기존 비번 확인 → 새 비번 입력 후 저장 → Firebase 업데이트
예상 결과	변경 완료 메시지 출력
비고	비번 형식 유효성 검사 포함

6.7 비밀번호 재설정

Test Case ID	TC_006
테스트명	비밀번호 재설정
입력값	이메일
테스트 절차	재설정 클릭 → 이메일 입력 → 메일 수신 후 링크 클릭 → 새 비번 입력
예상 결과	비밀번호 재설정 완료
비고	메일 수신 여부 확인 필요

6.8 회원 탈퇴

Test Case ID	TC_007
테스트명	회원 탈퇴
입력값	없음
테스트 절차	탈퇴 클릭 → 확인 → Firebase delete() 호출
예상 결과	계정 삭제 및 로그아웃 처리
비고	삭제 후 재접속 불가 확인 필요

6.9 통합 테스트 시나리오 표

Test ID	기능 / 조건	입력 조건 / 시나리오 설명	기대 결과	Pass/Fail
TC-01	현재 시간 표시	앱 실행	시스템 시간 출력 (예: 08:45)	
TC-02	혼잡도 API 연동	API 호출 정상	кан별 혼잡도 데이터 표시	
TC-03	혼잡도 없는 경우	API 오류 발생	"데이터 없음" 메시지 출력	
TC-04	추천 칸 표시	혼잡도 데이터 비교	가장 여유 있는 칸 강조	
TC-05	다른 시간대 보기	버튼 클릭	시간대별 예측 화면 전환	

TC-06	API 응답 = NULL	혼잡도 데이터 없음	"데이터 없음" 메시지 출력, 화면 기본 상태 유지	
TC-07	API 응답 = 예러(500/404)	서버 오류 발생	"데이터 없음" 메시지 출력, 캐시 있으면 최근 값 표시	
TC-08	혼잡도 값 = 0%	극단적 최소 혼잡도 입력	칸 색상 , "여유 있음" 메시지	
TC-09	혼잡도 값 = 100%	극단적 최대 혼잡도 입력	칸 색상 , "매우 혼잡" 메시지	
TC-10	혼잡도 값 = -5% 또는 120%	잘못된 범위 값 입력	무효 처리, "데이터 오류" 메시지 출력	
TC-11	네트워크 끊김	응답 없음	마지막 정상 데이터 2 분 표시 → 이후 "실시간 데이터 수신 불가" 메시지	
TC-12	API 응답 지연	응답 3 초 이상 지연	로딩 애니메이션 표시 → 타임아웃 시 "데이터 없음" 메시지	
TC-13	정상 혼잡도 값	예: 45% 입력	색상 표시, 실시간 갱신 반영	
TC-14	갱신 주기(30 초)	자동 업데이트	30 초마다 데이터 새로고침, UI 반영 확인	
TC-15	API 장애 후 복구	장애 후 정상 수신	정상 데이터 수신 시 자동으로 화면 갱신	

7 회의록

일시	3 월 10 일	장소	pc 22 실
참석자	최수빈, 박정현, 정예슬, 최원희, 한연수		

회의 안건/목적	LLM 기반 스마트 일정 관리 시스템 기획안 설계
회의 내용	<ul style="list-style-type: none"> • LLM 기반 개인 맞춤형 일정 관리 및 지식 관리 시스템 <ul style="list-style-type: none"> ◦ 음성 입력을 지원하는 AI 일정 관리 챗봇 ◦ 사용자의 학습/업무 패턴을 분석 ◦ 사용자의 시간대와 일정을 예측해 계획 추천 • 프로젝트 목적 <ul style="list-style-type: none"> ◦ 사용자의 일정과 지식을 효율적으로 관리할 수 있는 개인 맞춤형 일정 관리 시스템 개발이 목표임. ◦ 기존 일정 관리 도구의 불편함(수동 입력, 분산된 정보 등)을 해소하고, 자연어 기반 인터페이스를 통해 누구나 직관적으로 사용할 수 있게 함. ◦ LLM을 활용해 일정 자동 등록, 수정, 추천, 메모 정리 기능 등을 제공함으로써 사용자의 업무 효율성 및 생산성을 향상시키고자 함. • 필요성 <ul style="list-style-type: none"> ◦ 현대인의 복잡한 일정 속에서 시간 관리가 어려운 점을 해결할 필요 있음. ◦ 기존 캘린더/메모 앱은 입력이 번거롭고 추천 기능 부족함. ◦ LLM 기반 시스템은 자연어 처리로 일정을 쉽게 추가하고, 자동 정리 및 추천 기능까지 제공 가능. ◦ 최근 LLM 기술 발전으로 자연어 이해 및 예측 능력이 크게 향상되어, 사용자의 의도에 맞는 스마트한 일정 관리 구현이 가능해짐. • 기대 효과 <ul style="list-style-type: none"> ◦ 일정 관리 자동화 및 시간 절약 <ul style="list-style-type: none"> - 음성 또는 텍스트 입력만으로 일정을 자동 등록/수정 가능. - 사용자 부담 줄이고 생산성 향상 가능. ◦ AI 기반 맞춤 추천 <ul style="list-style-type: none"> - 일정 패턴 분석하여 개인화된 일정/업무 시간 추천 가능. - 예: 집중 잘 되는 시간에 중요한 업무 배치 등. ◦ 메모 및 지식 관리 최적화 <ul style="list-style-type: none"> - 회의록, 노트 등 텍스트를 자동 요약 및 정리, 검색 가능. - 태깅 기능으로 정보 체계화 가능. ◦ 통합 일정 관리 <ul style="list-style-type: none"> - Google Calendar, Notion, Slack 등과 연동 가능. - 하나의 인터페이스에서 일정 통합 관리 가능. ◦ 사용자 편의성 증대 <ul style="list-style-type: none"> - 자연어 기반 UI로 사용자가 손쉽게 일정을 추가/조회할 수 있음. - 다양한 사용자의 실생활에 실용적으로 활용 가능.

결정사항 및 향후 일정	
--------------------	--

일시	3 월 14 일	장소	pc 22 실
참석자	최수빈, 박정현, 정예슬, 최원희, 한연수		
회의 안건/목적	LLM 기반 스마트 일정 관리 시스템 기획안 설계		
회의 내용	<ul style="list-style-type: none"> • 기술 및 구현 계획 <ul style="list-style-type: none"> ◦ 기획 및 분석 단계 <ul style="list-style-type: none"> - 기존 일정 시스템 문제 분석 - LLM 도입 필요성, 핵심 기능 정리 (일정 추가/수정, 자동 정리, 추천 등) - 기술 스택 선정 (HTML, CSS, JavaScript 등) ◦ 핵심 기능 개발 <ul style="list-style-type: none"> - 자연어 일정 추가 기능 (예: “내일 회의 추가해줘”) - 일정 추천 기능: 일정 패턴 분석 후 최적 시간 추천 - 일정 데이터 저장: Google Sheets 또는 Firebase 활용 ◦ UI/UX 개발 <ul style="list-style-type: none"> - 웹 기반 인터페이스 구축 - 모바일 대응 간단한 반응형 디자인 적용 ◦ 테스트 및 개선 <ul style="list-style-type: none"> - 주요 기능 동작 확인 및 사용자 피드백 반영 - 성능 개선 및 UI 개선 진행 ◦ 최종 산출물 <ul style="list-style-type: none"> - 프로젝트 발표 자료 및 기술 문서 - 사용 매뉴얼 작성 • 기타 고려사항 <ul style="list-style-type: none"> ◦ 데이터 보안 <ul style="list-style-type: none"> - 민감한 정보 포함 가능성 있으므로 AES, TLS 암호화 및 • OAuth2 인증 필요 <ul style="list-style-type: none"> ◦ LLM API 사용 시 외부 전송 데이터에 대한 보안 대책 검토 필요 <ul style="list-style-type: none"> - 모델 선택 및 비용 문제 		

	<ul style="list-style-type: none"> ▪ GPT, Llama, Mistral 등 모델 비교 ▪ 캐싱 또는 소형 모델(GPT-3.5, MiniLM 등) 병용 고려 - 성능 및 확장성 <ul style="list-style-type: none"> ▪ 벡터 DB(Weaviate, Pinecone, FAISS) 활용 고려 ▪ 프롬프트 최적화 통해 응답 속도 개선 - 다국어 및 음성 명령 지원 <ul style="list-style-type: none"> ▪ 영어, 중국어 등 다국어 지원 여부 검토 ▪ Google Assistant, Siri 등과 연동 여부 검토
결정사항 및 향후 일정	

일시	3 월 17 일	장소	pc 22 실
참석자	최수빈, 박정현, 정예슬, 최원희, 한연수		
회의 안건/목적	주제 변경 보고		
회의 내용	<ul style="list-style-type: none"> • 프로젝트 주제 변경 관련 공유 <ul style="list-style-type: none"> ◦ 기존 주제 관련 <ul style="list-style-type: none"> - 멘토가 일정관리 앱 아이디어 좋게 봄 부분에 대해 감사함. 일정 관리 중심으로 사용자 편의성을 높이는 게 원래 목적이었음. ◦ 주제 변경 배경 <ul style="list-style-type: none"> - 출작 준비하면서 교수님과 소통하는 과정에서 약간 혼선 생김. 그동안 팀 내에서도 고민 많았고, 결과적으로 주제를 조금 바꾸게 됨. ◦ 변경된 주제 내용 <ul style="list-style-type: none"> - 새로운 주제는 지하철 이용 시 겪는 불편함 해결하는 앱임. 실시간 데이터 분석과 예측 시스템 활용해서 사용자에게 좌석이나 탑승 칸 추천해줌. - 즉, 더 효율적이고 쾌적한 지하철 이용 경험 제공하는 게 핵심임. ◦ 기존 주제와의 연결점 		

	<ul style="list-style-type: none"> - 기존 일정관리처럼 사용자 편의성 높이는 게 여전히 목적임. 다만, 일정 관리보다는 실제 생활에서 더 실용적인 기능 중심으로 바됨. 일정 관리 기능도 서브로 포함할 예정. ○ 요청사항 <ul style="list-style-type: none"> - 갑작스럽게 주제 바뀐 점에 대해 양해 구함.
결정사항 및 향후 일정	

일시	3월 24일	장소	pc 22실
참석자	최수빈, 박정현, 정예슬, 최원희, 한연수		
회의 안건/목적	지하철 좌석 예측 및 추천 앱 개발		
회의 내용	<ul style="list-style-type: none"> • 프로젝트 목적, 필요성, 수행 계획, 기대효과, 주요기능, 검토사항 및 기타 논의 사항 정리 <ul style="list-style-type: none"> ○ 프로젝트 목적 및 필요성 <ul style="list-style-type: none"> - 출퇴근 시간 지하철 좌석 부족 문제 해결 필요 - 하차·탑승 정보 공유 어려움 → 좌석 회전율 저조 - 실시간 간별 혼잡도 정보 부족 → 비효율적 탑승 - 앱을 통해 하차역 사전 입력, 좌석 정보 공유, 혼잡도 예측 가능 - AI 기반 혼잡도 예측 및 좌석 추천 시스템 필요성 대두 ○ 수행 내용 및 개발 기능 <ul style="list-style-type: none"> ○ AI 기반 기능 <ul style="list-style-type: none"> - 시간대·노선별 혼잡도 분석 및 추천 - 간별 실시간 혼잡도 예측 - 사용자 맞춤형 탑승 시간/위치 추천 - 머신러닝/딥러닝 기반 예측 모델 구축 ○ 데이터 수집 및 분석 <ul style="list-style-type: none"> - 교통공사 API 활용 (혼잡도, 탑승 인원 등 수집) - 노선·역·요일별 패턴 분석 후 예측 모델에 적용 ○ 앱 기능 개발 <ul style="list-style-type: none"> - 하차역 사전 입력 기능 		

- 실시간 좌석 정보 제공
- 목적지 기반 최적 좌석 추천
- 개인 맞춤형 알림 기능
- UI/UX 설계
 - 직관적 인터페이스 및 다국어 지원
 - 모바일 최적화 반응형 디자인
 - 테스트 및 고도화
 - 베타 테스트 → 예측 정확도, UX 개선
 - 사용자 피드백 기반 기능 보완 및 안정화
- 기대 효과
 - 승객 편의성 향상
 - 하차역 정보 공유 → 좌석 확보 용이
 - 쾌적한 이용 유도
 - 환승 효율 향상
 - 운영 효율성 증가
 - 좌석 회전율 향상
 - 승객 분산 유도
 - 배차 간격/차량 수 조정 가능
 - 사회적 효과
 - 사회적 효과교통 약자 배려 가능성 증가
 - 출퇴근 스트레스 완화
 - 탄소 저감 효과
 - 기술 및 확장성
 - 실생활 적용 AI 실증
 - 고속버스·기차 등으로의 확장 가능
 - 비상상황 대응
 - 자연·중단 시 대체 경로 안내
 - 긴급 좌석 정보 공유
 - 푸시 알림 통한 신속 정보 전달
- 주요 기능 및 확장 아이디어
 - 하차역 정보 공유
 - 좌석 배치도에 행선지 표시, 자리 교환 알림
 - 실시간 목적지 표시 알림
 - 앞/뒤칸 추천 기능
 - 통계 기능 (이용 빈도 등)
 - 빈 좌석 알림 기능
 - 자리 예약 또는 미리 확보 기능
 - 커뮤니티 기능 (혼잡도 공유, 후기 작성 등)

	<ul style="list-style-type: none"> ○ SNS 연동 기능 (경험 공유) • 검토 및 고려 사항 <ul style="list-style-type: none"> ○ 보안 및 개인정보 보호 <ul style="list-style-type: none"> - 하차/위치 정보 익명화 및 암호화 - 관련 법규 준수 및 동의 절차 마련 ○ 기술적 안정성 <ul style="list-style-type: none"> - 서버 안정성 및 오류 방지 - 교통공사 협업 필요 - AI 모델 경량화 및 비용 최적화 ○ 사용자 경험(UX) <ul style="list-style-type: none"> - 기능 선택적 제공 - 알림 최소화 및 UI 피로도 고려 • 기타 논의 및 예상 결과물 <ul style="list-style-type: none"> ○ 앱 로고 및 UI 시안 작업 중 ○ 결과물: <ul style="list-style-type: none"> - AI 기반 지하철 좌석 추천 앱 (iOS/Android 지원) - 웹 시연용 버전 ○ 발표 자료 및 데모 영상 제작 예정
결정사항 및 향후 일정	향후 기술 검토, 기능 우선순위 선정, 외부 API 연동 계획 등을 구체화 시킬 예정.

