

캡스톤 디자인 **10.26 진도 발표**

발표자 : 김가희

팀장 : 김현성

팀원 : 김가희, 정영훈

Bicycle Application

자전거 네비게이션

자전거 네비게이션 기능을 탑재한

자전거 전용 애플리케이션



Bicycle Application

목차

01

UI

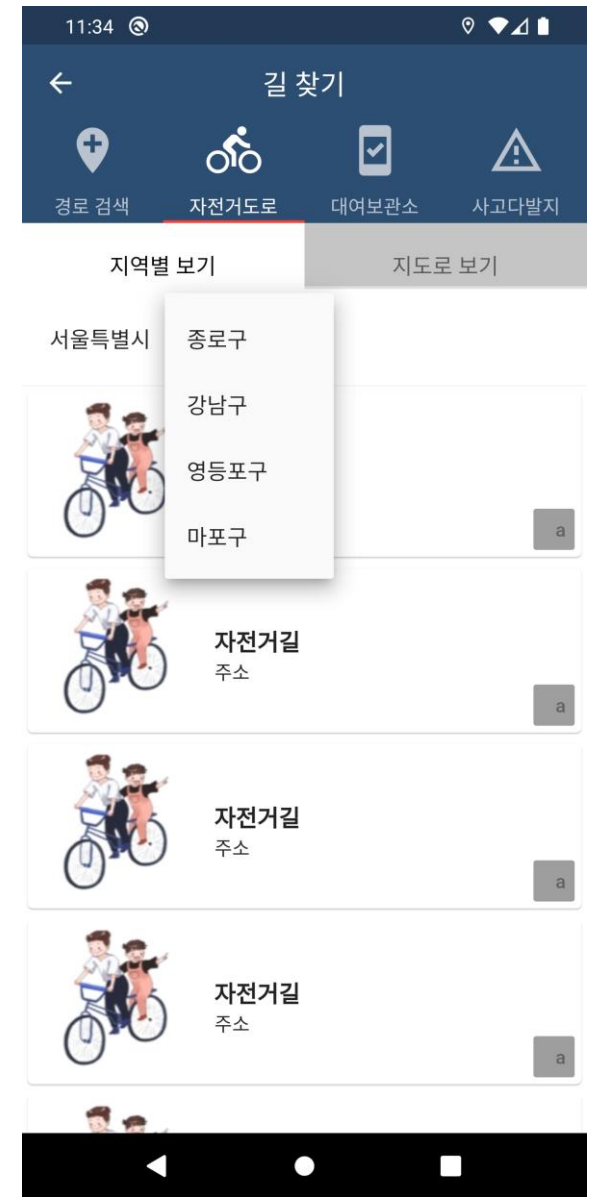
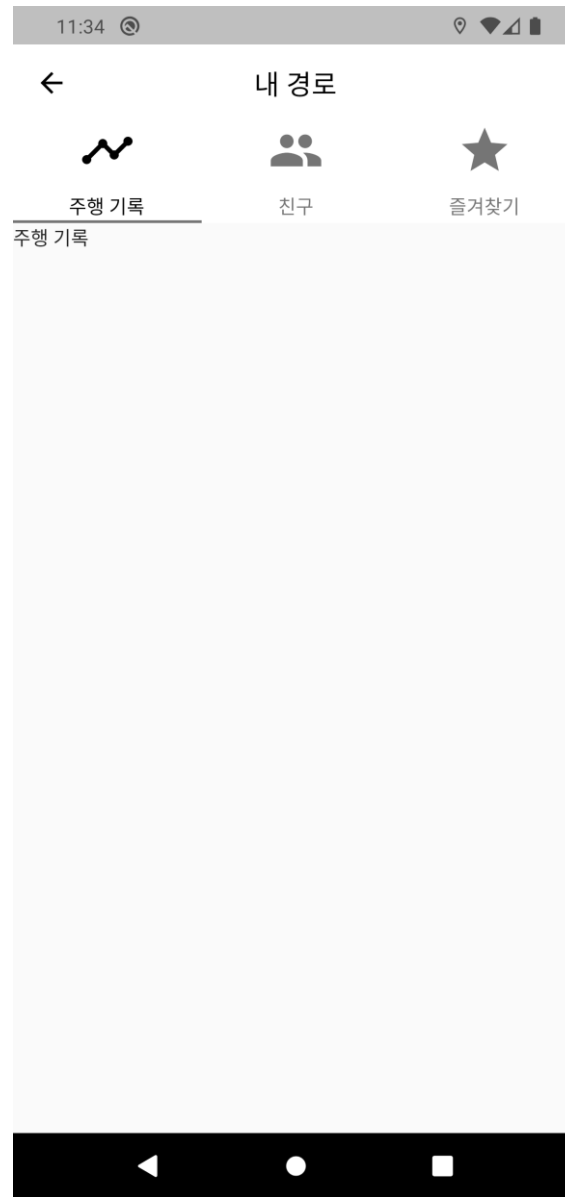
02

대여소, 보관소

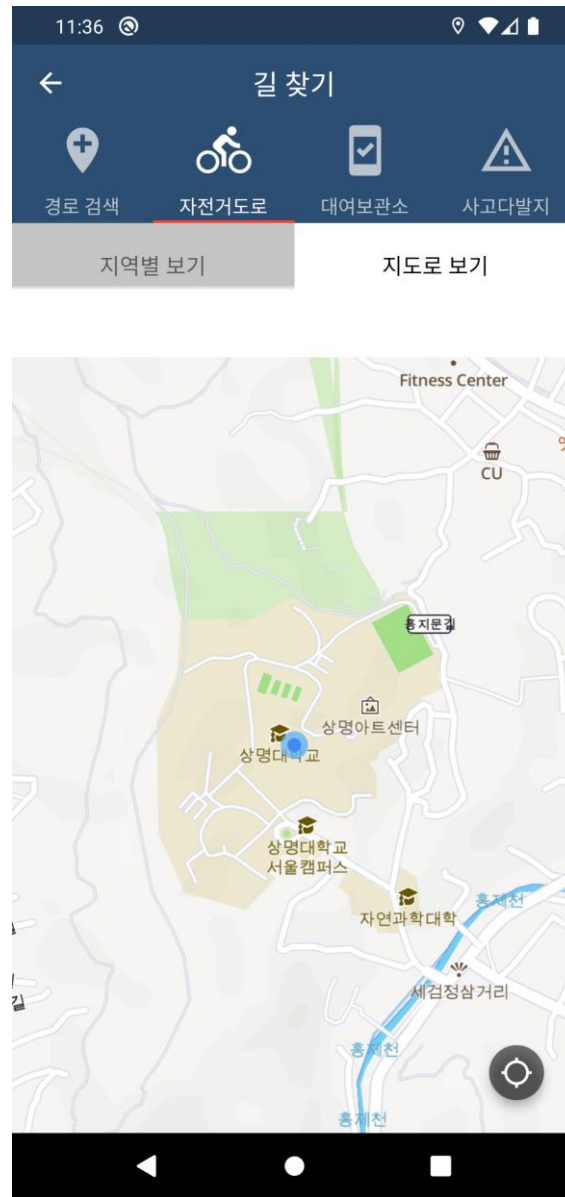
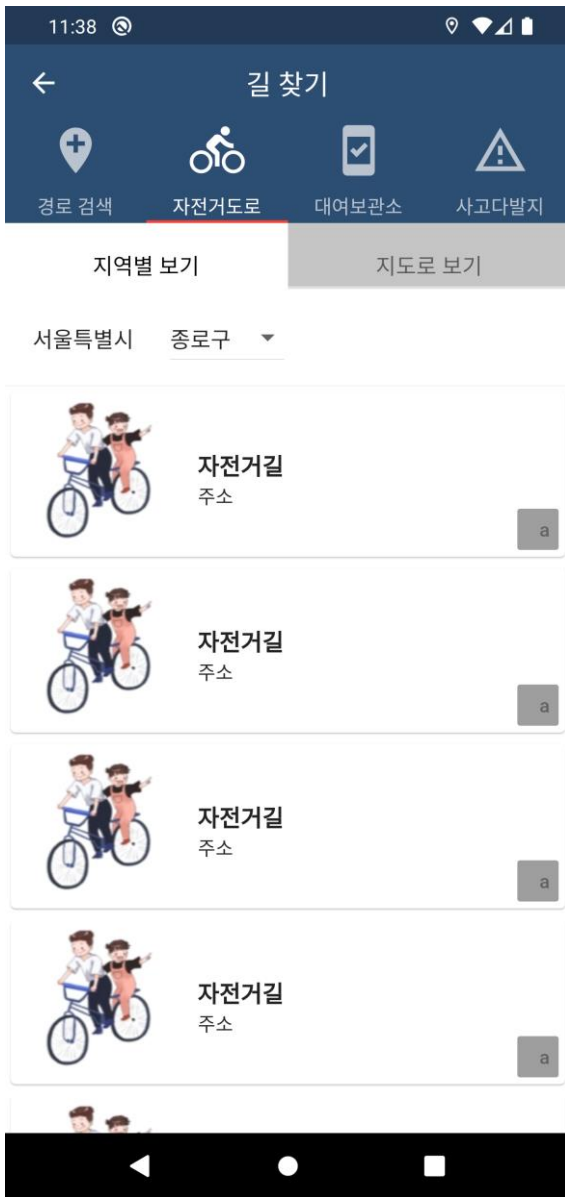
03

