

Capstone Design I

Final Project Report

Date.	2023.06.14	
Team2	2019111635	김서영
	2019112101	추예진
	2019112101	김동재
	2019112118	이효빈

Table of Contents

1. SYSTEM OVERVIEW	3
1.1. SUMMERY	3
1.2. PURPOSE	3
1.3. PRIOR RESEARCH	오류! 책갈피가 정의되어 있지 않습니다.4
2. PROJECT PROGRESS	4
3. PROJECT DESIGN AND IMPLEMENTATION APPROACH	5
3.1. MODULES	5
3.2. SERVER	5
3.3. FRONT-END	32
3.4. DATA	39
4. RESULTS	52
5.....	

1. System OverView

1.1 Summary

상기 문서는 팀 forKids가 개발한 앱 forKids의 목표와 설계, 구현 과정과 결과물에 대해 내용을 포함한다. forKids는 현대사회의 양육자들이 양육에 있어서 장소 선정에 어려움을 겪고 있거나, 사전 조사를 위한 충분한 시간이 없어 체험 프로그램 등을 놓치는 문제 상황을 해결하기 위해 팀이 개발한 앱 기능으로 사용자가 원하는 장소 정보를 손쉽게 얻을 수 있도록 앱을 설계하였다. 첫째로, 사용자는 챗봇과 대화 형식으로 아이의 특성, 연령대에 맞는 장소를 추천 받을 수 있으며 근처의 관련된 편의 정보 또한 얻을 수 있다. 다른 뷰에서는 사용자들의 선호도를 바탕으로 한 장소 리스트를 보여주고, 사용자의 위치를 기반으로 가까운 장소들을 보여주어 상황에 맞는 시설 정보를 얻을 수 있다. 마지막으로, 기간이 있으며 일회성으로 제공되는 체험 프로그램은 따로 큐레이션 캘린더를 통해 원하는 날짜에 운영하는 행사 정보를 얻을 수 있다.

본 팀은 Google에서 제공하는 Dialogflow를 사용하여 챗봇의 기본 대화를 구성하였으며, 추천, 정보 제공 알고리즘은 Python 코드를 사용하여 개발하였다. Expo를 사용한 React-native 프레임워크를 통해 Javascript 언어를 사용하여 클라이언트의 앱 디자인과 컴포넌트, 스크린 등을 구현하였고 서버는 Typescript, Node.js를 기반으로 챗봇 API를 호출하는 webhook서버, 클라이언트와 통신하는 웹 서버를 각각 모듈화 하여 분리 하였으며, 전체 시설 정보를 조회하기 위한 비정형 데이터베이스인 MongoDB를 사용하였다.

1.2 Purpose

위 프로젝트는 아이들에게 다양한 놀이 시설을 제공함으로써 아이들에게 다채롭고 특성에 맞는 놀이 시설을 제공하고자 하는 부모들의 니즈를 만족시킨다. 기존의 시, 구, 정부에서 제공하는 문화 시설 공간뿐만 아니라 실내 놀이 시설, 체험 클래스 등 아이들을 위한 경험의 질과 양이 과거에 비해 다채로워진 현대사회에서 양육자의 시간적, 물질적 측면에서의 부담을 덜어주는 것을 목표로 한다. 나아가 원활한 정보 제공 및 공유를 통해 사용자와 시설 관리자 간의 피드백이 즉각적으로 이뤄지면서 전반적인 양육 형태가 개선되고, 궁극적으로 관련 분야의 질이 올라가는 효과를 기대할 수 있다.

1.3 Prior Research

유사 서비스명				
	아이야 가자	맘맘	놀이의 발견	키드파인드
유사 기능	<ul style="list-style-type: none"> 아이와 함께 많이 찾는 장소 랭킹 조회 후기 및 꿀팁 공유 	<ul style="list-style-type: none"> 아이와 함께 가 볼만한 곳 추천 	<ul style="list-style-type: none"> 카테고리 별 장소 추천 테마 별 인기 있는 놀이 추천 	<ul style="list-style-type: none"> 주변 나들이 장소 확인(거리 순, 체크인순, 등록 순) 테마 별 나들이 장소 추천 장소 검색 (카테고리, 시설, 이벤트 별)