일시	3월 31일	장소	pc 22실
참석자	최수빈, 박정현, 정예슬, 최원희, 한연수		
회의 안건/목적	프로젝트 계획을 바탕으로 교수님 피드백		
회의 내용	<ul style="list-style-type: none"> • 앱 개발 방향 및 전략 <ul style="list-style-type: none"> ○ 앱 개발은 단순한 기능 구현보다 사용자 실효성에 집중해야 함 (예: 예측 결과 시각화, 인터페이스 단순화 등은 명확한 목적을 중심으로 서술) ○ "사용자 경험 최적화"라는 표현은 구체적이고 짧은 설명으로 대체해야 함 (예: 사용 흐름이 자연스럽게 이어지도록 구성) ○ 베타 테스트는 반드시 사용자 피드백을 반영하는 실사용 기반으로 진행해야 함 • 임산부 전용 하차 기능 		

	<ul style="list-style-type: none"> ○ 노약자석 관리 기능은 현실적으로 구분이 어려워 구현이 힘듦 ○ 임산부는 식별 및 기능 적용이 가능하므로, 임산부 전용 하차 입력 시스템을 적용할 수 있음 • 앱 개발 목적과 방향성 <ul style="list-style-type: none"> ○ 이번 프로젝트는 수익 창출이나 기업 설립이 목적이 아님 ○ 지하철 앱에 새로운 기능을 더한 실용적인 앱을 만드는 것이 목표임 ○ 기존 지하철 앱과 차별화되는 기능(하차 알림, 커뮤니티 등)을 포함해야 메리트가 있음 • 커뮤니티 기능 방향 <ul style="list-style-type: none"> ○ 지하철 앱에 커뮤니티를 넣어 이용자 간 정보 공유 가능하도록 할 계획임 ○ 혼잡도 공유, 후기 작성 등 참여형 기능을 고려함
결정사항 및 향후 일정	<ul style="list-style-type: none"> • AI 기반 설정 방법 파악, API 활용 혼잡도 판단 기준 설정, 기술 구현 방안 모색, 관련 자료 추가 자료조사 • 주요 기술과 기능 설정, 시장조사(비슷한 앱, 관련 기사 찾아보기)

일시	4 월 2 일	장소	pc 22 실
참석자	최수빈, 박정현, 정예슬, 최원희, 한연수		
회의 안건/목적	LLM 기반 스마트 일정 관리 시스템 기획안 설계		
회의 내용	<ul style="list-style-type: none"> • API 기반의 하차 알림 시스템 구현, 사용자의 경험 최적화를 위한 디자인 개선, API를 활용한 혼잡도 판단 기준 설정 <ul style="list-style-type: none"> ○ 지하철 좌석 정보 처리 <ul style="list-style-type: none"> - 지하철 좌석은 번호가 아닌 형태로 구분되어 있어 센서 기반 감지가 가능함 - 하차 앱을 설치하면, 해당 좌석 하차자 알림이 울려 양보 및 회전율이 유도됨 - 사용자 경험이 긍정적으로 이어지면 선순환 효과로 앱 지속 사용 가능성 높아짐 ○ 차량 단위 혼잡도 판단 <ul style="list-style-type: none"> - 좌석 단위 혼잡도보다는 차량 단위로 예측하는 것이 기술적으로 더 실현 가능함 - 지금은 이벤트 기능이 필수는 아니지만, 기술적으로 구현해 둘 필요가 있음 - 예: 8 량 중 4 량이 100% 탑승, 나머지 60~70%일 때 혼잡 기준을 어떻게 정할지 내부 기준 수립 필요 		

	<ul style="list-style-type: none"> ○ 좌석 관련 기능 설계 기준 <ul style="list-style-type: none"> - 좌석을 전면 기능으로 강조할 경우 비판 대상이 될 수 있음 - 좌석 관련 기능은 일부만 반영하거나 보조 기능으로 설계하는 것이 바람직함 - 좌석 효율 측정은 총 좌석 수 대비 실제 이용률을 기준으로 판단함
결정사항 및 향후 일정	

일시	4 월 5 일	장소	pc 22 실
참석자	최수빈, 박정현, 정예슬, 최원희, 한연수		
회의 안건/목적	LLM 기반 스마트 일정 관리 시스템 기획안 설계		
회의 내용	<ul style="list-style-type: none"> • 외부 API 활용 방안 <ul style="list-style-type: none"> ○ 한국도로공사, 교통공사 등에서 제공하는 공공 API를 활용할 수 있음 <ul style="list-style-type: none"> - 예: 2 호선 시청역, 한양대역 등에서 시간대별 차량별 승차 인원 데이터 확인 가능 ○ 앱 자체 연동보다는 외부 API 데이터를 가져오는 방식이 적절함 • 혼잡도 기준 정의 필요 <ul style="list-style-type: none"> ○ 시간대별 탑승 데이터를 기반으로 혼잡도를 판단할 기준이 필요함 ○ 차량별 인원 비율에 따라 혼잡/비혼잡을 구분하는 명확한 기준 정립이 필요함 ○ 예측 알고리즘 이전에 기준 수립이 우선되어야 함 		
결정사항 및 향후 일정			

일시	4 월 7 일	장소	pc 22 실
참석자	최수빈, 박정현, 정예슬, 최원희, 한연수		
회의 안건/목적	앱 이름 및 주 타겟층 설정, 하차/좌석 알림 UX 설계		

회의 내용	<ul style="list-style-type: none"> • 주요 사용자 타겟층 <ul style="list-style-type: none"> ◦ 주 사용층은 30~50 대 출퇴근 직장인으로 설정함 ◦ 장거리 지하철 이용자를 대상으로 좌석 이동 효율성 향상을 목표로 설계 ◦ 하차 알림 기능은 하차 준비에 도움, 사회적 배려 문화 확산에도 기여 • 주요 기능 및 UX 설계 <ul style="list-style-type: none"> ◦ 하차 알림 UX 시나리오 <ul style="list-style-type: none"> - 사용자 A 가 “강남역 하차” 입력 - 한 정거장 전, 사용자 A에게 진동 알림과 함께 하차 문 방향 안내 전송 (예: “이번 역 하차는 왼쪽 문입니다”) - 주변 사용자 B에게 “이 근처 좌석이 곧 비워질 예정입니다”라는 푸시 알림 전송 - 사용자 B는 미리 좌석을 확보할 수 있어 자연스러운 자리 이동이 유도됨 ◦ *임산부 좌석 요청 기능 <ul style="list-style-type: none"> - 임산부가 탑승 시 좌석 요청 버튼 사용 가능. - 요청 알림은 조용하게 주변 승객에게 전송되어 익명성과 비강제성이 유지 - 타인에게 부담 없이 자연스러운 배려 행동을 유도함. ◦ 예시 알림 메시지 <ul style="list-style-type: none"> - “몇 분 후 이 좌석이 비워질 예정입니다”, “지금 4-1칸이 가장 여유 있어요” - “내가 하차하면 자리가 비어요” 등 알림 메시지는 UX 흐름에 따라 시점별로 자동 제공되며, 혼잡 완화 및 착석 기회 제공에 기여함.
결정사항 및 향후 일정	<ul style="list-style-type: none"> • 계획서에 기존 회의 내용 추가, 계획서 요약, 요구사항과 기능 명세서 수정, 앱&팀 로고 AI 사용해서 생성할 것. • 추후 알림 타이밍, 문구 톤, 사용자 설정 범위 등에 대한 추가 테스트 및 개선.

일시	4 월 14 일	장소	pc 22 실
참석자	최수빈, 박정현, 정예슬, 최원희, 한연수		
회의 안건/목적	발표 준비 및 본격적인 앱 디자인, 코딩 진행		

회의 내용	<ul style="list-style-type: none"> • 대체과제 1부 제본 <ul style="list-style-type: none"> ◦ 회원관리 페이지가 어떻게 작동 되는지 유튜브에 올리고 링크를 보고서에 추가 ◦ 12 일까지 제본 값은 조교님한테 가서 카드 받기 • 발표도 있음 1명만 아니고 5 명 다 <ul style="list-style-type: none"> ◦ 시스템 구조 포함 ◦ 화면의 CRUD 구현, 첫번째 코드 구현 ◦ 유튜브에 올리고 링크를 보고서에 추가 • 추가 내용 <ul style="list-style-type: none"> ◦ 수빈 DB 설계서 ◦ 원희 CRUD SQL ◦ 정현 CRUD 화면, 코드랑 실행화면 캡쳐 ◦ 연수 Table 설계서 ◦ 예슬 요구사항 명세서 수정, 그거 바탕으로 기능 명세서 수정 • 회원가입 본인인증 <ul style="list-style-type: none"> ◦ 이름, 생년월일, 휴대폰 번호, 보안문자 ◦ 입력한 휴대폰 번호로 인증번호 전송 ◦ 받은 인증번호 입력후 본인인증 완료 • 앱 이름 투표 → 자리잇 결정
결정사항 및 향후 일정	<ul style="list-style-type: none"> • 기능 명세서 수정 • 회원관리 CRUD 코딩 • 개발 환경 다이어그램으로 표현 • 로고 수정 • 테이블 설계서 수정

일시	4 월 21 일	장소	pc 22 실
참석자	최수빈, 박정현, 정예슬, 최원희, 한연수		
회의 안건/목적	발표 및 피드백		
회의 내용	<ul style="list-style-type: none"> • 개발 계획서 <ul style="list-style-type: none"> ◦ 배경, 목적, 개요, 시스템의 범위, 기능 • 기본 설계서 <ul style="list-style-type: none"> ◦ UI, 시스템 아키텍처, SW/HW 아키텍처, 네트워크 아키텍처, 시스템 인터페이스 • 상세 설계서 <ul style="list-style-type: none"> ◦ 모듈 내부구조, 데이터 구조, 네트워크 구조, 처리 흐름, 메세지 교환 		

	<ul style="list-style-type: none"> • 프로젝트 종료 보고서 <ul style="list-style-type: none"> ◦ 배경, 목적, 개요, 시스템의 범위, 기능, 산출물, 측정 필요 ◦ 최대한 많은 승객을 태워야 하는 이유로 지정좌석제를 지하철에 도입하기엔 효율이 너무 떨어짐. 또한 앉고 일어서는 횟수가 많은 지하철 좌석 특성상 뛰어난 내구성의 좌석을 설치해야 하는 것이 필수적. • 기존 아이디어(지정좌석제) 수정 또는 보완 필요(피드백 부분) <ul style="list-style-type: none"> ◦ 전면 지정좌석제 대신 하차 알림 시스템 + 혼잡도 기반 좌석 추천 위주로 전환 검토 • 시스템 요구사항 문서 수정 <ul style="list-style-type: none"> ◦ 지정좌석 기능의 범위 조정 및 UI 흐름도 수정 • 기능 우선순위 재조정 및 프로토타입 시나리오 변경 <ul style="list-style-type: none"> ◦ 필수 기능에 "하차 알림"과 "칸 추천"을 우선 반영 • 하드웨어 구조 변경 여부 검토 <ul style="list-style-type: none"> ◦ 내구성 강화를 위한 물리적 좌석 개선은 제외하고, 소프트웨어 기반 알림/추천 시스템 위주 개발
결정사항 및 향후 일정	<ul style="list-style-type: none"> • 기능 명세서 수정안 초안, UI 흐름 재구성 및 아키텍처 도식, 문서 갱신, 지정좌석제 → 하차 알림 중심으로 발표자료 수정