사고 다발 지역

UI

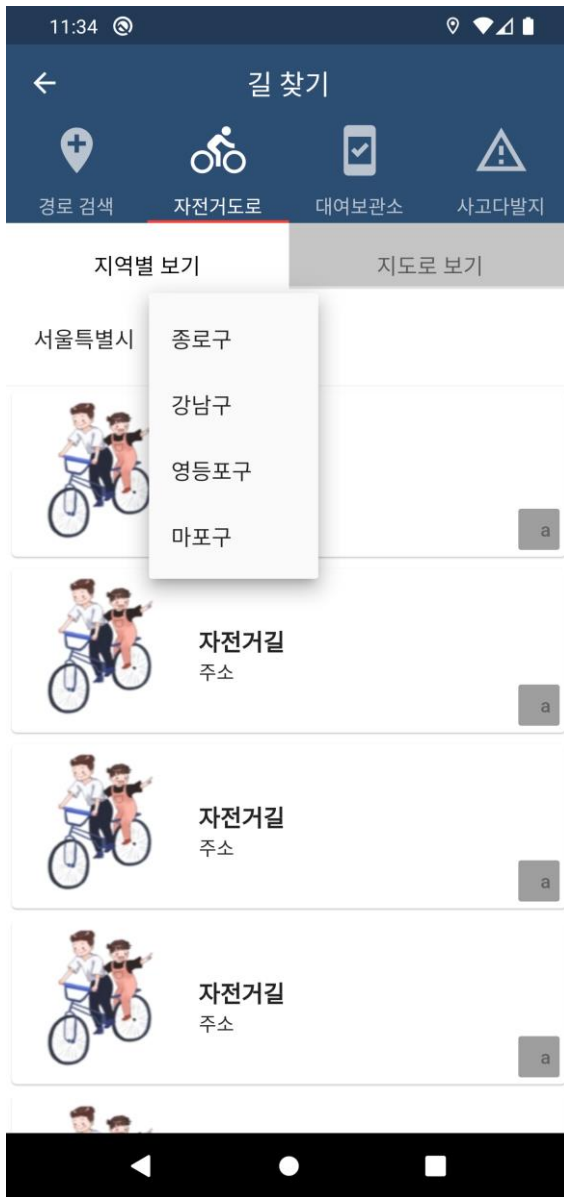


자전거 도로 UI



```
children: <Widget>[
  Material(
    color: Colors.black12,
    child: TabBar(
      unselectedLabelColor: Colors.black54,
      labelColor: Colors.black,
      indicatorColor: Colors.white,
      controller: _tabController,
      labelPadding: const EdgeInsets.all(0.0),
      tabs: [
        _getTab(0,
          Text("\n지역별 보기",
            textAlign: TextAlign.center,
            style: TextStyle(fontSize: 16.0))), // Text
        _getTab(1,
          Text("\n지도로 보기",
            textAlign: TextAlign.center,
            style: TextStyle(fontSize: 16.0))), // Text
      ],
      onTap: _onTap,
    ), // TabBar
  ), // Material
  Expanded(
    child: TabBarView(
      physics: NeverScrollableScrollPhysics(),
      controller: _tabController,
      children: [
        Road1(),
        Road2(),
      ],
    ),
  ),
]
```

