11주차 진행 과정

요구 사항을 더 수집하였다.

**일정 입력이 번거로움**

일정 추가 시 너무 많은 정보를 입력해야 해서 귀찮음

빠르게 일정 추가할 수 있는 기능 부족 (예: 간단한 메모 방식)

**알림 설정의 불편함**

중요한 일정이 있는데 알림이 제대로 오지 않음

사용자 맞춤 알림 설정이 부족 (반복 알림, 특정 조건 알림 등)

**일정 검색 & 관리 어려움**

일정이 많아지면 원하는 일정 찾기가 힘듦

태그나 카테고리 기능이 부족하여 일정 분류가 어렵고 관리가 불편함

일정 중복 확인 기능이 없어 동일한 이벤트를 여러 번 입력할 가능성 있음

**UI 및 사용성 문제**

너무 복잡한 인터페이스로 인해 필요한 기능을 찾기가 어려움

모바일과 데스크톱 간 UI 차이가 커서 활용성이 떨어짐

색상이나 테마 변경이 불가능해서 사용자 맞춤 설정이 부족

**다른 사람과 일정 공유 어려움**

회의나 모임 일정을 공유하기 어려움

일정 초대 기능이 제한적이고 상대방이 확인했는지 알기 어려움

오늘 한 것.

Pubspec.yaml에 캘린더 기능(table\_calender) 및 gps구현 기능(geolocator)을 임포트해옴.

캘린더 화면을 출력하는 코드 생성

import 'package:flutter/material.dart';  
import 'package:table\_calendar/table\_calendar.dart';  
import 'package:intl/date\_symbol\_data\_local.dart';  
  
void main() async {  
 WidgetsFlutterBinding.*ensureInitialized*();  
 await initializeDateFormatting('ko\_KR', null);  
 runApp(MyApp());  
}  
  
class MyApp extends StatelessWidget {  
 @override  
 Widget build(BuildContext context) {  
 return MaterialApp(  
 home: Scaffold(  
 appBar: AppBar(title: Text('캘린더 앱')),  
 body: TableCalendar(  
 firstDay: DateTime(2000, 1, 1),  
 lastDay: DateTime(2030, 12, 31),  
 focusedDay: DateTime.now(),  
 locale: 'ko-KR',  
 // 한글 지원  
 onDaySelected: (selectedDay, focusedDay) {  
 // 날짜 선택 시 처리  
 },  
 calendarStyle: CalendarStyle(  
 todayDecoration: BoxDecoration(  
 color: Colors.*blue*,  
 shape: BoxShape.circle,  
 ),  
 selectedDecoration: BoxDecoration(  
 color: Colors.*red*,  
 shape: BoxShape.circle,  
 ),  
 ),  
 ),  
 ),  
 );  
 }  
}

사용자의 위치를 파악 후 해당 위치에 따른 알림 전송 기능 구현

import 'package:flutter/material.dart';  
import 'package:geolocator/geolocator.dart';  
  
void main() {  
 runApp(const MyApp());  
}  
  
class MyApp extends StatelessWidget {  
 const MyApp({super.key});  
  
 @override  
 Widget build(BuildContext context) {  
 return MaterialApp(  
 home: LocationTracker(),  
 );  
 }  
}  
  
class LocationTracker extends StatefulWidget {  
 @override  
 \_LocationTrackerState createState() => \_LocationTrackerState();  
}  
  
class \_LocationTrackerState extends State<LocationTracker> {  
 Position? \_currentPosition;  
 final double targetLatitude = 37.7749; // 예시 장소 (위도)  
 final double targetLongitude = -122.4194; // 예시 장소 (경도)  
  
 void \_getCurrentLocation() async {  
 Position position = await Geolocator.*getCurrentPosition*(  
 desiredAccuracy: LocationAccuracy.high);  
 setState(() {  
 \_currentPosition = position;  
 });  
  
 // 특정 장소 도착 여부 확인  
 double distance = Geolocator.*distanceBetween*(  
 \_currentPosition!.latitude, \_currentPosition!.longitude,  
 targetLatitude, targetLongitude);  
  
 if (distance < 50) { // 50m 이내 도착 시 출력  
 \_showEventPopup();  
 }  
 }  
  
 void \_showEventPopup() {  
 showDialog(  
 context: context,  
 builder: (context) => AlertDialog(  
 title: const Text("도착 알림"),  
 content: const Text("해당 장소의 일정: 회의 @ 15:00"),  
 actions: [  
 TextButton(  
 onPressed: () => Navigator.*of*(context).pop(),  
 child: const Text("닫기"),  
 )  
 ],  
 ),  
 );  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 appBar: AppBar(title: const Text("GPS 기반 일정 알림")),  
 body: Center(  
 child: Column(  
 mainAxisAlignment: MainAxisAlignment.center,  
 children: [  
 Text(\_currentPosition != null  
 ? "현재 위치: ${\_currentPosition!.latitude}, ${\_currentPosition!.longitude}"  
 : "위치 정보를 가져오는 중..."),  
 ElevatedButton(  
 onPressed: \_getCurrentLocation,  
 child: const Text("현재 위치 확인"),  
 ),  
 ],  
 ),  
 ),  
 );  
 }  
}