일시	5 월 5 일	장소	pc 22 실
참석자	최수빈, 박정현, 정예슬, 최원희, 한연수		
회의 안건/목적	발표 준비		
회의 내용	<ul style="list-style-type: none"> • 앱 개발 발표 및 과제 제출 준비를 위한 역할 분담 및 일정 조율 <ul style="list-style-type: none"> ◦ 수빈 <ul style="list-style-type: none"> - PPT 목요일까지 완성 후 공유 예정 - 전체 발표 흐름 정리 ◦ 예슬 <ul style="list-style-type: none"> - PPT 및 공유 문서 내용 확인 - 금요일까지 발표 대본 작성 후 단톡방에 업로드 - 내용 숙지 ◦ 정현 <ul style="list-style-type: none"> - 회원가입 화면 캡처 및 실행화면 녹화 - 영상 유튜브 업로드 		

	<ul style="list-style-type: none"> - 깃허브에 코드 업로드 및 링크 공유 ○ 원희 & 연수 <ul style="list-style-type: none"> - 과제 산출물 정리 및 제본형태로 출력 - 교수님이 올린 문서의 순서에 맞춰 구성
결정사항 및 향후 일정	<ul style="list-style-type: none"> • 발표용 로고, 시스템 구조 다이어그램 정리 • 교수님 피드백 반영 사항 점검 • 앱 기능 시연 영상이 PPT 나 발표에 포함되었는지 확인 • 각자 맡은 파트 정확히 숙지하고 발표 연습

일시	5 월 12 일	장소	pc 22 실
참석자	최수빈, 박정현, 정예슬, 최원희, 한연수		
회의 안건/목적	발표 후 피드백		
회의 내용	<ul style="list-style-type: none"> • 앱 이름 및 상세 설명 보강 필요 <ul style="list-style-type: none"> ○ 앱 이름 ‘자리잇(ZARIIT)’의 의미와 구성 설명 필요, 앱 소개에 구체적인 기능 설명 포함, 어떤 문제를 해결하기 위한 앱인지, 핵심 기능은 무엇인지 명확하게 기술 • 개발 과정 이미지 추가 <ul style="list-style-type: none"> ○ 안드로이드 스튜디오로 구현한 과정이나 UI 제작 과정의 캡쳐 이미지 필요 • Man-Month 관련 설명 부족 <ul style="list-style-type: none"> ○ 우리가 이 앱을 개발하면서 쓴 시간과 인건비 개념 정리 ○ 총 투입 인원 수 × 주당 시간 × 참여 주차 계산 ○ 실제 인건비로 환산한 결과까지 포함해도 좋음 • 기술 설명 보완 <ul style="list-style-type: none"> ○ 사용 기술에 대한 설명 부족 ○ 사용자 맞춤형 기능(예: 하차역 설정), API 활용, AI 도입 예정 기술 등 ○ 향후 혼잡도 기반 칸 추천 기능은 AI 기반임을 명확히 강조 • 혼잡도 분석 내용 보강 <ul style="list-style-type: none"> ○ 앱의 핵심이 하차알림 + 혼잡도 분석이므로 혼잡도 분석 방법, 수집 방식, 적용 사례 등을 상세하게 기술해야 함 • 개발 선택 과정 기술 <ul style="list-style-type: none"> ○ 여러 기능/아이디어 중 어떤 과정을 통해 이 앱을 선택했는지 작성 ○ 아이디어 발상 → 기능 정리 → 최종 선택 과정을 서사 형식으로 설명해도 좋음 		

결정사항 및 향후 일정	<ul style="list-style-type: none"> • 앱 이름, 기능 구체적인 설명 추가 • 개발과정 이미지, 맨먼스 설명 추가 • 백엔드 연동
---------------------	--

일시	5월 19일	장소	pc 22실
참석자	최수빈, 박정현, 정예슬, 최원희, 한연수		
회의 안건/목적	앱에서 저장할 데이터의 종류 조사		
회의 내용	<ul style="list-style-type: none"> • 앱에서는 크게 두 가지 종류의 데이터를 저장 <ul style="list-style-type: none"> ◦ 서비스 관련 데이터 <ul style="list-style-type: none"> - 예: 지하철 노선, 실시간 훈잡도, 하차 알림 설정, 역 정보 등 - → 주로 읽기 위주, 즉 사용자들이 조회만 많이 하고 자주 바뀌지 않거나 외부 서버에서 받아오는 데이터 ◦ 회원 정보 관련 데이터 <ul style="list-style-type: none"> - 예: 로그인 정보, 사용자 하차 역 설정, 즐겨찾는 노선, 좌석 지정 기록 등 - → 읽고 쓰기 모두 필요하고, 사용자마다 다른 개인화된 정보 • 각 데이터 저장공간 <ul style="list-style-type: none"> ◦ 서비스 관련 데이터 → 외부 서버(API) 또는 로컬 캐시 <ul style="list-style-type: none"> - 외부 API (ex. 공공데이터포털 API) - 로컬 캐시는 Room(로컬 앱), SharedPreferences - 실시간 정보는 서버에서 받아와야 최신 상태 유지 가능 - 자주 바뀌지 않으면 앱 안에 저장해도 됨 ◦ 회원 정보 (로그인, 즐겨찾기 등) → 로컬 DB or Firebase <ul style="list-style-type: none"> - Firebase Realtime Database / Firestore - Room (SQLite 기반) - Firebase 는 사용자별 데이터 관리에 강함 - Room 은 기기 내부 저장용으로 간단한 구조에 적합 		
결정사항 및 향후 일정	<ul style="list-style-type: none"> • 저장공간 조사 후 백엔드 연동 • 데이터 동기화 		

일시	5월 19일	장소	pc 22실
참석자	최수빈, 박정현, 정예슬, 최원희, 한연수		

회의 안건/목적	앱에서 저장할 데이터의 종류 조사
회의 내용	<ul style="list-style-type: none"> • 앱에서는 크게 두 가지 종류의 데이터를 저장 <ul style="list-style-type: none"> ◦ 서비스 관련 데이터 <ul style="list-style-type: none"> - 예: 지하철 노선, 실시간 훈잡도, 하차 알림 설정, 역 정보 등 - → 주로 읽기 위주, 즉 사용자들이 조회만 많이 하고 자주 바뀌지 않거나 외부 서버에서 받아오는 데이터 ◦ 회원 정보 관련 데이터 <ul style="list-style-type: none"> - 예: 로그인 정보, 사용자 하차 역 설정, 즐겨찾는 노선, 좌석 지정 기록 등 - → 읽고 쓰기 모두 필요하고, 사용자마다 다른 개인화된 정보 • 각 데이터 저장공간 <ul style="list-style-type: none"> ◦ 서비스 관련 데이터 → 외부 서버(API) 또는 로컬 캐시 <ul style="list-style-type: none"> - 외부 API (ex. 공공데이터포털 API) - 로컬 캐시는 Room(로컬 앱), SharedPreferences - 실시간 정보는 서버에서 받아와야 최신 상태 유지 가능 - 자주 바뀌지 않으면 앱 안에 저장해도 됨 ◦ 회원 정보 (로그인, 즐겨찾기 등) → 로컬 DB or Firebase <ul style="list-style-type: none"> - Firebase Realtime Database / Firestore - Room (SQLite 기반) - Firebase 는 사용자별 데이터 관리에 강함 - Room 은 기기 내부 저장용으로 간단한 구조에 적합
결정사항 및 향후 일정	<ul style="list-style-type: none"> • 저장공간 조사 후 백엔드 연동 • 데이터 동기화

일시	5 월 21 일	장소	pc 22 실
참석자	최수빈, 박정현, 정예슬, 최원희, 한연수		
회의 안건/목적	추천 시나리오별 정리		
회의 내용	<ul style="list-style-type: none"> • 추천 시나리오별 정리 <ul style="list-style-type: none"> ◦ 서비스 데이터 (훈잡도, 지하철 정보) <ul style="list-style-type: none"> - 훈잡도: 외부 API 에서 받아오는 게 정확하고 최신임 (예: 서울시 지하철 API) 		

	<ul style="list-style-type: none"> - 지하철 역/노선 정보: 앱 내 Room DB에 한 번 저장해두면 빠르게 불러옴 - 처음 앱 설치 시 한 번 받아서 Room에 저장, 주기적으로 업데이트 ○ 회원 정보 (로그인, 하차 알림, 즐겨찾기 등) <ul style="list-style-type: none"> - 로그인: Firebase Authentication 쓰면 구글/이메일 로그인 쉽게 가능 - 하차 알림 설정, 즐겨찾기: Firebase Firestore 또는 Firebase Realtime Database 추천(사용자 ID 별로 데이터 구분하고 저장 가능) - Firebase는 클라우드 기반이라 여러 기기에서 정보 동기화가 가능 ● 전체 구조 요약 <ul style="list-style-type: none"> ○ [사용자 스마트폰] <ul style="list-style-type: none"> - Room DB (지하철 역 정보, 하차 설정 로컬 저장) - SharedPreferences (단순 설정값 저장) - Firebase (회원 정보, 즐겨찾기, 로그인) ○ [외부 서버] <ul style="list-style-type: none"> - 공공 API (지하철 혼잡도, 실시간 도착 정보 등)
결정사항 및 향후 일정	<ul style="list-style-type: none"> ● 위 시나리오 내용을 기술 설계서 문서 형태로 정리 ● 정리된 기술 구성 내용을 보고서 본문에 들어갈 수 있는 문장 형태로 다듬기 ● Firebase나 Room 등을 이용한 하차 알림 예시 코드나 참고 자료 확보

일시	5월 26일	장소	pc 22실
참석자	최수빈, 박정현, 정예슬, 최원희, 한연수		
회의 안건/목적	서비스 모듈 구성 및 아이디어 구체화, 기술적 방향 설정		
회의 내용	<ul style="list-style-type: none"> ● 서비스 모듈 구성 <ul style="list-style-type: none"> ○ 서비스 모듈 구분 <ul style="list-style-type: none"> - 서비스를 여러 모듈로 나눠서 개발 → 혼잡도 분석, 하차 알림, 좌석 선택 등 - 화면 설계는 "회원 관리" 중심으로 구성 가능 - 기능명세서 포함된 구체적인 요구사항 필요 - 백엔드/프론트엔드 분리 개발 ○ 백엔드와 프론트엔드 역할 <ul style="list-style-type: none"> - 백엔드: DB 관리, 사용자 정보 저장 		

- 프론트엔드: 웹 로고, 디자인, 시각화 UI 담당, 역할 구분은 특성에 따라 유동적
- 업데이트와 협업
 - ‘회원 관리’ 모듈 중심으로 화면 설계, 백엔드-프론트엔드 협업 강조
- 서비스 아이디어
 - 시간대별 혼잡도 조회 기능
 - 현재 위치와 시간대 기반 혼잡도 제공, 혼잡도 판단은 API 기반, 자체 분석은 최소화
 - 사용자가 시간대 선택 → 해당 데이터 조회
 - 아이디어 구체화
 - 사용자 입력 → 혼잡도 결과 제공 방식
 - 실시간 분석 대신, 선택 기반 데이터 제공, 경쟁사 분석은 참고 용도
- 서비스 개발 및 데이터 관리
 - 개발 필요성과 과제
 - 기존 서비스 문제점 → 개선안 도출
 - 개발환경 설정 + 사례 기반 필요성 강조
 - 기능 정의 및 역할 분담 필요
 - 데이터 저장/분석 환경 세팅 필수
 - 기술적 고민
 - 예측 시스템, 개인화 서비스 등 기술 요소 고민
 - 알고리즘/AI 도입 가능성 검토
 - 데이터 분석을 통한 맞춤형 정보 제공에 초점
 - 개선 방안
 - 개인 선호 반영한 기능 개발
 - 좌석 예약 시스템 구현 논의
 - 기술적 접근 방식 명확화 필요
- 로고 제작 및 프로젝트 운영
 - 로고 제작
 - 생성형 기법 활용한 로고 제작
 - 결과물 개선 위한 과정 기록 및 수정 필요
 - 로고 제작 과정 이슈 → 문서화하여 관리
 - 프로젝트 팀 역할
 - 팀원별 역할 명확히 지정
 - 테크니컬 라이터: 문서 정리, 기술 설명 책임
 - 역할 분담에 따라 문서 작성 및 관리 수행
 - 프로젝트 운영 유의사항
 - 발표 역할 분담 명확화

	<ul style="list-style-type: none"> - 계획서, 기여도 문서화, 버전 관리 지속적으로 진행, 정보 공유 및 최신화 유지 필수
결정사항 및 향후 일정	<ul style="list-style-type: none"> • 백엔드/프론트엔드 역할 분담 명확하게 설정 • 주요 기능(좌석) 구현 구체화 • 로고 수정 및 API 신청 • 앱 색상 설정