자전거 도로 세부 메뉴



대여소, 보관소 CSV 데이터

map_210521 D:\map_210521

> .dart_tool

> .idea

> android

> Assets

data

AccidentData.json

park_seodaemun.csv

rentalSeoul.csv

> Icon

> Image

> sprite_images

license.txt

pin.png

style.json

> build

> image

> ios

> lib

config

search

Road

SeoulRoad

GangNam.dart

JongRo.dart

Mapo.dart

Yeongdeungpo.dart

Road1.dart

Road2.dart

Accident.dart

bicycleRoad.dart

MapStorage.dart

place.dart

popup.dart

popup_ac.dart

search.dart

searchRoute.dart

Storage.dart

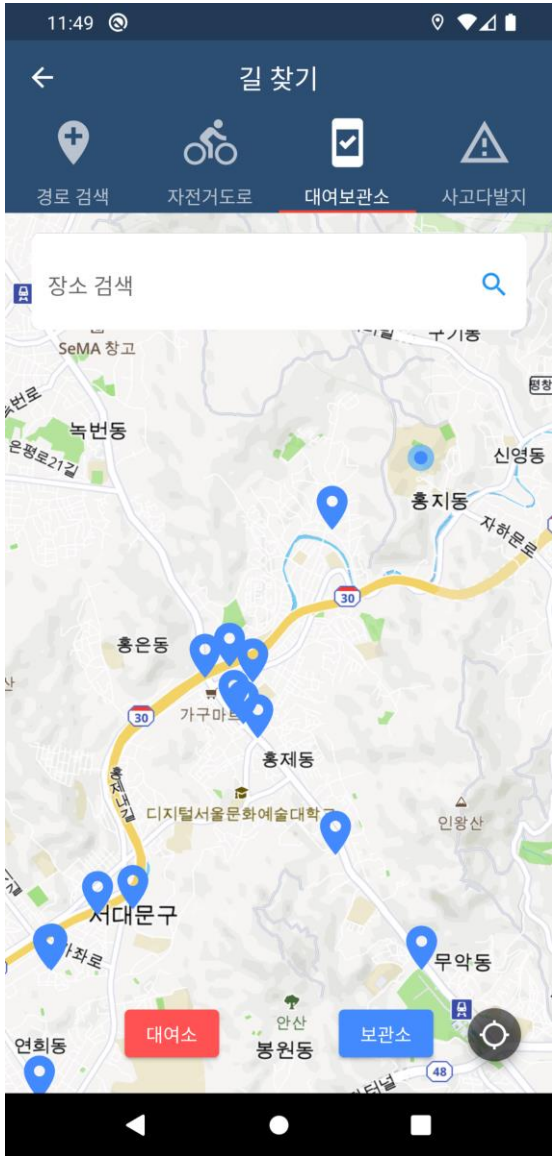
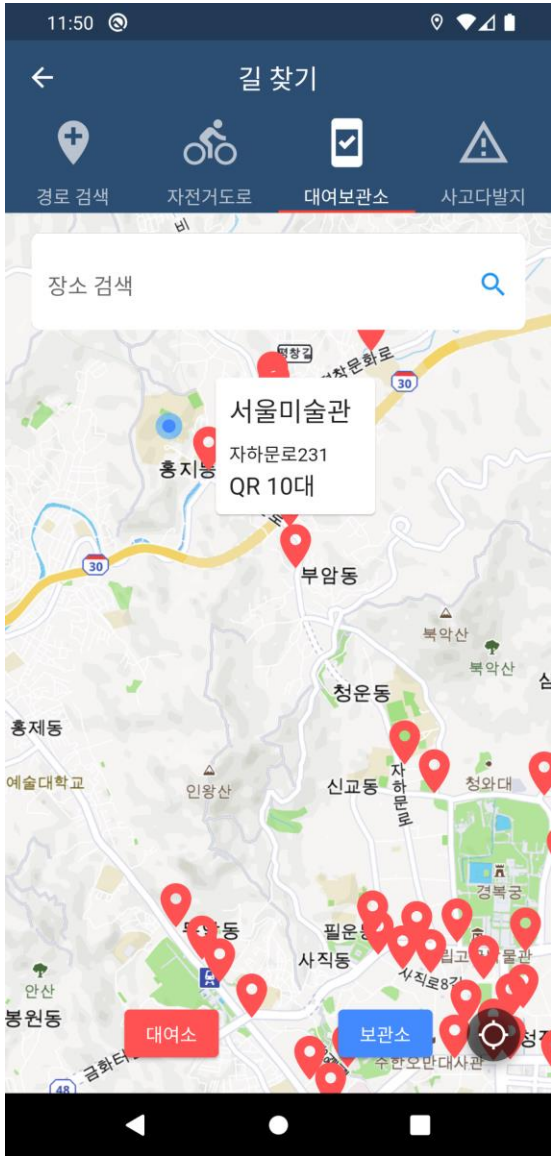
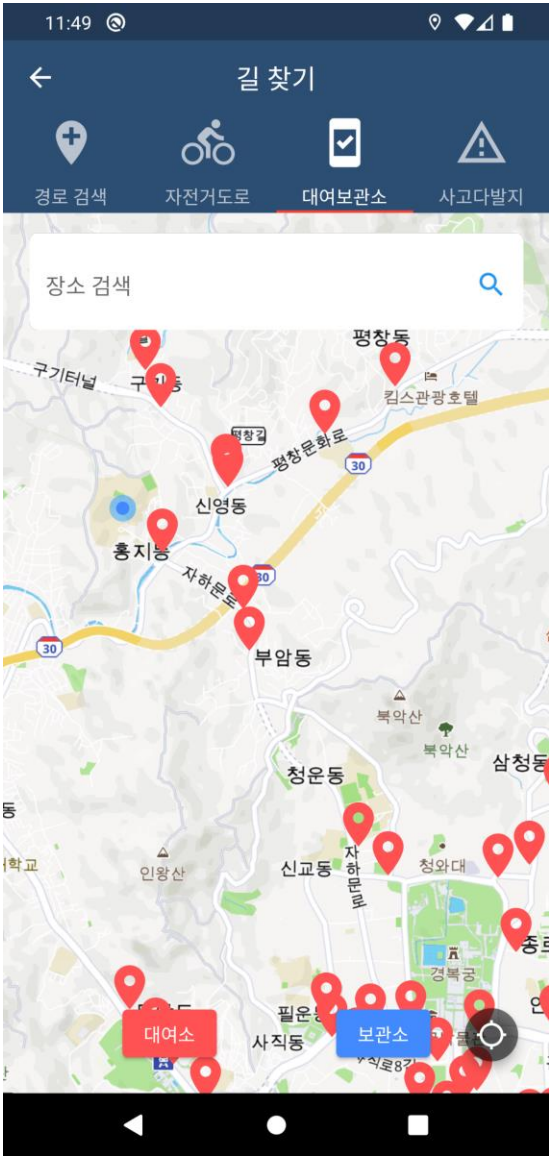
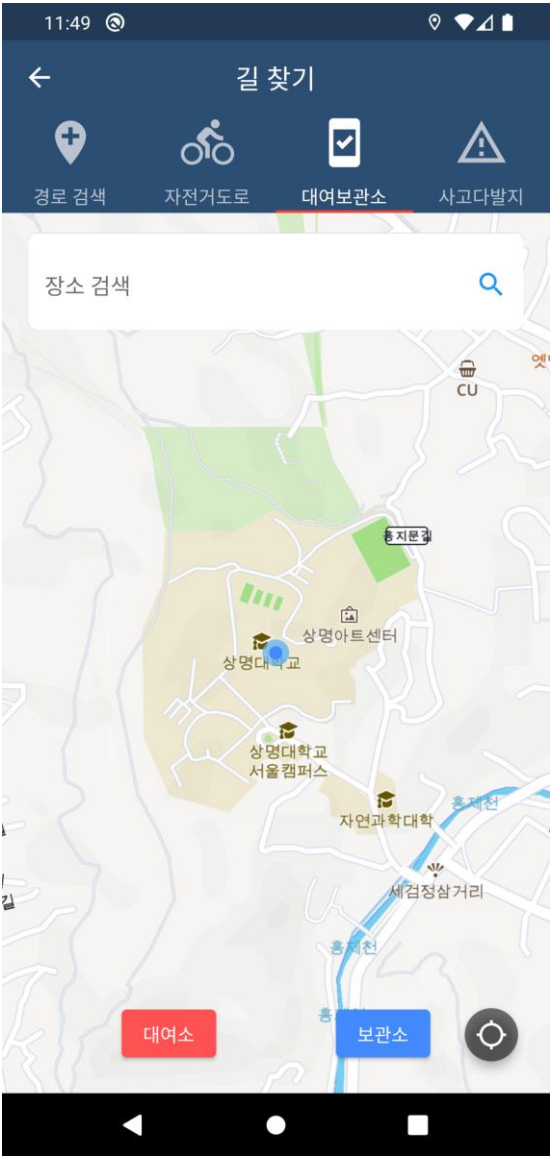
Plugins supporting *.csv files found.