일시	6 월 2 일	장소	pc 22 실
참석자	최수빈, 박정현, 정예슬, 최원희, 한연수		
회의 안건/목적	프로젝트 관리와 프로그래밍 방법론		
회의 내용	<ul style="list-style-type: none"> • 프로젝트 관리 및 프로그래밍 방법론 <ul style="list-style-type: none"> ◦ 전략적 접근 <ul style="list-style-type: none"> - 코딩 전 설계 단계의 중요성 강조 - 폭포수 모델 기반 단계별 관리 → 설계 후 코딩 진행 - 설계 단계 문제 발생 시 애자일 방식 또는 단계적 접근 고려 - 설계-코딩 사이 중간 점검 필수 ◦ 회원 관리 모듈 설계 및 구현 <ul style="list-style-type: none"> - 설계 완료 후 모듈별 코딩 시작 - 회원 관리 모듈 → 기반 기능으로 다른 기능 확장 - 기능별 모듈화 통해 코드 유지 관리 용이 ◦ 혼잡도 분석 기능 (Android 기반) <ul style="list-style-type: none"> - Android 기기를 활용한 실시간 혼잡도 분석 - 혼잡도 데이터는 파일로 저장되며, 언어별 읽기 가능 - 조건 기반 혼잡도 판정 → 좌석 대비 승객 수로 측정 - 정확한 혼잡도 분석을 통해 좌석 가용성 향상 필요 • 혼잡도 분석 및 앱 개발 <ul style="list-style-type: none"> ◦ 혼잡도 정의 및 계산 <ul style="list-style-type: none"> - 혼잡도 = 특정 시간대 기준 초과 이용 시 ‘혼잡’ 간주 - 도로교통사 등 기준 기관 참고 - 앱 설계 시 혼잡도 트렌드와 사용 빈도 반영 필수 ◦ 예측 기반 앱 개발 <ul style="list-style-type: none"> - 시간대별 혼잡도 예측 기능 구현 - 사용 트렌드 분석 → 시스템 개선 시나리오 생성 - 사용자 경험(UX) 고려한 기능 중심 구조 설계 ◦ 사용자 반응 기반 앱 개선 		

	<ul style="list-style-type: none"> - 사용자 피드백 수집 및 반영 - 개선 사항 알림, 리뷰 등으로 지속적 소통 중요 - 혼잡도 트렌드 변화를 알림 기능 통해 전달
	<ul style="list-style-type: none"> • 서비스 아키텍처 <ul style="list-style-type: none"> ◦ 서비스 계획 및 구조 <ul style="list-style-type: none"> - 구조 및 개발 환경 명확화 - 사용자 설정은 필수 아님 (추후 추가 가능) - 사용자 중심 구조 고려한 아키텍처 설계 ◦ 역할 분담 및 방향성 <ul style="list-style-type: none"> - 문서 담당자와의 협업 필요 - 쿼리/서버 모듈 분리 고려 - 사용자 설정 시 노약자 고려 - 유연한 구조 설계로 수정 용이하게 설계 ◦ 레퍼런스 및 수정 관리 <ul style="list-style-type: none"> - 서비스 참고 자료 정리 필수 - 수정 사항은 문서에 반영 - 아키텍처 및 문서는 유지/보완 가능한 형태로 작성
	<ul style="list-style-type: none"> • 인공지능(AI) 기능 확장 <ul style="list-style-type: none"> ◦ 네트워킹 및 알림 <ul style="list-style-type: none"> - Firebase 기반 클라우드 알림 기능 활용 - 실내 위치 파악 및 이동 탐지 기능 탑재 - 위성, 센서 기반 위치 추정 기능 검토 ◦ AI 기반 혼잡도 분류 <ul style="list-style-type: none"> - AI는 학습 + 기능 제공 모두 충족해야 - 데이터 학습을 통해 이동 가능 여부 판단 - 기능별 AI 학습 범위 명확히 설정 ◦ AI 모듈 아키텍처 설계 <ul style="list-style-type: none"> - 각 AI 모듈별 업무 정의 필요 - 데이터 전송, 위치 파악, 좌석 분석 기능 분리 - 팀원 역할 = 모듈 기반으로 분담 - 전체 시스템 흐름에 맞춰 모듈 기반 아키텍처 설계
	<ul style="list-style-type: none"> • 혼잡도 기준 및 앱 색상 정의 <ul style="list-style-type: none"> ◦ 혼잡도 기준 <ul style="list-style-type: none"> - 참고 자료- 서울교통공사 메트로 열차 혼잡도, 열린데이터광장 API ◦ 앱 디자인 색상 <ul style="list-style-type: none"> - colorPrimary: #0D3B66 (딥 네이비) - colorPrimaryVariant: #14477A (세컨 네이비) - colorAccent: #1E6091 (강조 블루) - grayLight: #D3D3D3 (밝은 회색)

	- grayDark: #696969 (어두운 회색)
결정사항 및 향후 일정	<ul style="list-style-type: none"> 시스템 아키텍처 수정 경쟁사 자료 수정

일시	6 월 9 일	장소	pc 22 실
참석자	최수빈, 박정현, 정예슬, 최원희, 한연수		
회의 안건/목적	지하철 사용자 이용 패턴 조사		
회의 내용	<ul style="list-style-type: none"> 사용자의 지하철 이용 패턴 예시 <ul style="list-style-type: none"> 시간 기반 패턴 <ul style="list-style-type: none"> 출퇴근 시간대 반복: 매일 오전 8 시경 A 역에서 승차, 오후 6 시경 B 역에서 하차 요일별 차이: 평일과 주말 이용 시간대가 다름 특정 요일 이용: 매주 화요일과 목요일에만 특정 노선을 이용 장소 기반 패턴 <ul style="list-style-type: none"> 특정 역 반복 이용: 주로 A 역 ↔ B 역만 왕복 다중 환승 패턴: C 역에서 항상 환승 후 특정 노선 이용 빈도 기반 패턴 <ul style="list-style-type: none"> 정기적 이용: 하루에 두 번 정해진 시간에 지하철 이용 비정기적 이용: 월 3~4 회 특정 시간에만 이용 행동 기반 패턴 <ul style="list-style-type: none"> 앱 실행 시점: 지하철 탑승 5 분 전 앱 실행 → 알림 설정 → 하차 알림 받음 알림 선호도: 특정 혼잡도 이상일 때만 알림 받는 사용자 어떻게 학습하는가? <ul style="list-style-type: none"> 사용자 ID, 탑승 시간, 하차 시간, 탑승역, 하차역, 요일, 날씨, 혼잡도 기계학습 알고리즘 적용 - 시계열 분석 순환 신경망, K-평균 군집화 의사결정트리 / 랜덤포레스 등 활용 예시 <ul style="list-style-type: none"> 매일 8:10 A 역 탑승 → 8:45 B 역 도착 내일도 비슷한 시간에 앱 알림 제공 금요일마다 C 역에서 환승 금요일 오전에 C 역 혼잡 예측 가능 수요일 저녁은 지하철 이용 X 해당 시간 알림 비활성화 가능 		

	<ul style="list-style-type: none"> 결론 <ul style="list-style-type: none"> 지하철 이용 패턴은 사용자의 시간, 장소, 행동 반복성을 중심으로 다양한 방식으로 나타날 수 있으며, 이를 기계학습을 통해 학습하면 다음 이용 시간 예측, 맞춤형 알림, 혼잡도 예측 등에 활용.
결정사항 및 향후 일정	<ul style="list-style-type: none"> 깃허브 연동 데이터베이스 설계 수정 소셜 로그인, 모듈, 홈화면/검색화면 구현 피피티 자료 만들기

일시	6 월 12 일	장소	pc 22 실
참석자	최수빈, 박정현, 정예슬, 최원희, 한연수		
회의 안건/목적	홈 화면 / 로그인 화면 구현		
회의 내용	<ul style="list-style-type: none"> 로그인 화면 <ul style="list-style-type: none"> 앱의 시작 화면으로, 사용자 이메일과 비밀번호를 입력하여 로그인하거나 회원가입 할 수 있음. 하단에 Google, Kakao, Naver 소셜 로그인 버튼이 있으며, 각각 브랜드 컬러로 시각적으로 구분 가능. 홈 화면 <ul style="list-style-type: none"> 검색창에 본인이 위치한 지하철역 입력. 하단에 사용자가 자주 찾는 지하철역이 노선 색상과 함께 버튼으로 표시. 노선 선택 화면 (역 검색 결과) <ul style="list-style-type: none"> 특정 역(예: "왕십리")을 검색했을 때, 해당 역이 속한 노선 리스트가 색상별로 나타남. Google 로그인 화면 <ul style="list-style-type: none"> Google 계정 로그인 페이지로 이메일 또는 전화번호 입력 후 Google 계정 인증을 진행. Kakao 로그인 화면 <ul style="list-style-type: none"> Kakao 계정 로그인 페이지로 ID(이메일, 전화번호)와 비밀번호를 입력하여 로그인. Naver 로그인 화면 <ul style="list-style-type: none"> Naver 계정 로그인 페이지로 자동 로그인 또는 ID/비밀번호 수동 입력을 통해 로그인. 		

결정사항 및 향후 일정	<ul style="list-style-type: none"> • 모듈 구현 (계속) • 발표 준비(피피티, 대본)
--------------------	--

일시	6 월 16 일	장소	pc 22 실
참석자	최수빈, 박정현, 정예슬, 최원희, 한연수		
회의 안건/목적	발표 및 피드백		
회의 내용	<ul style="list-style-type: none"> • 프로젝트 진행 방식 <ul style="list-style-type: none"> ◦ 진행 방식: 프로젝트를 어떤 방식으로 기획하고 개발했는지 설명 필요 ◦ 방법론: 폭포수 방식 사용 → 단계적 접근과 문서화 강조 ◦ 맨먼스(MM): 투입된 총 개발 인력과 기간을 정리하여 프로젝트 사이즈를 추정할 수 있도록 명시 필요 • 데이터베이스(DB) 관련 피드백 <ul style="list-style-type: none"> ◦ 정형 데이터: 지하철 역/노선 정보, 사용자 정보 등 → 구조화된 DB 사용 설명 (예: Firebase Firestore, Room DB) ◦ 비정형 데이터: 이미지, 사운드 등 멀티미디어 데이터 사용 여부 질문 있음 → 현재는 없음 or 향후 계획 언급 ◦ DB 구조 설명: 테이블 관계 및 주요 필드 중심으로 간단한 ERD 나 예시 포함해서 설명 • UI / 프론트엔드 관련 <ul style="list-style-type: none"> ◦ UI 디자인 방식: Figma, XML 등을 통한 화면 설계 흐름 설명 필요 ◦ 기술 스택 연계: 프론트(UI)와 백엔드(DB, API 등)가 어떻게 연결되는지 기술적 흐름 언급 ◦ 실제 화면 캡처/시연 영상 활용: 시각적 자료 강조 • 인공지능(AI) 사용 여부 <ul style="list-style-type: none"> ◦ AI 사용 여부: 실제 AI 모델은 미탑재 or 향후 적용 예정으로 정리 ◦ 사용 시 모델 설명: 사용할 경우, 어떤 알고리즘 기반인지(Python 모델 예시 포함) 명확히 • 전체 서비스 구조 설명 <ul style="list-style-type: none"> ◦ 모듈 구성: 기능별 모듈 명시 (예: 로그인, 하차알림, 혼잡도 조회 등) ◦ 앱 구조: 로컬 DB + Firebase + 공공 API + 사용자 단 UI 흐름으로 정리 ◦ 핵심 기능 위주 발표: 회원관리의 CRUD 구현 상세는 생략하고 핵심 기능 위주로 발표할 것 		

	<ul style="list-style-type: none"> ○ 보조자료 안내: 자세한 내용은 배포 자료 몇 페이지 참고하라고 안내 (예: “자세한 내용은 p.12 참조”) • 혼잡도 기능 설명용 참고자료 요약 <ul style="list-style-type: none"> ○ 혼잡도 수집 방식: 객차별 센서(승객 수 감지, 출입문 무게 감지) 기반 ○ 제공 채널: ‘또타 지하철’ 앱, 네이버/카카오 지도 (대중교통 탭), 역사 내 전광판, 열차 안내 디스플레이 ○ 데이터 기반: 고정 센서 + IoT 기반 센서 + AI 알고리즘 → 실시간 수요 반영 • 발표 시 유의 사항 요약 <ul style="list-style-type: none"> ○ 회원관리 CRUD 상세 구현 내용은 발표에서 제외 ○ 앱의 핵심 기능 중심으로 발표 (하차 알림, 혼잡도 분석 등) ○ 세부 기술 설명은 “자료집 몇 페이지 참조” 방식으로 전달 ○ 총 발표 시간은 15~20 분 이내로 조절할 것
결정사항 및 향후 일정	<ul style="list-style-type: none"> • 프론트엔드/백엔드 구현 • 피드백 받은 내용 보고서 수정

일시	7 월 15 일	장소	pc 22 실
참석자	최수빈, 박정현, 정예슬, 최원희, 한연수		
회의 안건/목적	백엔드 Firebase 연동		
회의 내용			
결정사항 및 향후 일정			

일시	7 월 20 일	장소	pc 22 실
참석자	최수빈, 박정현, 정예슬, 최원희, 한연수		
회의 안건/목적	방학 중 프로젝트 점검		

회의 내용	<ul style="list-style-type: none"> • 보고서 고치기 <ul style="list-style-type: none"> ◦ 개요, 아키텍처, 맨먼스 • 핵심적인 기능만 피피티에 넣기 • 회의 내용 정리하고 한 폴더에 모아놓기 • 서버 구축할건지 <ul style="list-style-type: none"> ◦ 파이어베이스 서버 마이에스큐엘 • 교수님이랑 지피티한테 물어봤는데 서버 없이 파이어베이스만으로 데이터베이스 구현해도 된다고 해서 • 일단 저번주에 정현이랑 만나서 백엔드는 파이어베이스 데이터베이스 생성까지 됐고 실시간으로 정보 업데이트 하는 거 하기로 했음 • 실시간 정보, 호선 파이어베이스에 연동 시키기 • 9월 말까지 프론트엔드 완성시키는 것으로 목표
결정사항 및 향후 일정	