Install plugins

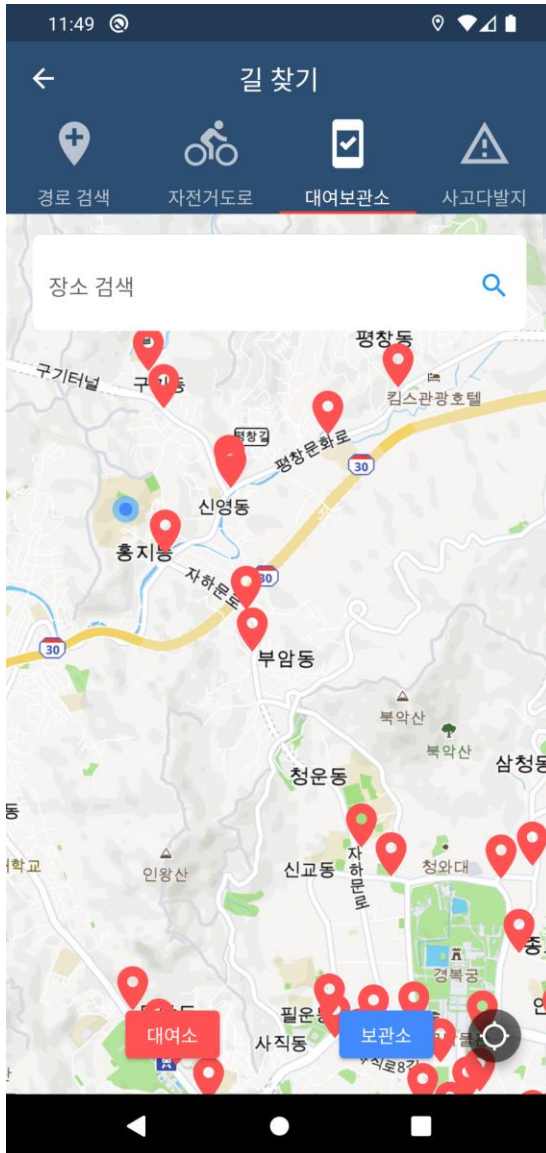
Ignore extension

1	"자전거보관소명", 소재지도로명주소, 소재지지번주소, 위도, 경도, 보관대수, 설치연도, 설치형태, 차양막설치여부, 공기주입기비치여부, 1
2	"홍대문역2번출구", , 서울특별시 서대문구 충정로2가 8-2, 37.591049, 126.941365, 20, 2013, 단독형, N, Y, 수동식, N, 02-330-1906, 서대문구청 교통행정과, 2020-12-1
3	"서대문역8번출구", , 서울특별시 서대문구 미군동102-1, 37.5626363387, 126.9668051023, 15, 2016, 단독형, N, Y, 기계식+태양광식, N, 02-330-1906, 서대문구청 교통행정과, 2020-12-1
4	"충정로역9번출구", , 서울특별시 서대문구 충정로3가 360-25, 37.5604990813, 126.9634393687, 8, 2003, 연결형, N, N, , N, 02-330-1906, 서대문구청 교통행정과, 2020-12-1
5	"독립문역4번출구", , 서울특별시 서대문구 현저동 101, 37.5761055433, 126.9554445886, 30, 2002, 연결형, Y, Y, 수동식, N, 02-330-1906, 서대문구청 교통행정과, 2020-12-1
6	"아현역1번출구", , 서울특별시 서대문구 북아현동136-13, , 11, 2016, 단독형, N, Y, 기계식+태양광식, N, 02-330-1906, 서대문구청 교통행정과, 2020-12-1
7	"가좌역 4번출구", , 서울특별시 서대문구 남가좌동 293-64, 37.5688870324, 126.9151497912, 15, 2014, 연결형, Y, N, , N, 02-330-1906, 서대문구청 교통행정과, 2020-12-1
8	"가좌역 4번출구", , 서울특별시 서대문구 남가좌동 293-64, 37.5688870324, 126.9151497912, 15, 2016, 단독형, N, Y, 기계식+태양광식, N, 02-330-1906, 서대문구청 교통행정과, 2020-12-1
9	"가좌역 3번출구", , 서울특별시 서대문구 남가좌동 293-64, 37.5688870324, 126.9151497912, 15, 2014, 연결형, Y, N, , N, 02-330-1906, 서대문구청 교통행정과, 2020-12-1
10	"이대역4번출구", , 서울특별시 서대문구 대현동 45-9, 37.5570713640, 126.9463217315, 5, 2013, 단독형, N, N, , N, 02-330-1906, 서대문구청 교통행정과, 2020-12-1
11	"홍제역4번출구", 서울특별시 서대문구 통일로457, , 37.5892089495, 126.9432416000, 19, 2016, 단독형, N, N, , N, 02-330-1906, 서대문구청 교통행정과, 2020-12-1
12	"홍제역3번출구", 서울특별시 서대문구 통일로437-1, , 37.5879510856, 126.9447860048, 11, 2016, 단독형, N, N, , N, 02-330-1906, 서대문구청 교통행정과, 2020-12-1
13	"무악재역3번출구", 서울특별시 서대문구 통일로357, , 37.5819884999, 126.9498460711, 7, 2013, 단독형, N, N, , N, 02-330-1906, 서대문구청 교통행정과, 2020-12-1
14	"홍제역3,4번출구 사이", 서울특별시 서대문구 통일로451, , 37.5886701766, 126.9438499068, 9, 2016, 단독형, N, N, , N, 02-330-1906, 서대문구청 교통행정과, 2020-12-1
15	"아현역1번출구", , 서울특별시 서대문구 북아현동 1015, 37.5586416507, 126.9559727797, 20, 2017, 단독형, N, N, , N, 02-330-1906, 서대문구청 교통행정과, 2020-12-1
16	"신촌역 2번출구", 서울특별시 서대문구 신촌로 93, , 37.5556893522, 126.9365678281, 11, 2017, 단독형, N, N, , N, 02-330-1906, 서대문구청 교통행정과, 2020-12-1
17	"신촌역 3번출구", 서울특별시 서대문구 신촌로 99, , 37.5556905692, 126.9371588202, 16, 2017, 단독형, N, Y, 기계식+태양광식, N, 02-330-1906, 서대문구청 교통행정과, 2020-12-1
18	"신촌역 1번출구", 서울특별시 서대문구 신촌로 91, , 37.5555881567, 126.9363751793, 5, 2017, 단독형, N, N, , N, 02-330-1906, 서대문구청 교통행정과, 2020-12-1
19	"서대문구청 버스정류장(13208)", , 서울특별시 서대문구 연희동 743, 37.5788349700, 126.9344773839, 34, 2007, 연결형, Y, N, , N, 02-330-1906, 서대문구청 교통행정과, 2020-12-1
20	"사천교 버스정류장", , 서울특별시 서대문구 연희동 375-4, , 14, 2007, 연결형, N, N, , N, 02-330-1906, 서대문구청 교통행정과, 2020-12-1
21	"정원여중 정류장 지난 건널목", , 서울특별시 서대문구 홍은동 산 11-51, , 34, 2005, 연결형, Y, N, , N, 02-330-1906, 서대문구청 교통행정과, 2020-12-1
22	"정원여중 정류장", , 서울특별시 서대문구 홍은동 산 11-75, , 28, 2012, 연결형, Y, N, , N, 02-330-1906, 서대문구청 교통행정과, 2020-12-1
23	"연희동자치회관", , 서울특별시 서대문구 연희동 122-16, 37.5693360135, 126.9307755346, 10, 2005, 연결형, N, N, , N, 02-330-1906, 서대문구청 교통행정과, 2020-12-1
24	"27구간 노외주차장", 서울특별시 서대문구 연희동 537-145, 37.5700097908, 126.9214394629, 14, 2007, 연결형, Y, N, , N, 02-330-1906, 서대문구청 교통행정과, 2020-12-1
25	"2구간 노외주차장", 서울특별시 서대문구 연희동 541-49, , 17, 2007, 연결형, Y, N, , N, 02-330-1906, 서대문구청 교통행정과, 2020-12-1
26	"홍은4공영주차장", , 서울특별시 서대문구 홍은동 9-539, , 10, 2009, 단독형, N, Y, 기계식, N, 02-330-1906, 서대문구청 교통행정과, 2020-12-1
27	"홍은1공영주차장", 서울특별시 서대문구 포방터2길10, , 37.5986792508, 126.9495412415, 7, 2016, 단독형, N, N, , N, 02-330-1906, 서대문구청 교통행정과, 2020-12-1
28	"충정로KT&G 앞 육교 밑", , 서울특별시 서대문구 충정로2가 157-2, , 10, 2007, 연결형, N, N, , N, 02-330-1906, 서대문구청 교통행정과, 2020-12-1
29	"삼창빌딩", , 서울특별시 서대문구 충정로3가 63-1, 37.5605682920, 126.9626211487, 18, 2003, 단독형, N, N, , N, 02-330-1906, 서대문구청 교통행정과, 2020-12-1
30	"농협빌딩(구 임광빌딩)", , 서울특별시 서대문구 미군동 267, 37.5623619358, 126.9686005457, 14, 2007, 연결형, Y, N, , N, 02-330-1906, 서대문구청 교통행정과, 2020-12-1
31	"봉원고가앞", , 서울특별시 서대문구 대신동 115, 37.5664180410, 126.9463410590, 10, 2013, 단독형, N, N, , N, 02-330-1906, 서대문구청 교통행정과, 2020-12-1
32	"서대문구청", , 서울특별시 서대문구 연희동 168-6, 37.5791546257, 126.9367591751, 50, 2007, 연결형, Y, Y, 기계식, N, 02-330-1906, 서대문구청 교통행정과, 2020-12-1
33	"서연중학교", 서울특별시 서대문구 연희동 267-1, 37.5682252703, 126.9273335812, 65, 2007, 연결형, Y, Y, 기계식, N, 02-330-1906, 서대문구청 교통행정과, 2020-12-1
34	"주택가", , 서울특별시 서대문구 연희동 179-123, 37.5759685704, 126.9315023937, 10, 2005, 연결형, N, N, , N, 02-330-1906, 서대문구청 교통행정과, 2020-12-1
35	"홍연연립 77동앞", , 서울특별시 서대문구 연희동 179-64, 37.5761274399, 126.9315440206, 14, 2004, 연결형, N, N, , N, 02-330-1906, 서대문구청 교통행정과, 2020-12-1

대여소, 보관소 표시



대여소, 보관소 표시



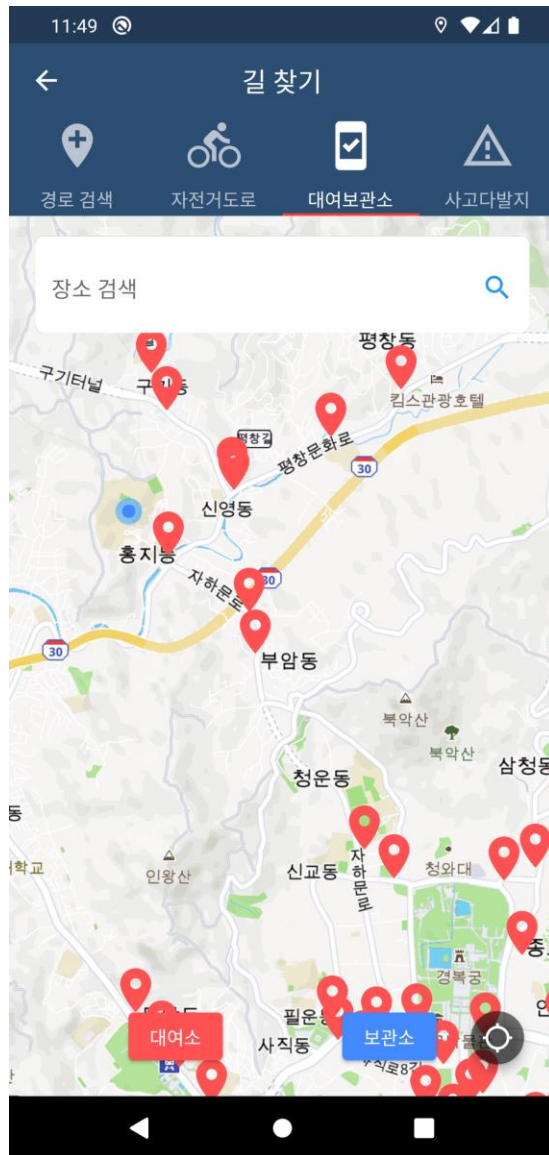
```
List<LatLng> _markerPositions = [];  
List<String> _markerAddress = [];  
List<List<dynamic>> csvdata = [];
```

- 위도 경도 값을 저장할 LatLng 리스트
- 해당 위치 값의 주소를 저장할 String 리스트
- 대여보관소 데이터를 받아올 이중 dynamic 타입 리스트

```
loadCSV_rental() async {  
  final Data = await rootBundle.loadString('Assets/data/rentalSeoul.csv');  
  List<List<dynamic>> csvData = CsvToListConverter().convert(Data);  
  setState(() {  
    csvdata = csvData;  
    for (int i=0; i<2170; i++){  
      if (csvdata[i][2] == '종로구'){  
        double la = csvdata[i][4];  
        double lo = csvdata[i][5];  
        _markerPositions.add(LatLng(la, lo));  
        _markerAddress.add(csvdata[i][1]);  
      }  
    }  
    rental = true;  
  });  
}
```

- CsvToListConverter 함수로 csv 데이터를 변환하여 이중 리스트에 저장
- 반복문으로 이중 리스트의 4, 5 index에 입력되어있는 double 타입의 위도, 경도 값을 LatLng 타입으로 변환 후 리스트에 저장
- 같은 방식으로 주소값 문자열 리스트 저장
- 대여소 버튼을 누를때마다 rental = true;로 설정하여 보관소와 구분

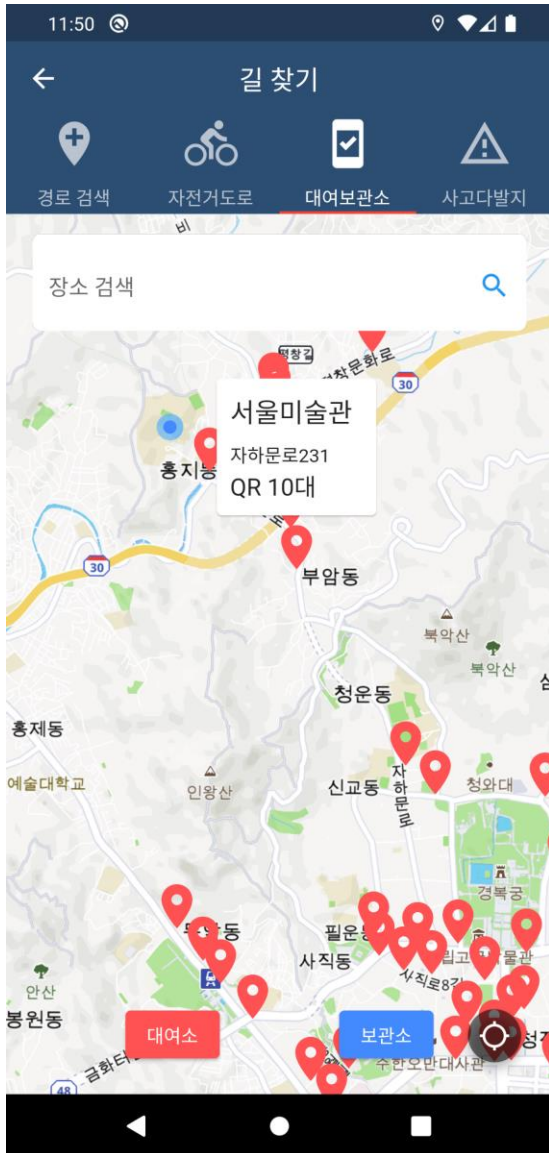
대여소, 보관소 표시



```
Container(  
  child: ElevatedButton(  
    onPressed: () async {  
      _popupLayerController.hideAllPopups();  
      setState(() {  
        _markerPositions.clear();  
      });  
      await loadCSV_rental();  
      print(rental);  
    },  
    style: ButtonStyle(  
      backgroundColor: MaterialStateProperty.all(Colors.redAccent),  
    ), // ButtonStyle  
    child: Text("대여소"),  
  ), // ElevatedButton  
, // Container  
  Spacer(),  
  Container(  
    child: ElevatedButton(  
      onPressed: () async {  
        _popupLayerController.hideAllPopups();  
        setState(() {  
          _markerPositions.clear();  
        });  
        await loadCSV_park();  
        print(rental);  
      },  
      style: ButtonStyle(  
        backgroundColor: MaterialStateProperty.all(Colors.blueAccent),  
      ), // ButtonStyle  
      child: Text("보관소"),  
    ), // ElevatedButton  
, // Container
```

버튼을 누를때마다
_markerPositions.clear();
이전에 표시됐던 marker
리스트를 모두 삭제한 후
대여소 또는 보관소 함수 호출

대여소, 보관소 정보 팝업



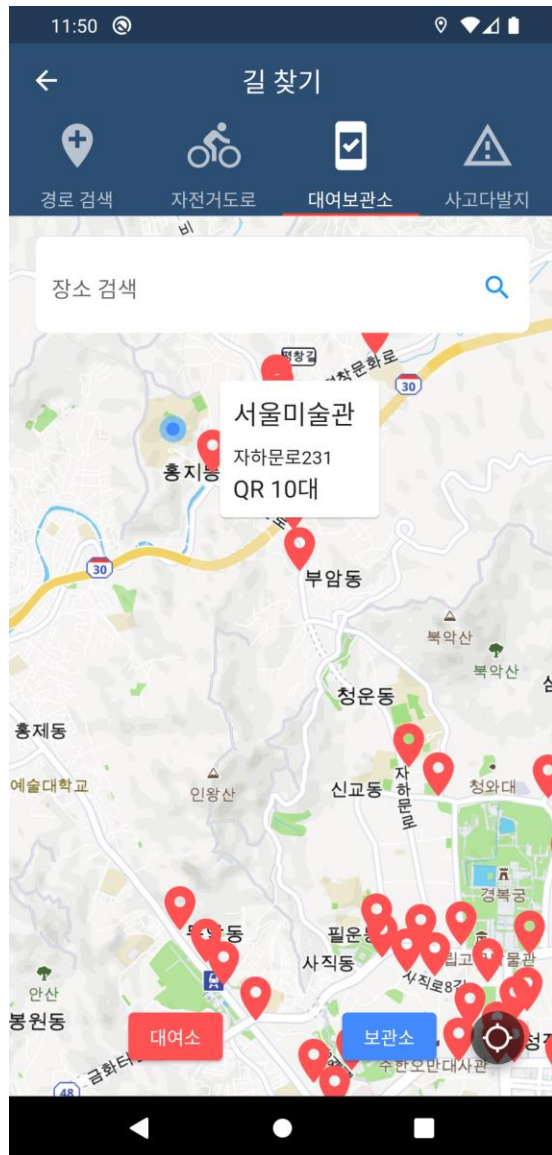
```
List<Marker> get _markers => _markerPositions
    .map(
      (markerPosition) => Marker(
        point: markerPosition,
        width: 40,
        height: 40,
        builder: (_) => Icon(Icons.location_on, size: 40, color:
          (rental == true ? Colors.redAccent : Colors.blueAccent),),
        anchorPos: AnchorPos.align(AnchorAlign.top),
      ), // Marker
    ).toList();
```

- 위치 값이 저장된 리스트에서 point값을 가져와 marker 생성
- rental = true 일 경우와 아닐 경우 아이콘의 색이 바뀜
- 생성한 marker를 _markers라는 리스트로 변환 => .toList();

```
PopupMarkerLayerWidget(
  options: PopupMarkerLayerOptions(
    popupController: _popupLayerController,
    markers: _markers,
    popupAnimation: PopupAnimation.fade(duration: Duration(milliseconds: 700)),
    markerRotateAlignment:
      PopupMarkerLayerOptions.rotationAlignmentFor(AnchorAlign.top),
    popupBuilder: (BuildContext context, Marker marker) =>
      Popup(marker),
  ), // PopupMarkerLayerOptions
), // PopupMarkerLayerWidget
```

- 팝업 마커 위젯 패키지 이용

대여소, 보관소 정보 팝업 위젯



```
String Addr = '';
String LCDQR = '';
String AddrF = '';
List<List<dynamic>> csvdata = [];

@override
void initState() {
  super.initState();
  _loadCSV();
}
```