일시	7월 28일	장소	pc 22실
참석자	최수빈, 박정현, 정예슬, 최원희, 한연수		
회의 안건/목적	인공지능 관련 정보 수집		
회의 내용	<ul style="list-style-type: none"> • 안드로이드와 인공지능의 융합 <ul style="list-style-type: none"> ◦ 안드로이드의 발전과 인공지능의 적용 <ul style="list-style-type: none"> - 안드로이드는 구글에서 개발, 기존의 패턴을 분석해 개선함 - 안드로이드 앱을 만들어줌으로써 사용자의 원하는 대로 코딩까지 가능하게 함 - 위자드 방식을 사용하여 사용자의 편리성을 높임 - 안드로이드가 위자드 방식을 사용하는 것은 대표적인 방식 중 하나임 ◦ AI의 발전과 '챗' 기술 <ul style="list-style-type: none"> - 챗은 AI의 대표적인 예시로, 사용자의 요청에 응답하는 기능을 제공함 - AI는 자연어 처리 기술을 사용하여 사용자의 요청에 맞는 응답을 제공함 - 챗의 발전에 따라 AI는 자연어 처리 기술을 넘어서, 인간의 언어를 이해하고 반응하는 수준에 도달함 ◦ 에이전트의 활용과 AI의 발전 		

- 에이전트는 사용자의 편리성을 높이기 위해 사용자와 상호작용하는 도구임
- 에이전트는 사용자의 성향을 파악하여 개인화된 서비스를 제공함
- AI는 에이전트의 기능을 확장하여 사용자의 편의성을 높이는 방향으로 발전하고 있음
- 시스템 요구사항
 - 시스템 요구사항 분석
 - 에이전트, 모델, 엠씨피 등 시스템 요구사항 공부 중심
 - 인공지능과 소프트웨어 개발의 트렌드에 관심을 가짐
 - 피그마와 M CP에 대한 이해가 필요함
 - 인공지능 기술과 소프트웨어 개발 능력은 필수 역량임
 - 요구사항 명세서 작성 시, 회원 관리, 빈자리 조회, 빈자리 공유 등의 주요 기능을 포함시켜야 함
 - 요구사항 축약과 구체화
 - 요구사항을 정확하게 축약하고, 추가 및 수정 사항을 고려함
 - 시스템 요구사항의 특성에 따라, 구체적인 문서 작성 시 줄을 두고 작성하는 것을 권장함
 - 교수님은 기능 구현의 시급성을 강조하심
 - 기능 축약 및 추가에 따라, 졸업 작품의 타이밍을 고려해야 함
 - 학교에서 배운 내용을 실제 프로젝트에 구현하는 것이 중요함
 - 과제 수행과 피드백
 - 과제는 '지하철 빈자리 공유' 플랫폼을 개발하는 것임
 - 현재 개발된 기능은 없었으나, 5월에 업그레이드 되었음
 - 경쟁 상품보다 기능이 늦어졌다고 평가함
 - 타이밍을 놓쳤다고 판단하고, 다른 방향으로 과감하게 변경했음
 - 학교에서 배운 내용을 실제 프로젝트에 적용하는 것이 중요함
- 앱 기획과 개발
 - 기존 앱과 차별점
 - 기존 앱과 유사한 점이 있어도 현실감 있게 접근해야 함
 - 9월에 경쟁 상품을 대상으로 새로운 기능을 추가함
 - 앱의 현실감 있는 이미지를 강조해야 함
 - 회원 관리가 가장 중요하며, 이 부분에 신선감이 없으면 안 됨
 - 소프트웨어 개발 시, 데이터베이스 설계를 철저히 해야 함
 - 회원 관리의 중요성
 - 회원 관리는 앱 개발에서 가장 중요한 부분 중 하나임
 - 데이터베이스 설계 시, 여러 형태의 데이터를 모두 고려해야 함
 - 여러 데이터를 구분하고, 이를 효율적으로 관리하는 방법을 알아야 함
 - 다양한 데이터를 수정하고 조회하는 등 데이터베이스의 기본적인 기능을 알아야 함

- 업데이트, 삭제, 변경 등 데이터베이스의 여러 단계를 이해해야 함
- 설계 시 고려 사항
 - 앱의 주요 기능을 설계한 후, 공유와 혼잡도 등도 고려해야 함
 - 다양한 기술적인 용어를 사용하지 말고, 서비스 위주의 용어로 전환해야 함
 - 여러 모듈에 대한 분석을 하고, 중요한 모듈 4 가지를 강조해야 함
 - 여러 모듈에 대한 분석을 통해 빈자리 공유, 혼잡도, 시간 분석 등 주요 기능을 설계해야 함
 - 서비스의 주요 기능을 강조한 설계가 되어야 함
- 캡스톤 설계 발표
 - 캡스톤 과제 진행 계획
 - 8 월 7 일에 코딩 발표를 위해, 8 월에 다시 만나면 설계서에 대한 의견을 전달받기로 함
 - 다음 주에 다다음 주에 만난 후 코딩에 대한 얘기를 나눔
 - 코딩에 대한 이야기가 끝나면, 9 월에 코딩에 대한 더 깊은 논의를 이어감
 - 방학이 끝난 후, 10 월에 코딩에 대한 발표가 완성되면 좋겠음
 - 코딩에 대한 발표는 10 월에 피티 모드로, 그 이후에는 프레젠테이션 모드로 전환될 예정임
 - 외부 인사 초청과 캡스톤 대회 준비
 - 캡스톤 대회에 외부 인사를 초청할 계획임
 - 캡스톤 교수님들은 학생들이 받은 상이 A+가 아니면, 발표를 잘해야 함을 강조함
 - 다른 캡스톤 교수님들은 상을 받으면 A+라고 함
 - 외부 인사도 초청해 캡스톤 대회를 준비하는 것에 대한 고민을 나눔
 - 10 월에 발표가 완성되면, 2 주 후에 발표를 다듬고, 2 월에 졸업 작품에 대한 교수님들의 피드백을 받음
 - 졸업 작품 준비와 캡스톤 대회 전략
 - 1 차 발표를 통해 2 차 발표를 준비하고, 2 주 후에 졸업 작품에 대한 피드백을 받음
 - 다른 교수님들과의 피드백을 통해 2 차 발표를 개선함
 - 12 월 초에 캡스톤 대회에 12 팀을 보내는 것에 대해 논의함
 - 캡스톤 대회에 여러 대회 출품을 목표로 함
 - 혼잡도에 대한 정의를 설명하고, 사용자 의견을 반영한 혼잡도 측정 방법을 소개함

결정사항 및 향후 일정	<ul style="list-style-type: none"> • 보고서 내용 *4 • OPAL 해보기 • 공모전 찾아보기
--------------------	---

일시	8 월 7 일	장소	pc 22 실
참석자	최수빈, 박정현, 정예슬, 최원희, 한연수		
회의 안건/목적	프로젝트 개요 명확히 정리, 피드백		
회의 내용	<ul style="list-style-type: none"> • 프로젝트 과정 <ul style="list-style-type: none"> ◦ 요구사항 분석 <ul style="list-style-type: none"> - 프로젝트 목표를 명확히 함 - 사용자의 요구를 정확히 이해하고 명확히 표현하는 것이 중요함 - 프로젝트 요구 사항을 정리하고, 요구사항 명세서와 기능 명세서를 1대1로 맞춤 - 분석 활동을 통해 요구사항을 정확히 이해하고, 5 가지 모듈을 생성함 ◦ 시스템 아키텍처 설계 <ul style="list-style-type: none"> - 5 가지 모듈을 포함한 전체 시스템 아키텍처를 설계함 - 회원 관리, 자료 공유, 사업 등 5 가지 모듈을 각각의 레벨에 맞게 설계함 - 레벨이 높은 레벨에 해당하는 모듈을 포함한 전체 설계를 5.1로 표현함 - 5.1에서 더 세부적인 기술적인 내용을 추가해야 함 - 설계 레벨이 5.7.1.1 까지 올라가야 함 ◦ 화면 설계 <ul style="list-style-type: none"> - 회원 관리, 홈 화면, 5 가지 모듈을 포함한 화면 설계를 진행함 - 화면 설계는 5 가지 모듈에 대한 화면 설계를 포함함 - 화면 설계도 5 개의 모듈을 포함해야 함 - 각 모듈에 대한 화면 설계도 5.1만큼 세부적인 기술적인 내용이 포함되어야 함 - 설계 레벨이 높은 레벨에 해당하는 모듈을 포함한 전체 화면 설계를 5.7.1.1로 표현함 • 프로젝트 설계 및 진행 방향 <ul style="list-style-type: none"> ◦ 프로젝트 요구사항 분석 및 설계 		

	<ul style="list-style-type: none"> - 프로젝트의 주요 기능을 찾고, 해당 기능에 대한 요구사항을 세부적으로 정의함 - 주요 기능에 대한 요구사항을 분석하고, 해당 요구사항을 통해 기능명세서를 생성함 - 기능명세서를 기반으로 화면 설계 및 데이터베이스 설계를 진행함 - 각 과정에서 역할 분담을 통해 효율적으로 작업을 진행함 <ul style="list-style-type: none"> ○ 프로젝트 일정 및 계획 <ul style="list-style-type: none"> - 11월 첫째 주에 졸업 심사가 진행되며, 이후 2주 후에 2차 심사를 진행함 - 12월에는 프로젝트가 거의 끝나며, 10월에는 발표를 통해 프로젝트 경험을 공유하고 면접 준비에 활용함 - 일정은 8월, 9월, 10월, 11월로 계획되어 있으며, 10월에는 코딩이 완성되도록 노력함 ○ 프로젝트 진행 방향 <ul style="list-style-type: none"> - 각 팀은 주어진 과제를 완성하는 데 집중하며, 요구사항을 정의하고 기능명세서를 작성함 - 프로젝트의 효율성을 위해, 코딩은 8월이나 9월에 시작하는 것을 고려함 - 각 팀은 개발 팀원들의 역할을 명확히 하며, 프로젝트 진행 상황을 주기적으로 보고함 - 8월 마지막에 만나 기존 문서의 5배 분석을 진행하고, 이후 다시 만나 일정을 조정함
결정사항 및 향후 일정	<ul style="list-style-type: none"> • 모듈 설계 정리 • 세부 기술 정리 • 5 가지 모듈 화면 구현&API 문제 해결

일시	8월 17일	장소	pc 22실
참석자	최수빈, 박정현, 정예슬, 최원희, 한연수		
회의 안건/목적	역할 분담 세분화		
회의 내용	<ul style="list-style-type: none"> • 5 가지 모듈별 역할 분담 및 담당 업무 <ul style="list-style-type: none"> ○ 혼잡도 시간대 분석 및 추천 <ul style="list-style-type: none"> - 담당: 원희 - 주요 업무: - 서울교통공사 데이터 수집 및 전처리 		

- 출퇴근 시간대별 혼잡 패턴 분석
- 분석 결과를 토대로 “추천 시간대” 기능 구현
- 기능 명세서와 DB 설계에서 관련 테이블 정의 (예: 시간대별 혼잡도 테이블)
- 이 역할은 서울교통공사에서 제공하는 공공데이터를 기반으로 출퇴근 시간대 혼잡 패턴을 분석하는 것입니다.
 - 단순히 데이터를 가져오는 것에 그치지 않고, 시간대별로 탑승객 수가 어떻게 달라지는지를 통계적으로 분석하여 이용자에게 더 쾌적한 시간대를 추천할 수 있도록 합니다.
 - 추가할 일:
 - 데이터 정제 및 시각화 (차트/그래프) → 보고서/발표 자료에 활용
 - “추천 근거”를 설명할 수 있는 자료 준비 (예: 8 시~9 시 혼잡률 80%, 10 시대 혼잡률 40%)
- 실시간 칸별 혼잡도 예측
 - 담당: 연수
 - 주요 업무:
 - 교통공사 API 연동 → 실시간 혼잡도 데이터 가져오기
 - 혼잡도를 시각적으로 표현 (예: 색상 변화)
 - UI 설계: 현재 시간대 + 혼잡도 표시 화면 제작
 - 테스트 케이스 설계 및 검증 (예: API 값이 NULL 일 때 처리)
- 이 모듈은 API 연동을 통해 실시간으로 열차 혼잡도를 불러오고, 칸별 비교를 가능하게 합니다.
 - 단순 예측뿐만 아니라, 시각적으로 “어느 칸이 여유 있는지” 색상 변화 등으로 보여줘야 합니다.
 - 추가할 일:
 - API 장애 상황 대비 → “데이터 없음”일 때 안내 메시지 처리
 - UI 프로토타입 제작 (칸별 색상 변화 화면)
 - 실시간 데이터 갱신 주기(예: 30 초마다 갱신) 설정
- 개인 맞춤형 탑승 제안
 - 담당: 예슬
 - 주요 업무:
 - 사용자 이용 패턴 학습 → 개인화된 시간·칸 추천
 - 하차 알림, 좌석 추천 기능 포함
 - 사용자 피드백 수집 및 반영 (모델 지속 개선)
 - 화면 설계: 개인화 추천 UI, 알림 기능 UX 설계
- 이 역할은 사용자의 이동 패턴을 학습해서, 개인에게 최적화된 시간과 칸을 추천하는 것입니다.

- 예를 들어, 평소 7호선을 이용해 9시에 탑승하는 사람에게 자동으로 “이번 주에는 8시 50분이 덜 혼잡합니다”라고 알려줄 수 있습니다.
- 또, 하차 알림, 좌석 추천 기능까지 포함되므로 UX 중심의 설계가 중요합니다.
- 추가할 일:
- 피드백 수집 기능 → 사용자 평가 반영 (추천이 도움이 되었는지)
- 알림 기능 구체화 (푸시 알림, 앱 내 알림 중 선택)
- 맞춤 추천 모델 정확도 비교 (초기 버전 vs 개선 버전)
- 데이터 분석 기반 예측 시스템 구축
 - 담당: 수빈
 - 주요 업무:
 - 공공데이터(승객 수, 열차 위치 등) 기반 예측 모델 개발
 - 머신러닝/딥러닝 적용 → 예측 정확도 향상
 - 모델 학습 및 평가 → 성능 비교 (예: Random Forest vs LSTM)
 - 데이터베이스와 예측 결과 연동 (예: 예측 테이블 저장)
- 이 모듈은 승객 수, 열차 위치 같은 공공데이터를 머신러닝/딥러닝으로 학습시켜 예측 모델을 만드는 핵심 역할입니다.
 - 단순 분석이 아니라 “미래의 혼잡도와 좌석 점유율을 예측”하는 게 목표입니다.
 - 추가할 일:
 - 여러 알고리즘 실험 (예: Random Forest, LSTM) → 최적 모델 선정
 - 정확도 지표 산출 (MAE, RMSE 등) → 발표 자료에 포함
 - DB에 예측 결과 저장 → 다른 모듈(추천/알림)이 활용 가능하도록 연동
- 탑승역 설정 후 UI 변화
 - 담당: 정현
 - 주요 업무:
 - 사용자가 탑승역/하차역 설정 시 UI 동적 변화 구현
 - 시간대별 색상 변화 기능 (혼잡도 시각화)
 - UI/UX 최종 점검 및 화면 설계 통합
 - 프론트엔드 개발 + 사용자 편의성 테스트
- 이 모듈은 사용자가 직접 탑승역과 하차역을 설정했을 때, 시간대별로 UI가 달라지는 기능을 맡습니다.

	<ul style="list-style-type: none"> - 예를 들어, 강남역에서 신도림역까지를 입력하면, 하차 시간대에 맞춰 혼잡도 색상이 변하거나 알림 메시지가 출력됩니다. - 추가할 일: <ul style="list-style-type: none"> ▪ UI/UX 테스트 (사용자 입장에서 보기 편한지) ▪ 색상 변화 규칙 정의 (예: 초록=여유, 노랑=보통, 빨강=혼잡) ▪ 전체 UI 흐름(홈 화면 → 설정 화면 → 결과 화면)을 연결
결정사항 및 향후 일정	<ul style="list-style-type: none"> • 필요한 자료 보충 • 모듈 화면 구현

일시	8 월 26 일	장소	pc 22 실
참석자	최수빈, 박정현, 정예슬, 최원희, 한연수		
회의 안건/목적			
회의 내용	<ul style="list-style-type: none"> • 보고서/문서 구성 관련 <ul style="list-style-type: none"> ◦ 100 페이지 채운 점: → 유지. ◦ 회의록 첨부(40~50 페이지): 좋은 아이디어 → 계속 포함. ◦ 로고 위치 조정: 보고서 앞부분(개요 근처)에 넣을 것. ◦ 구성 순서 재조정 필요: <ul style="list-style-type: none"> - 로고 → 프로젝트 개요 → 주요 설계/분석 → 코딩 → 테스트 → 회의록(마지막). ◦ 내용이 작아 잘 안 보이는 표·테이블: ◦ 썸네일 기법 활용 → 일부 확대해서 예시로 보여주기. ◦ 테이블 중간 일부만 잘라서 “데이터가 이런 식으로 구성된다”는 걸 보여주도록 수정. • 설계 부분 <ul style="list-style-type: none"> ◦ 현재: 데이터 설계, 화면 설계, 로직 설계, 기타 등등 8 개 정도. ◦ 수정: 5 개 절(챕터)로 재구성 <ul style="list-style-type: none"> - 데이터 설계 - 화면 설계 - 로직 설계 - API 설계 - 전체 아키텍처(위 설계 결과를 그림으로 표현) 		

- 시스템 명칭: ‘시스템 마케팅처’ 같은 생소한 명칭은 나중에 수정 필요.
- 테스트 관련
 - 테스트 케이스 시트(TC-001~015):
 - 현재 ‘PASS’ 표시 있음 → 실제 테스트 전이라면 빈칸으로 두어야 함.
 - 테스트 실행 후 PASS/FAIL 기입.
 - 앞부분에 Summary Table (테스트 케이스 개수, 범위 요약) 추가.
- 주요 기능/앱 이름
 - 앱 이름 ‘자리잇’: 설명 부족 → 이름만으로 이해 안 됨.
 - → 이름 + 부연설명 필요 (예시: “지하철 빈 좌석 찾기 & 하차 알림 앱” “개인 맞춤형 좌석 공유 및 알림 서비스” 등)
 - 제목 아래 1~2 줄 설명을 글자 크기 다르게 해서 3 줄 정도 배치.
- 인공지능/머신러닝 부분
 - 추후 개발 예정 기능 (현재 구현 X → 아이디어 제시 수준).
 - 약어 설명 필요:
 - ETL(Extract, Transform, Load) → 처음 등장 시 풀네임 제시 후 약어 사용.
 - ML(머신러닝), DL(딥러닝)도 마찬가지.
 - ROC, IQR, ARIMA, Prophet, XGBoost, Random Forest 등 → 풀네임+간단한 설명.
 - 모델 선택 설명 보강:
 - 단순히 “AI/ML 사용”이 아니라 어떤 모델을 선택했는지, 왜 이 모델을 썼는지 설명 필요.
 - 예: ARIMA = 시계열 예측, XGBoost = 성능 좋은 양상을 모델 → 이런 식.
 - Feature 생성 설명:
 - 현재 설명 부족 → 지하철 데이터에서 어떤 특징을 뽑아내는지 구체화 필요.
 - 품질 정의 보강:
 - 품질 = 데이터 정확도 (예: 빈자리인데 실제로는 아니었던 경우 → 품질 저하).
 - 실험 설계 설명:
 - 모델 학습 → 강화 학습 개념 → 데이터셋 변화 → 성능 비교 흐름 정리.
 - 기능 vs 성능 구분 강조:
 - 기능: 작동 여부
 - 성능: 속도/정확도
- 데이터베이스/SQL 부분
 - 현재 문제점: SQL 만 덩그러니 있어서 설명 부족.
 - 해야 할 일:
 - SQL 문에 대한 설명 추가 (예: 어떤 테이블, 어떤 목적).

	<ul style="list-style-type: none"> ○ ERD, 테이블 설명서 같이 제시. ● 기타 <ul style="list-style-type: none"> ○ 기대효과, 산출물 테이블, 프로젝트 일정: ○ 현재 뒷부분에 있음 → 보고서 앞쪽으로 이동. ○ 불필요한 기술(예: 중량 센서): → 우리 앱과 관련 없는 부분은 제외. ○ 수치 결과 제시 시 주의: <ul style="list-style-type: none"> ○ 근거 없는 수치는 공격받을 수 있음 → 숫자는 신중히 다듬기. ● 일정 및 발표 준비 <ul style="list-style-type: none"> ○ 다음 주 수요일: 교수님과 대면 → 자료 인쇄본/정리본 준비. ○ 9 월부터: 팀별 회의(교수님이 교시별로 시간 정해서 진행 예정). ○ 10 월 발표 연습: <ul style="list-style-type: none"> ○ 머신러닝, AI 모델 관련 용어·흐름 자유롭게 설명할 수 있도록 준비. ○ 발표 시 “기능 + 성능 + 모델 선택 이유”를 강조할 것.
결정사항 및 향후 일정	<ul style="list-style-type: none"> ● 회의록 추가 ● 자리잇 앱 이름 부연 설명 추가 ● 설계, 분석 챕터 재구성

일시	9 월 3 일	장소	pc 22 실
참석자	최수빈, 박정현, 정예슬, 최원희, 한연수		
회의 안건/목적	계획서 피드백		
회의 내용	<ul style="list-style-type: none"> ● 문서 구성 방향 <ul style="list-style-type: none"> ○ 색상·스타일 같은 부가 요소는 비중을 줄이고, 기능/설계 중심으로 재구성. ○ 문서 목차는 다음 순서 추천: <ul style="list-style-type: none"> - 아키텍처 - 데이터 설계 - 화면(UI) 설계 - 로직 설계 - 인터페이스(API) 설계 - 기타(테스트·운영) ● 세부 내용 정리 <ul style="list-style-type: none"> ○ 상세 설계: 데이터 정의 → 저장방식 → 화면 → 로직 → 인터페이스 순으로. ○ 스키마/SQL 코드: 너무 깊으니 요구사항이 아닌 설계 파트로 이동. 		

	<ul style="list-style-type: none"> ○ 테스트 케이스: 초반에 두지 말고, 뒤에 “테스트” 챕터에서 대표 시나리오 그룹핑. ○ 요구사항: ○ 주요 기능(탑승역·하차역 입력, 혼잡도 조회, 개인 맞춤형 추천, 공유)만 간단히. ○ ML(Random Forest 등)은 요구사항/분석 섹션에 간략하게. ○ API 키 발급 같은 기술적 절차는 개요가 아닌 인터페이스 설계에 배치. ● 교수님 피드백 포인트 <ul style="list-style-type: none"> ○ 너무 깊게 들어간 기술 설명은 뒷부분으로 내리기. ○ 회색 처리한 ML 조사 내용은 삭제하지 말고 분석/부록으로 이동. ○ 문서는 잘 작성됐으니 넘버링/챕터 체계만 정리하면 됨. ○ 기존 코드와 새로 생성한 코드 병합은 별도 프로젝트/브랜치에서 시도. ● 다음 할 일 <ul style="list-style-type: none"> ○ 문서 재구성 <ul style="list-style-type: none"> - 목차를 교수님 제안 순서로 재배치. - 색상/스타일 강조 부분 축소. - ML·통계 데이터는 요구사항/분석 챕터로 이동. - SQL, 테스트 케이스 등은 해당 챕터로 이동. - 넘버링(챕터/섹션/요구사항 코드) 정리. ○ 설계 보강 <ul style="list-style-type: none"> - 아키텍처 다이어그램 추가. - DB 스키마 → ERD 정리. - 화면 설계는 피그마 등으로 정리. - 로직/시퀀스 다이어그램 보강. ○ 구현/테스트 준비 <ul style="list-style-type: none"> - 안드로이드 APK 빌드 가이드 정리. - 대표 테스트 시나리오 그룹핑(탑승/하차, 혼잡도 조회, 개인화 추천, 공유). - 기존 코드와 새 코드 분리된 브랜치에서 병합 실험. ○ 데모 준비 (다음 회의 전) <ul style="list-style-type: none"> - 통합 문서 초안 PDF 제출. - APK 빌드본 1개 준비. - 필요 시 시연 영상/링크 확보.
결정사항 및 향후 일정	<ul style="list-style-type: none"> ● 문서 재구성(개요, 분석, 설계) ● 안드로이드 스튜디오 테스팅 ● 모듈 구현