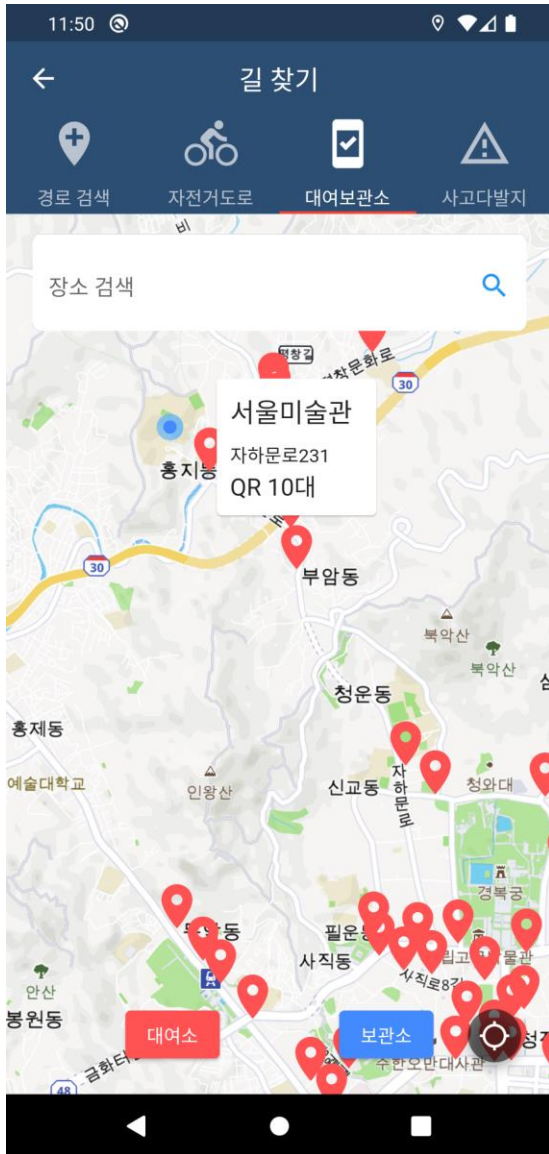
- 대여, 보관소 이름
- LCD / QR 여부
- 주소

- 자동으로 _loadCSV 함수 호출

```
late String path;
_loadCSV() async {
  if (rental == true){
    path = 'Assets/data/rentalSeoul.csv';
  }
  else if (rental == false){
    path = 'Assets/data/park_seodaemun.csv';
  }
  final Data = await rootBundle.loadString(path);
  List<List<dynamic>> csvData = CsvToListConverter().convert(Data);
}
```

- 대여소 버튼을 눌렀을 때 (rental == true) 대여소 데이터 링크 호출
- 보관소 버튼을 눌렀을 때 (rental == false) 보관소 데이터 링크 호출하여 convert

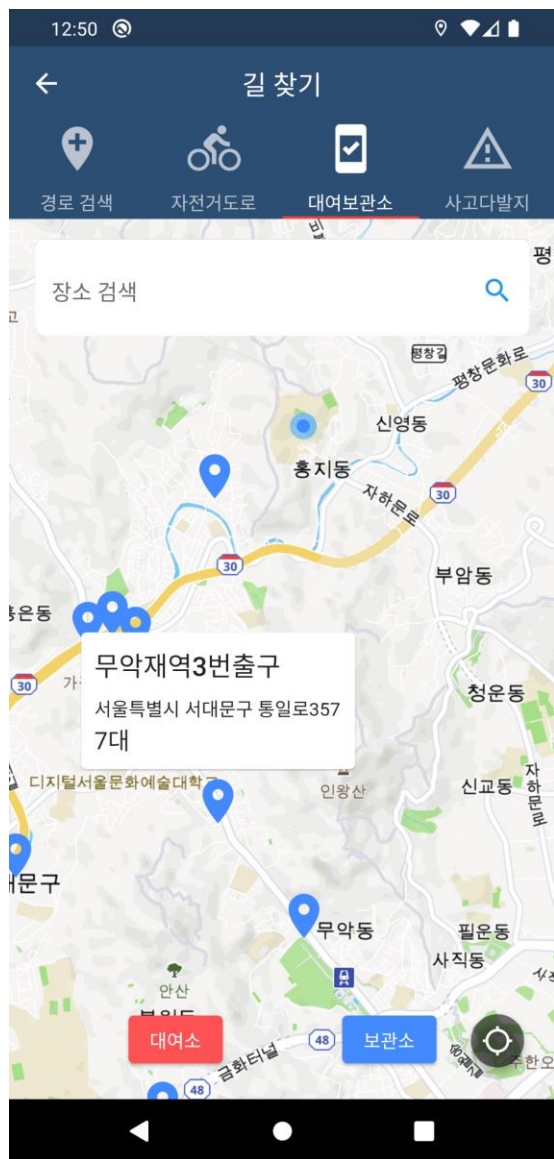
대여소, 보관소 정보 팝업 위젯



```
if (rental == true){
    for (var i=0; i<2170; i++){
        if (_marker.point.latitude == csvdata[i][4] && _marker.point.longitude == csvdata[i][5]){
            Addr = csvdata[i][1];
            if (csvdata[i][9] == 'LCD'){
                LCDQR = "LCD " + csvdata[i][7].toString() + "대";
            }
            else {
                LCDQR = "QR " + csvdata[i][8].toString() + "대";
            }
            AddrF = csvdata[i][3];
        }
    }
}
else if (rental == false){
    for(var i=1; i<51; i++){
        if (_marker.point.latitude == csvdata[i][3] && _marker.point.longitude == csvdata[i][4]){
            Addr = csvdata[i][0];
            LCDQR = csvdata[i][5].toString() + "대";
            AddrF = csvdata[i][1];
        }
    }
}
```