일시	9 월 10 일	장소	pc 22 실
----	----------	----	---------

참석자	최수빈, 박정현, 정예슬, 최원희, 한연수
회의 안건/목적	
회의 내용	<ul style="list-style-type: none"> • 서비스 개발과 요구사항 분석 <ul style="list-style-type: none"> ◦ 서비스 개발 프로세스와 피드백 시스템 <ul style="list-style-type: none"> - 서비스 개발 프로세스의 시작은 '아키텍처 결정'으로 설명함 - 아키텍처 결정 후에는 '데이터 결정', '화면 결정', '로직 결정', '인터페이스 결정' 순서로 진행 - 이 과정에서 문제가 발생하면 피드백을 통해 개선을 하는 시스템을 구축함 - 개선 작업이 완료되면, '버전 관리'를 통해 최신 버전을 유지하며 서비스를 계속 개선함 ◦ 요구사항 분석과 순서 결정 <ul style="list-style-type: none"> - 서비스 개발 과정에서 요구사항 분석이 중요한 역할을 함 - 요구사항 분석을 통해 '기능명세서'를 생성하며, 이는 서비스 개선의 기반이 됨 - '분석에 포함된 요구사항'이 '기능명세서'의 기준이 됨 - 분석 결과에 따라 서비스 개선 작업을 진행하며, 이를 '버전 관리'를 통해 유지함 ◦ 기능명세서와 애플리케이션 <ul style="list-style-type: none"> - '기능명세서'는 서비스 개선의 결과로, 서비스의 주요 기능을 명확하게 정의함 - '인터페이스 결정'을 통해 사용자 인터페이스를 결정하며, 사용자 인터페이스는 사용자 인터페이스를 인터페이스로 만드는 작업임 - 서비스 개발 과정에서 '아키텍처', '데이터', '화면', '로직', '인터페이스', '저장 방식' 등을 결정함 - 이들 결정사항은 '기능명세서'를 통해 관리되며, 이는 서비스의 향상을 위해 필수적인 요소임 • 지하철 혼잡도 분석 및 시각화 <ul style="list-style-type: none"> ◦ 지하철 혼잡도 분석 도입 <ul style="list-style-type: none"> - 지하철 혼잡도를 분석하고, 분석 결과를 시각화하는 프로세스 설명 - 대상 노선과 시기를 설정하여 분석을 진행하고 결과를 표로 정리 - 시각화된 결과를 통해 혼잡도의 변동성을 파악하고 특정 노선에 대한 효과성을 평가

- 지하철 혼잡도 분석 결과 해석
 - 분석 결과를 통해 각 노선의 혼잡도 변동 사항을 파악
 - 1호선과 2호선에서 혼잡도 증가 현상 발생
 - 분석 결과에 따라 노선별 혼잡도 증가 추이 확인
 - 지하철 혼잡도 분석 결과 시각화
 - 분석 결과를 시각화하여 앱 사용자에게 명확한 정보 제공
 - 지하철역별 혼잡도 변동 사항을 한눈에 파악 가능
 - 분석 결과를 통해 앱의 사용자에게 필요한 정보와 시각화 자료 제공 방향 제시
- 화면 설계와 AI의 중요성
 - 화면 설계의 중요성
 - 혼잡도 분석의 결과를 바탕으로 화면 설계를 진행함
 - 다양한 색상으로 기차의 흐름을 시각화하면, 분석의 결과를 명확하게 이해하기 용이함
 - 실제 화면 설계 시, 사용자의 편의성을 고려해야 함
 - 사용자의 여행 패턴을 파악하여 자동차 시스템을 개선하는 것이 중요함
 - AI의 활용 방안
 - 혼잡도 분석 결과를 바탕으로 개인 맞춤형 제안을 제공해야 함
 - 사용자의 출퇴근 시간, 타는 역 등을 고려하여 사용자의 프로파일을 생성함
 - 인공지능을 통해 사용자의 이동 경로를 예측하고 개인화된 정보를 제공함
 - 사용자의 편의성을 위해 데이터를 학습하고 예측하는 과정이 중요함
 - AI의 필요성과 향후 전망
 - 데이터를 기반으로 사용자의 프로파일을 생성하고, 이를 바탕으로 개인화된 정보를 제공함
 - 머신러닝을 통해 사용자의 이동 경로를 예측하고, 이를 사용자에게 알려주는 시스템이 필요함
 - 사용자의 선호와 소비 패턴을 분석하여 자동차 시스템을 최적화하는 것이 중요함
 - 향후 AI 기술의 발전과 함께 사용자 중심의 AI 기술이 더욱 중요해질 것으로 예상됨
- AI 모델 개발
 - 프로젝트 소개
 - 프로젝트 주요 기능은 시간대, 칸, 개인 맞춤형, 머신러닝 분석 포함임
 - 기능별 세부 내용과 함께 각 모듈의 추진 사항을 설명함
 - 연구 사항을 강조하며 향후 변경 계획을 밝힘

- 프로젝트 기호를 분석 또는 설계 모듈로 변경 제안함
- 연구 사항의 중요성을 강조하며 관련 내용을 개선할 것을 제안함
- 기능 세부 사항
 - 기능별 주요 세부 사항을 강조하며 개선 사항을 제안함
 - 시간대 분석, 칸 분석, 개인 맞춤형, 머신러닝 분석을 포함한 다양한 기능을 설명함
 - UI 변화 수준과 함께 모듈의 개선 사항을 정리할 것을 제안함
 - 연구 사항의 중요성을 강조하며 관련 내용을 개선할 것을 제안함
 - 프로젝트의 추진 사항을 구체적으로 정리할 것을 제안함
- 사례 설명
 - 기능의 개선 사항을 구체적으로 제안하며 사례를 설명함
 - 시간대 분석, 칸 분석, 개인 맞춤형, 머신러닝 분석을 포함한 다양한 기능을 예시로 제시함
 - 회원 관리 기능의 중요성을 강조하며 관련 사항을 설명함
 - 연구 사항을 강조하며 관련 내용을 개선할 것을 제안함
 - 기능의 중요성을 다시 한번 강조하며 마무리함
- 시스템 아키텍처 설계
 - 요구사항 정리 및 기능명세서
 - 5 가지 모듈 만드는 것 가장 중요함
 - 기능명세서 끝나면 분석 시작함
 - 요구사항을 듣고 분석해 봄
 - 분석 활동의 결과로 설계를 하고 기능명세서로 넘어감
 - 분석을 마치고 요구사항을 기록해놓으면 나중에 확인할 수 있음
 - 시스템 아키텍처 설계
 - 시스템 아키텍처 설계를 그림으로 그려야 함
 - 테이블로만 시각화하면 이해하기 어려우니 그림으로 설명해야 함
 - 화면 설계를 할 때 스토리별로 재구성해서 설명해야 함
 - 스토리별 재구성 제안을 포함해 요구사항을 기록한 후, 테이블보다는 큰 그림으로 시각화해야 함
 - 개발하면서 매번 바뀌어도 버전을 유지해 두고 로그를 남겨야 함
 - 코드 작성 및 업데이트
 - 로그를 참고해서 코드를 수정해야 함
 - 코드 작성과 문제 해결을 위해 9 월에 발표 자료를 보내고, 10 월에 다시 만나 얘기하기로 함
 - 10 월에 발표 자료를 보내고 11 월에 다시 만나 얘기하기로 함

결정사항 및 향후 일정			
일시	9 월 17 일	장소	pc 22 실
참석자	최수빈, 박정현, 정예슬, 최원희, 한연수		
회의 안건/목적			
회의 내용	<ul style="list-style-type: none"> • 혼잡도 예측 <ul style="list-style-type: none"> ◦ 시스템 구현 문제점 <ul style="list-style-type: none"> - 1호선부터 8호선까지 경이선 지하철 구현 되었음 - 분당선 지하철 1호선부터 8호선까지도 구현 되었음 - 수인분당선과 수원분당선까지 연결되어야 함 - 환승은 차량이 여러 개일 경우 불가능함 - 환승 횟수를 고정해놓고, 혼잡도를 고려해 평균 혼잡도를 계산하는 방안 제안함 ◦ 환승 횟수와 혼잡도 <ul style="list-style-type: none"> - 환승 횟수에 따라 혼잡도 기반 예측이 달라짐 - 환승 횟수를 고정하고 혼잡도 예측을 기반으로 사용자에게 결과 제공하는 방안 제안함 - 사용자가 혼잡도에 민감하면 고려해줌 - 최단 거리와 최단 시간을 제공하는 것과 혼잡도 기반 예측을 함께 제공하거나, 기본 알고리즘을 사용해 최단 거리 위주로 제공할 수 있음 - 과도한 혼잡도 예측으로 인한 문제점 제시함 ◦ 시간표 제공의 어려움 <ul style="list-style-type: none"> - 역마다 시간표가 방대하므로 시간대에 따른 엇갈림 문제 있음 - 코레일은 차량 출발 시간을 예측해 제공함 - 시스템을 연동해 필요한 시간대의 시간표를 제공해야 함 - 최단 시간과 혼잡도 예측을 함께 제공하는 방안 제안함 • 지하철 앱 <ul style="list-style-type: none"> ◦ 시간 표와 혼잡도 <ul style="list-style-type: none"> - 열차 시간표를 제공하고, 기차 도착 시간을 정확히 알려주는 앱을 만들고자 함 - 사용자가 원하는 시간대에 지하철역을 이용할 수 있도록 시간 표를 제공하는 것이 중요함 		

- 시간대별 기차 도착 시간을 알고 싶은 사용자의 수요를 충족시키는 것이 목표임
- 지하철 앱은 기차 시간표와 혼잡도 정보를 제공하는 등 특화된 기능을 제공해야 함
- 시간표와 혼잡도 정보는 앱을 안보는 것이 아니라, 앱 내부에서 제공하도록 함
- 아키텍처와 코드
 - 안드로이드 아키텍처는 여러 파일, 프로그램 코드, 엑셀 파일 등을 포함함
 - 소프트웨어 및 서비스에서 중요한 기능만 살리고, 나머지는 보완하거나 수정함
 - 컨제스션 위크는 혼잡도를 파악하는 기능이며, 시간대별 혼잡도와 특정 기차에 대한 혼잡도를 나타냄
 - 아키텍처에서 중요한 것은 아키텍처의 컨제스션 위크, 타임 액티비티, 카, 코드, DB 등임
 - 코드 트리에 대한 코드 확인이 필요함
- DB 와 서버
 - DB 에는 회원 정보, 역 정보, 지하철 노선 정보 등이 저장되어 있음
 - 구글과 같은 기업의 서버는 파이썬파이브, 안드로이드의 단말기는 안드로이드 앱의 아키텍처에 포함됨
 - DB 에는 시스템, 사용자, 요금, 노선 정보 등이 저장되어 있으며, 이 정보는 시스템과 사용자 간에 교환됨
 - 서버는 안드로이드의 단말기와 연동되어 있지 않으며, 중간에 서버를 거치지 않음
 - DB 와 서버 간의 데이터 교환은 시스템, 사용자, 요금 정보 등을 포함함
- 교통정도 예측
 - 지하철역 반사장치
 - 지하철역 출발지 조교할 때 역전보 확인 가능함
 - 역전보 업데이트는 1년에 한 번 또는 2년에 한 번씩 함
 - 데이터는 DB 구축과 함께 회원 정보와 지하철 관련 정보로 구성됨
 - 출발지, 도착지, 최단 시간, 최단 거리, 최단 시간 등을 계산할 수 있음
 - 혼잡도를 기준으로 평균 혼잡도를 사용하여 최저 혼잡도를 보여줌
 - 머신러닝 예측
 - 머신러닝, 딥러닝 기반 예측을 강조함
 - 머신러닝과 딥러닝의 차이점을 알고 예측에 활용해야 함

- 기존에는 개입할 수 있는 수준의 머신러닝 방식이 있었으나, 이제 외부 기술의 도움이 필요함
- 두 방식 중 하나만 활용하거나 변경할 수 있음
- 기술 개발 방향
 - 현재 안드로이드에서 개발한 기존 방식을 이어받으면 됨
 - 서버 하나, 단말기, 서버 여러 개로 구성된 구조임
 - 간단한 구조이며, 라이브 테스트를 통해 결과를 확인할 수 있음
 - 모듈을 안드로이드 위에 올리면 완성된 프로젝트가 됨
 - 안드로이드에서 서울 교통공사로 돼 있는 것을 통해 팀을 나갈 때 필요한 파일들을 확인해야 함
- 빈자리 탐색
 - 빈자리 알림 기능 구현
 - 사용자 입력에 따라 기차의 현재 위치를 표시함
 - 입력된 정보가 기차의 현재 위치와 일치하면, 빈자리 정보를 제공함
 - 빈자리가 있는 경우, 해당 정보를 사용자에게 알림 창으로 제공함
 - 사용자가 빈자리의 위치를 알림 창에 표시하여, 쉽게 확인하도록 함
 - 사용자가 현재 위치를 알림 창으로 표시하여, 쉽게 빈자리의 위치를 확인하도록 함
 - 빈자리 탐색 알고리즘
 - 2 시 40 분에 해당 기차가 한양대역에 도착했을 때, 빈자리가 있을 경우 DB에 테이블로 생성함
 - DB에 테이블을 생성할 때, 기차의 출발 시간과 빈자리의 위치를 포함시킴
 - 빈자리가 있는 경우, 해당 자리는 '비어있다'고 표시함
 - 빈자리 탐색 알고리즘을 통해 사용자가 현재 위치를 알림 창으로 표시하도록 함
 - 테이블 설계 시, 빈자리의 개수에 따라 분할하여 테이블을 설계함
 - 빈자리 탐색 시각화
 - 사용자의 현재 위치를 알림 창으로 표시하여, 빈자리의 위치를 시각화함
 - DB에 테이블을 생성할 때, 기차의 출발 시간과 빈자리의 위치를 포함시킴
 - 빈자리가 있는 경우, 해당 자리는 '비어있다'고 표시함
 - 인공지능을 통해 빈자리의 오차를 줄이고, 시각화된 정보를 제공함

	<ul style="list-style-type: none"> - 사용자가 현재 위치를 알림 창으로 표시하여, 빈자리의 위치를 쉽게 확인하도록 함
결정사항 및 향후 일정	

일시	9 월 24 일	장소	pc 22 실
참석자	최수빈, 박정현, 정예슬, 최원희, 한연수		
회의 안건/목적			
회의 내용	<ul style="list-style-type: none"> • 좌석 공유 시스템의 개념과 개선 <ul style="list-style-type: none"> ◦ 좌석 공유 시스템의 개념 <ul style="list-style-type: none"> - 좌석 공유 시스템은 여러 사용자가 한 공간에서 동시에 이용할 수 있도록 함 - 사용자는 필요한 자리에 바로 앉을 수 있도록 시스템을 구성함 - 한 번만 이용할 수 있는 좌석은 복잡성과 책임감을 줄 수 있음 - 여러 차량이 동시에 이용할 수 있는 시스템은 탑승과 하차 시 별도의 조정이 필요함 - 좌석 공유 시스템은 기본적으로 한 번만 이용 가능한 자리를 제공함 ◦ 좌석 공유 시스템의 장점과 사용자 의견 반영 <ul style="list-style-type: none"> - 좌석 공유 시스템은 환승 시점에 대한 애매함을 해결해줌 - 여러 차량이 이용할 수 있어 효율성과 편의성을 제공함 - 하지만, 일부 사용자들은 좌석 공유 시스템의 사용을 꺼릴 수 있음 - 예를 들어, 한 번에 모든 차량을 이용하려는 사용자가 있음 - 이러한 사용자 의견을 반영하여 시스템을 개선할 필요가 있음 ◦ 좌석 공유 시스템의 구현과 향후 개선점 <ul style="list-style-type: none"> - 현재 좌석 공유 시스템은 POC 단계이며, 컨셉만 구현됨 - 사용자의 의견을 반영하여 시스템을 개선하는 것이 중요함 - 사용자들의 선호도와 기호를 반영하여 사용자 맞춤형 시스템을 제공함 - 예를 들어, 많은 사용자가 특정 차량을 이용하는 경향을 고려할 수 있음 - 이러한 개선점들을 통해 시스템의 편의성과 사용자 만족도를 높일 수 있음 		

	<ul style="list-style-type: none"> ● 기차 환승 시스템 설명 <ul style="list-style-type: none"> ○ 환승 시스템 설명 및 문제점 <ul style="list-style-type: none"> - 환승 시스템에서는 첫 번째 기차의 노선을 보여주고, 그 다음 기차로의 화면을 설계함 - 각 기차의 도착 시간을 더해 시작 시간을 보여주는 방식을 사용함 - 화면 설계시에는 경우의 수가 많아 고민이 필요하며, 단계적으로 접근하는 것이 중요함 - 환승 시스템이 제대로 구현되지 않을 경우, 시간표와 API를 검증해야 함 ○ 공공 API 의 활용 <ul style="list-style-type: none"> - 공공 API 를 활용하여 시간표와 도착역을 확인하는 시스템을 설명함 - 5 시 이후에 출발하는 기차에 대한 정보를 API 를 통해 확인할 수 있음 - 이 때, 특정 기차에 대해 여유 여부를 물어보는 API 를 활용함 - 해당 API 를 통해 혼잡한 기차인지, 아니면 여유 있는 기차인지 파악 가능함 ○ 환승 가능 여부 판단 <ul style="list-style-type: none"> - API 를 통해 현재 혼잡한 기차인지, 아니면 여유 있는 기차인지 확인함 - 해당 기차가 혼잡한 경우, 해당 정보를 무시하고 다음 기차로 넘어감 - 선택한 기차에 탄 후, 빈자리에 대한 정보를 확인하여 다시 환승 가능성을 검토함 - 이 과정을 통해 7077 기차에 탔음을 확인하고, 해당 정보를 다시 표시함
결정사항 및 향후 일정	

일시	10 월 1 일	장소	pc 22 실
참석자	최수빈, 박정현, 정예슬, 최원희, 한연수		
회의 안건/목적			

회의 내용

- 프로젝트 일정: 작업은 10 월 내 마무리 목표, 발표는 11 월 셋째 주 예정.
- 현재 구현 현황
 - 완료/가까운 완료: 혼잡도(칸별), 시간표, 도착정보(실시간) — SK TMAP API 연동 중/부분 구현.
 - 미완/진행 중: 좌석 배치도(한 량 내 좌석 표시), 환승 단계에서의 좌석 선택 흐름.
- 환승 시간 데이터 소스
 - 공공데이터포털의 환승 거리/소요시간은 CSV(파일 제공, 비실시간) 형태.
 - 당장 실시간 API 가 없다면 DB(Firebase)에 적재 후 경로 계산 시 가산하는 방식으로 사용.
 - 임시 방안: 전역 평균 1분 51초를 환승가산 시간으로 적용 → 추후 역별 데이터로 정교화.
- 좌석 모델링
 - 실제 편성/차종별 좌석 수는 복잡 → MVP로 ‘한 량 = 50 석(가정)’ 등 단순화하여 표시.
- UX 흐름(환승)
 - 이상적: 출발 구간 좌석 선택 → 환승역 도착 시 다음 구간 좌석 자동 제안(환승시간 고려).
 - 임시: **구간 분할 선택(출발→환승 / 환승→도착 두 번 선택)**로 먼저 제공.
- 멘토링
 - 회사 측 백엔드 중심 멘토 1명 연결 가능성 있음(확답 전).
 - 평일 19 시 이후 Zoom 중심 원격 지원 + 사전 질문 메일 공유 방식 선호.
- 세부 논의 정리
 - 공공데이터포털 접속 이슈는 일시적(보안/장애) 가능성. 실시간 미지원이므로 CSV 기반 정적 참조가 전제.
 - 환승 예시(성수→건대입구→7 호선): 건대입구 2→7 환승 77m / 1분 4 초 같은 레코드가 CSV에 존재 → 이런 값들을 DB에 넣어 경로 시간에 +환승가산.
 - 우선순위 전략: 앱의 핵심 목적이 빈좌석 파악/표시이므로 좌석 관련 기능을 최우선. 환승은 평균치 가산으로 먼저 불이고, 여력이 생기면 역별 데이터 적용.
 - 팀 구성(요지): 2 명 좌석 UI/XML, 2 명 프런트 좌석/혼잡도연동, 1 명 문서/정리. 주 1회 수요일 대면/회의, 나머지는 각자 진행.
- 데이터/기술 메모
 - 실시간: 혼잡도·도착·시간표는 SK TMAP 사용(이미 연동 중).
 - 정적 환승 시간: 공공데이터 CSV → 가공 → Firebase 적재.

- 경로 계산: (구간 주행시간 합) + (환승가산 시간들) → 다음 열차 선택 타이밍에 반영.
 - 좌석 표시: MVP는 고정 좌석수로 시작, 이후 차종/편성 반영 여지.
- 다음 할 일 (실행 체크리스트)
 - 환승 데이터 정비
 - 공공데이터 환승 CSV 수집/정제(컬럼: 출발역/노선, 도착역/노선, 거리(m), 소요(s)).
 - Firebase 스키마 설계: /transfer_times/{stationId}/{fromLine}-{toLine} → {distance, seconds, updatedAt}.
 - 배치 적재 스크립트 작성(한 번 업로드 후 필요 시 수동 갱신).
 - 경로 계산 로직에 환승가산(초 단위) 적용. 초기값은 전역 111초(1:51) 옵션 제공 → 역별 데이터 존재 시 우선 사용.
 - 좌석 기능(MVP)
 - 좌석 데이터 모델 확장: cars(편성수), seatsPerCar(기본 50), seatId, status(빈/예약/혼잡추정) 등.
 - 좌석 배치도 XML 1안 확정(그리드/플렉스) 및 CarOnLyAdapter 연동.
 - 칸별 혼잡도↔좌석 UI 연동: 혼잡도(혼잡/보통/여유)를 좌석 색/투명도 규칙으로 매핑.
 - 구간 분할 선택 임시 플로우 구현: 출발→환승, 환승→도착을 두 단계 선택으로 제공.
 - API/연동 안정화
 - TMAP 엔드포인트/쿼터/오류 케이스 점검 체크리스트 작성(타임아웃/리트라이/스로틀링).
 - 공공데이터포털 접속 이슈 재확인(장애 해소 후 재다운로드, 해시로 버전관리).
 - 문서/발표/커뮤니케이션
 - 현재 산출물(요구·기능명세, 아키텍처, 데이터 설계, UI 와이어, 테스트 계획) 패키징 후 이메일 전달.
 - 우선순위 표 확정(좌석 표시 1순위 → 환승 평균적용 → 역별 환승 정교화).
 - 멘토링 운영안 확정:
 - 질문 리스트를 미리 메일로 송부(백엔드: DB 스키마, API 캐싱, 경로계산 구조).
 - 주 1회 평일 19 시 이후 Zoom 40-60 분 진행(필요 시 수시 Q&A).
 - 타임라인 업데이트:
 - 10 월 2~3 주차에 MVP 동작 데모(좌석 표시+환승 평균 적용), 10 월 말 기능 얼 freeze → 11 월 셋째 주 발표 리허설.
 - 리스크/대안

	<ul style="list-style-type: none"> ○ 공공데이터 자연 시: 평균 환승시간 계속 사용 + 주요 환승역 Top 20 만 수작업 등록. ○ 16. 좌석 혼잡도(차종/편성) 이슈: MVP 유지, 사용자 피드백 수집 후 2 차 반영 로드맵에 기재.
결정사항 및 향후 일정	

일시	10 월 15 일	장소	pc 22 실
참석자	최수빈, 박정현, 정예슬, 최원희, 한연수		
회의 안건/목적	교수님 피드백 및 발표/영상 개선 방향		
회의 내용	<ul style="list-style-type: none"> • 데모 영상 및 발표 구성 관련 <ul style="list-style-type: none"> ○ 현재 영상 문제점: <ul style="list-style-type: none"> - 영상이 너무 조용하고, 설명이 부족해 관객이 기능을 따라가기 어렵다는 피드백. - 해결방안: 영상 내에 설명 음성 또는 자막(서브타이틀) 을 추가하여 기능 흐름을 명확히 전달할 것. ○ 앱 기능 구성: <ul style="list-style-type: none"> - 발표에서 강조한 주요 기능은 ①혼잡도 조회 ②하차 알림 ③좌석 공유(정보 공유) 등 3 가지임. - 데모 영상에 이 3 가지 기능이 모두 시나리오별로 포함되어야 함. - 현재는 일부만 표현되어 있으므로, 각 기능별 시연 장면을 분리하여 명확히 보여줄 것. ○ 발표 진행 방식 제안: <ul style="list-style-type: none"> - 기능별로 구간을 나누어 “첫 번째 기능은 혼잡도입니다 → 영상 시연 → 잠시 멈춤 후 두 번째 기능으로 전환” - 이렇게 발표자 멘트와 영상 전환을 섞어 구성하면 전달력이 높아짐. • 영상 및 발표 전달 개선 <ul style="list-style-type: none"> ○ 온라인 제출용 영상의 경우 음성 설명이 불가하므로, 하단 자막 또는 텍스트 설명을 직접 삽입해야 함. <ul style="list-style-type: none"> - (예: “현재 화면은 칸별 혼잡도 실시간 분석 기능을 보여줍니다.” 등) • 데이터 관련 질의 및 설명 보강 		

	<ul style="list-style-type: none"> ○ 수인선·서해선 데이터 미지원 문제: <ul style="list-style-type: none"> - 네이버 지도 등에서는 보이지만, 팀이 사용하는 공공데이터(API)에서는 정상적으로 제공되지 않음. - 발표 시 단순 오류로 넘기지 말고, “데이터 연동 과정에서 특정 노선은 정보 제공이 제한되어 있었다”는 점을 공공데이터 품질 한계 사례로 설명하면 좋음. ○ 교수님 조언: <ul style="list-style-type: none"> - “기술적 이유로 특정 노선이 제외된 과정도 하나의 분석 내용으로 전개하라.” - 예를 들어, “1~8 호선, 신분당선은 연동이 원활했으나 수인선과 서해선은 동일 API임에도 응답이 누락되었습니다. 이를 통해 공공데이터의 제공 범위가 아직 완전하지 않음을 확인했습니다.” 이런 식으로 발표에서 문제 인식과 개선 의지를 함께 보여줄 것. ● 발표 전개 및 구성 원칙 <ul style="list-style-type: none"> ○ MEC(Mutually Exclusive, Collectively Exhaustive) 원칙 강조: <ul style="list-style-type: none"> - 기능 간 중복 없이, 빠진 부분 없이 전체를 포괄적으로 설명하라. - (예: 혼잡도-하차 알림-공유 기능이 각각 독립적이면서도 앱 전체 구조를 완성하는 방식으로 구성) ● 교수님 총평 <ul style="list-style-type: none"> ○ 전체적으로 구현 및 내용은 잘 진행되었음. ○ 발표 구조와 영상 전달 방식을 보완하면 완성도 높은 발표가 될 것. ○ “기술 구현 + 공공데이터 활용 한계 인식 + 개선 방향” 이 세 가지를 균형 있게 담을 것.
결정사항 및 향후 일정	