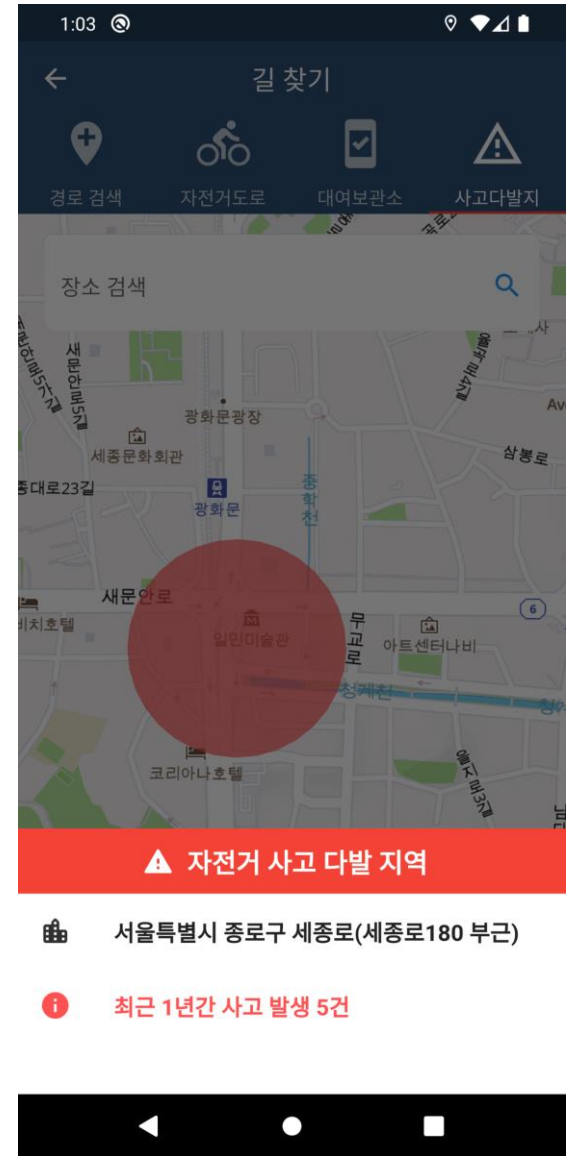
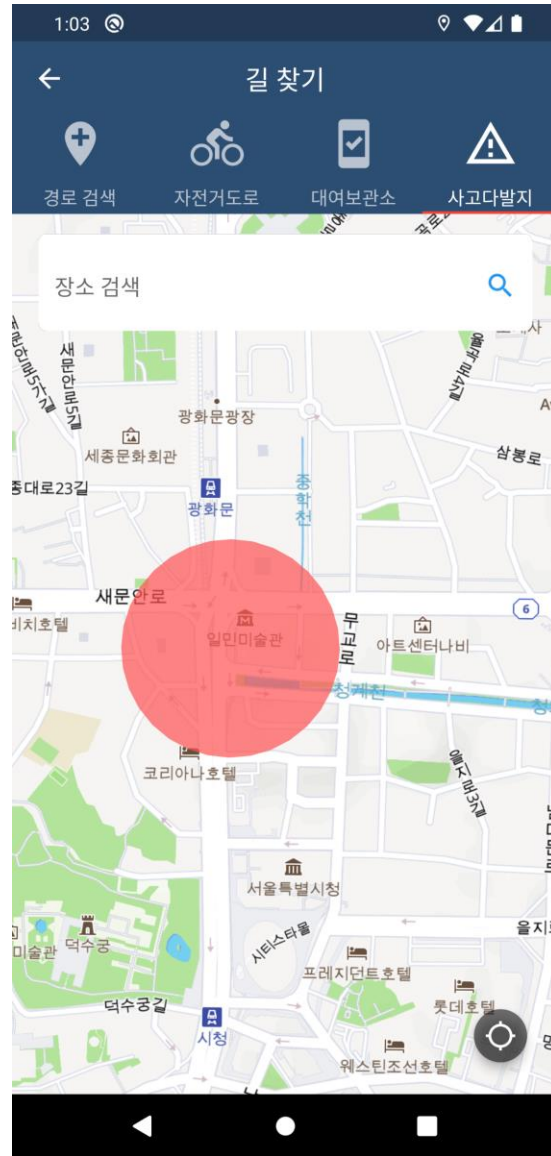
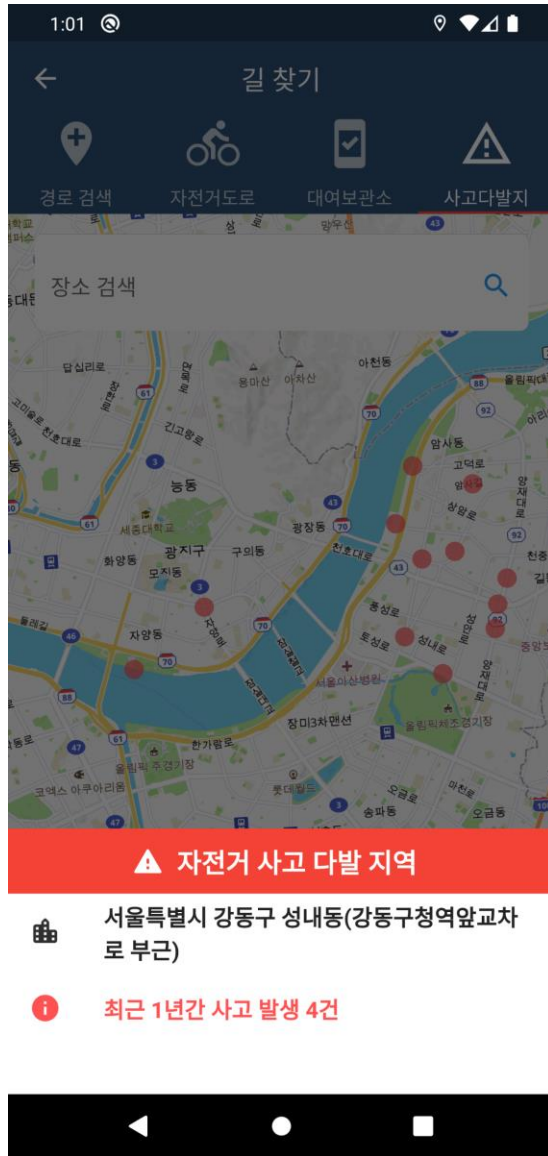
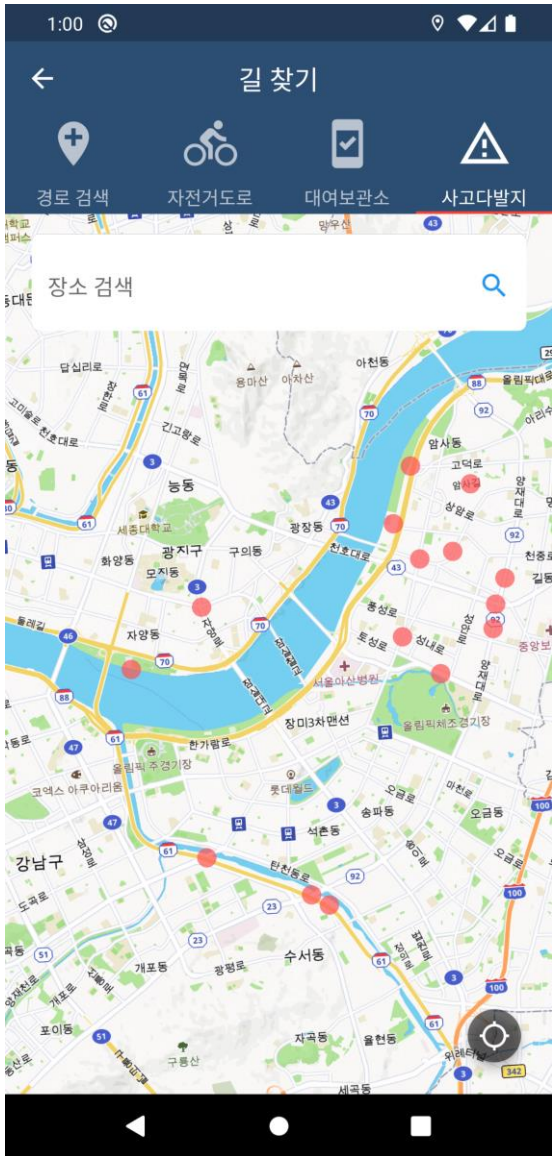
터치한 marker point의 위치값과 일치하는 대여, 보관소 데이터를 반복문을 이용하여 찾고 해당 주소와 정보들을 각각의 변수에 저장

대여소, 보관소 정보 팝업 위젯



```
Widget _cardDescription(BuildContext context) {  
  return Padding(  
    padding: const EdgeInsets.all(10),  
    child: Container(  
      constraints: BoxConstraints(minWidth: 100, maxWidth: 300),  
      child: Column(  
        crossAxisAlignment: CrossAxisAlignment.start,  
        mainAxisAlignment: MainAxisAlignment.start,  
        mainAxisAlignment: MainAxisAlignment.min,  
        children: <Widget>[  
          Text(  
            Addr,  
            overflow: TextOverflow.fade,  
            softWrap: false,  
            style: const TextStyle(  
              fontWeight: FontWeight.w500,  
              fontSize: 20.0,  
            ), // TextStyle  
          ), // Text  
          const Padding(padding: EdgeInsets.symmetric(vertical: 4.0)),  
          Text(  
            AddrF,  
            style: const TextStyle(fontSize: 14.0),  
          ), // Text  
          Text(  
            LCDQR,  
            style: const TextStyle(fontSize: 18.0),  
          ), // Text  
        ],  
      ),  
    ),  
  );  
}
```

사고 다발 지역 표시



map_210521 D:\map_210521

- > .dart_tool
- > .idea
- > android
- > Assets
 - data
 - AccidentData.json
 - park_seodaemun.csv
 - rentalSeoul.csv
- > Icon
- > Image
- > sprite_images
 - license.txt
 - pin.png
 - style.json
- > build
 - image
 - ios
 - lib
 - config
 - search
 - Road
 - SeoulRoad
 - GangNam.dart
 - JongRo.dart
 - Mapo.dart
 - Yeongdeungpo.dart
 - Road1.dart
 - Road2.dart
- Accident.dart
- bicycleRoad.dart
- MapStorage.dart
- place.dart
- popup.dart
- popup_ac.dart
- search.dart
- searchRoute.dart

```
{
  "resultCode": "00",
  "resultMsg": "NORMAL_CODE",
  "items": {
    "item": [
      {
        "afos_fid": 6687123,
        "afos_id": "2021028",
        "bjd_cd": "1168011500",
        "spot_cd": "11680001",
        "sido_sgg_nm": "서울특별시 강남구1",
        "spot_nm": "서울특별시 강남구 수서동(수서IC 부근)",
        "occurnc_cnt": 5,
        "caslt_cnt": 7,
        "dth_dnv_cnt": 0,
        "se_dnv_cnt": 1,
        "sl_dnv_cnt": 2,
        "wnd_dnv_cnt": 4,
        "geom_json": "{\"type\": \"Polygon\", \"coordinates\": [[[127.10543984, 37.4916075], [127.10540532, 37.4916075], [127.10540532, 37.4916075], [127.10543984, 37.4916075]]]}",
        "lo_crd": "127.103643208169",
        "la_crd": "37.491607501286"
      },
      {
        "afos_fid": 6686746,
        "afos_id": "2021028",
        "bjd_cd": "1168011400",
        "spot_cd": "11680002",
        "sido_sgg_nm": "서울특별시 강남구2",
        "spot_nm": "서울특별시 강남구 일원동(강남구송파구 부근)",
        "occurnc_cnt": 4,
        "caslt_cnt": 5,
        "dth_dnv_cnt": 0,
        "se_dnv_cnt": 3,
        "sl_dnv_cnt": 1,
        "wnd_dnv_cnt": 1,
        "geom_json": "{\"type\": \"Polygon\", \"coordinates\": [[[127.10543984, 37.4916075], [127.10540532, 37.4916075], [127.10540532, 37.4916075], [127.10543984, 37.4916075]]]}",
        "lo_crd": "127.103643208169",
        "la_crd": "37.491607501286"
      }
    ]
  }
}
```

원 형태의 polygon 위도, 경도 값 리스트

자전거 사고 발생 건수

중심 위치 값

자전거 사고 발생 건수

원 형태의 polygon 위도, 경도 값 리스트

사고 다발 지역 표시

```
String path = 'Assets/data/AccidentData.json';
loadJsonData() async {
  final String response = await rootBundle.loadString(path);
  Map<String, dynamic> acd = jsonDecode(response);
  String jsonData;
  String jsonData_la, jsonData_lo;
  String jsonData_add;
  int jsonData_cnt;
  int num = acd['totalCount'];
```

- ① JSON 데이터를 String 형태 response로 불러온 뒤 Map<String, dynamic> 이라는 타입의 변수에 jsonDecode 함수를 이용해 decode 하여 저장
- ② 반복문을 이용해서 json 데이터의 key 값에 맞는 value 값들을 각각 타입에 맞는 리스트들에 저장
- ③ 하나의 지역이 원 모양으로 표시되고 지도에 여러 개의 지역을 표시해야 하기 때문에 하나의 원을 만드는 25개 위도 경도값들의 리스트를 이중 리스트에 저장
- ④ 이중 리스트에 저장된 point로 30개 polygon 생성

```
for (i = 0; i < 30; i++) {
  jsonData_la = acd['items']['item'][i]['la_crd'];
  jsonData_lo = acd['items']['item'][i]['lo_crd'];
  _la = double.parse(jsonData_la);
  _lo = double.parse(jsonData_lo);
  centerPoints.add(LatLng(_la, _lo)); // 중심값

  jsonData_add = acd['items']['item'][i]['spot_nm'];
  _Address.add(jsonData_add); // 주소값

  jsonData_cnt = acd['items']['item'][i]['occrrnc_cnt'];
  _Occr.add(jsonData_cnt); // 발생건수

  jsonData = acd['items']['item'][i]['geom_json'];
  temp = jsonDecode(jsonData)['coordinates'][0];
  areaPoints.add(temp);
  for (j = 0; j < temp.length; j++) {
    latlng = LatLng(areaPoints[i][j][1], areaPoints[i][j][0]);
    _areaPoints.add(latlng);
  }
  areaList.add(_areaPoints.toList());
  _areaPoints.clear();
}
polygon = [Polygon(points: areaList[0], color: Colors.redAccent.withOpacity(0.7))];
for(k=1; k<30; k++){
  polygon.add(
    Polygon(points: areaList[k],
      color: Colors.redAccent.withOpacity(0.7)),
  );
}
```

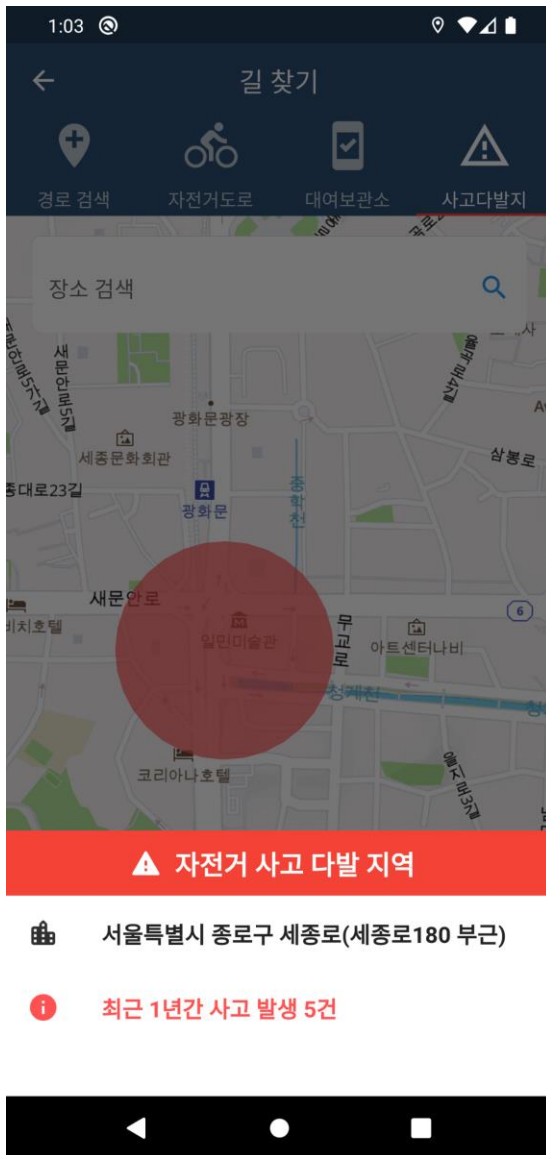
사고 다발 지역 정보 팝업



```
List<Marker> get _markers => centerPoints
    .map(
        (markerPosition) => Marker(
            width: 40.0,
            height: 40.0,
            point: markerPosition,
            builder: (ctx) =>
                new Container(
                    child: IconButton(
                        icon: Icon(Icons.brightness_1_outlined),
                        color: Colors.transparent,
                        iconSize: 0.0,
                        onPressed: () {
                            _showModalBottomSheet(markerPosition);
                        },
                    ) // IconButton
                ) // Container
            ) // Marker
    ).toList();
```

기본적인 polygon에는 onPressed 기능 X
mapbox 지도에 사용 가능한 관련 패키지도 존재하지 않음
=> polygon의 중심 위치값에 투명 marker 추가하여
사용자가 해당 부분 터치 시 위치 정보 제공

사고 다발 지역 정보 팝업

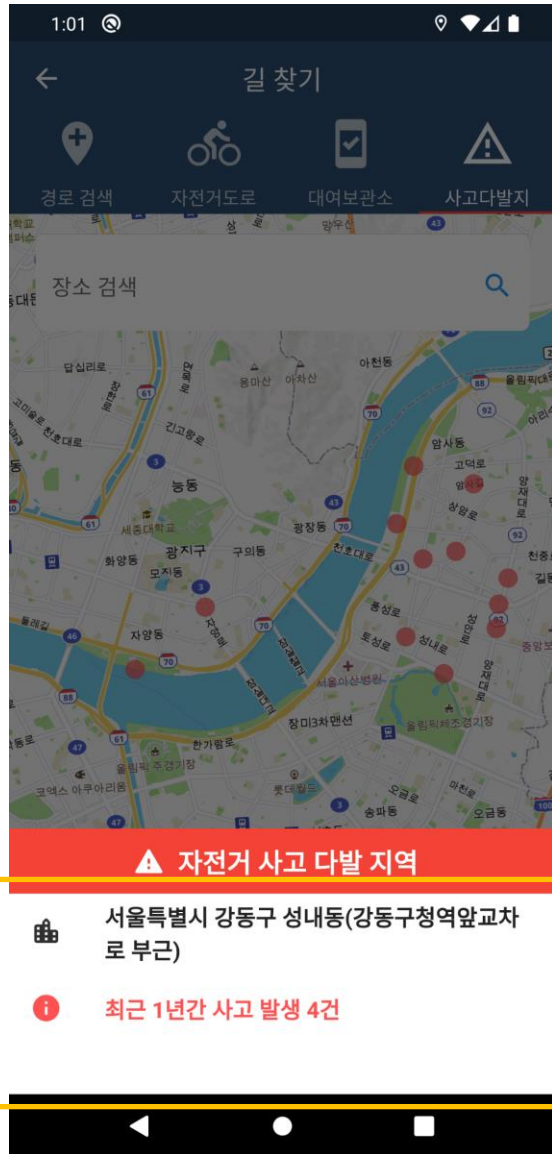


```
void _showModalBottomSheet(LatLng latLng){  
  for (var p=0; p<30; p++){  
    if(latLng == centerPoints[p]){  
      Addr = _Address[p];  
      Occr = _Occr[p];  
    }  
  }  
  showModalBottomSheet(  
    context: context,  
    builder: (builder) {  
      return Container(  
        width: double.infinity,  
        color: Colors.white,  
        height: 200.0,  
        child: Column(  
          children: <Widget>[  
            Container(  
              width: double.infinity,  
              color: Colors.red,  
              child: Row(  

```

터치한 marker가 사고 다발 지역 위치와 일치할 경우
해당 위치의 주소값과 사고 발생 건수를
showModalBottomSheet 위젯을 이용하여 출력

사고 다발 지역 정보 팝업



```
Align(
  alignment: Alignment.center,
  child: Column(
    children: <Widget>[
      Container(
        child: ListTile(
          leading: Icon(Icons.location_city_rounded,
            color: Colors.grey[850],
          ), // Icon
          title: Text(Addr,
            style: TextStyle(fontSize: 17.0,
              fontWeight: FontWeight.w700)), // TextStyle, Text
          onTap: (){
            Navigator.pop(context);
          },
        ), // ListTile
      ), // Container
      Container(
        child: ListTile(
          leading: Icon(Icons.info,
            color: Colors.redAccent,
          ), // Icon
          title:
            Text('최근 1년간 사고 발생 ' + Occr.toString() + '건',
              style: TextStyle(fontSize: 17.0, color: Colors.redAccent,
                fontWeight: FontWeight.w700)), // TextStyle, Text
          onTap: (){
            Navigator.pop(context);
          },
        ),
      ),
    ],
  ),
```


THANK YOU