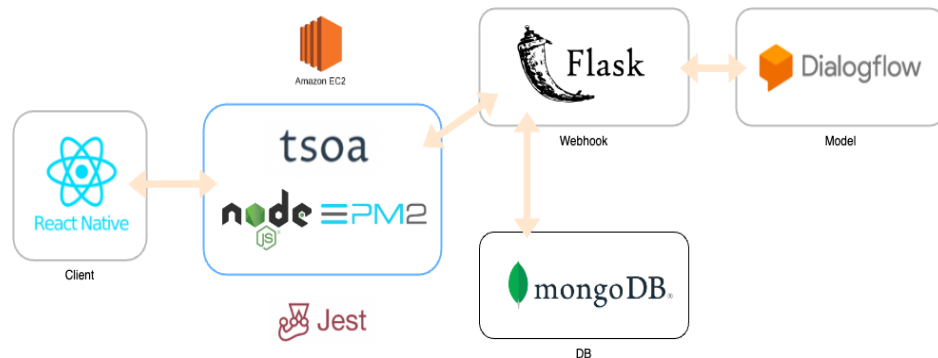
본 프로젝트에서 구현하려고 하는 기능과 유사한 기능을 제공하는 기존 어플리케이션이다. 구현하고자 하는 서비스와 유사한 서비스인 아동 놀이 장소 및 프로그램 추천 서비스는 '아이야 가자', '맘맘', '놀이의 발견', '키드파인드' 등이 있으며 기존 서비스에서는 추천 장소 정보들을 카테고리화 하여 해당 카테고리별로 인기 순으로 리스트화 하는 것에 초점이 맞춰지며, 해당 장소들을 조회하는 기능에 초점이 맞춰져 있다. ForKids 서비스는 사용자 자녀의 연령대와 취향을 고려한 개인 맞춤형 장소 추천을 제공하고 이 과정을 챗봇을 통해 제공한다. 해당 지역의 날씨, 챙겨야 할 준비물은 있는지, 장소 근처에 주차 공간은 있는지, 영업시간 및 영업일에 대한 정보 등 장소에 대한 다양한 부가 정보를 파악할 수 있는 정보의 접근 방향이 다양하며 그 과정이 챗봇이어서 편의를 갖춘 점에서 차별점을 가지고 있다. 아이의 성향에 맞춘 장소 추천을 통해 아이들의 놀이 만족도 향상에 기여하고, 이에 함께 참여하는 부모님이 알고 싶은 정보를 챗봇으로 편리하게 제공하여 아이와 놀러 갈 계획을 세우는 과정에 있어서 시간적 · 비용적 부담을 덜어줄 수 있다.

2. Project Progress

작업	주차					
	10주차	11주차	12주차	13주차	14주차	15주차
프로젝트 구현						
back-end						
front-end						
data preprocessing						
data modeling						
develop prototype						
test						
최종 보고서 및 발표						
final demonstration video						
final report						
team presentation						

3. Project Design and Implementation Approach

3.1 Modules



- Server : node.js와 Express를 사용한 웹 서버, 챗봇의 응답을 호출하는 webhook 서버를 통해 Client와 Data간의 원활한 정보 교류가 가능하도록 구현하였다.
- Front-end : 챗봇 화면과 캘린더 화면, 트렌드와 플레이스 화면을 통해 사용자가 원하는 정보를 다양한 방법으로 얻을 수 있도록 UI를 구성하였으며 컴포넌트 재사용성을 위해 시설 정보를 보여주는 Block을 모듈화 하였다.
- Data : 웹 크롤링을 통해 시설 정보 데이터 셋을 구축하고, 키워드를 붙여 추천 알고리즘 구현에 사용하였으며 라벨링한 데이터를 mongo DB에 추가하여 Client 측에서 데이터를 얻을 수 있도록 설계하였다.

3.2 Server

3.2.1 Architecture

포키즈 서비스는 Dialogflow 챗봇 모델을 사용하기 때문에 챗봇 API 를 호출하는 webhook 서버, 클라이언트와 통신하는 웹 서버를 모듈 두개로 분리하여 구현하기로 결정했다.

개발 초기에는 Node Express 웹서버 하나로 모듈만 분리하기로 설계 계획을 세웠으나 모델 장소 추천 알고리즘을 구현 시 python 라이브러리를 사용한 코드의 효율성이 더 좋다고 느껴 Flask 서버에서 해당 API 를 호출하는 방식으로 현재와 같이 서버 두개를 구축했다.

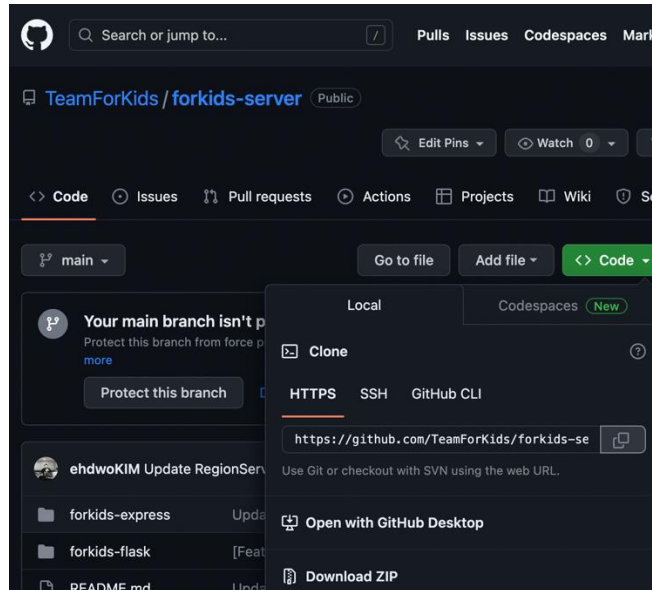
Dialogflow 모델에서 정해진 답변 이외에 사용자가 원하는 답변으로 동적으로 대화를 할 수 있게 하기 위해 Dialogflow Webhook 을 구현했는데, 이를 통해 클라이언트로부터 받은 질문에 대한 답을 동적으로 출력할 수 있다. 해당 webhook 서버를 Flask 를 통해 만들었다.

챗봇 API 와 장소추천, 거리계산 API 를 Python 기반의 Flask Webhook 서버에서 호출하기로 했고, 챗봇 이외의 기능은 Typescript 기반의 Express 서버에서 구현했다.

장소 정보 조회, 문화행사 정보 조회, 주차공간 정보 조회 등 데이터 조회가 많은 서비스 특성상 NoSQL 기반인 mongoDB 를 사용했다.

3.2.2 Development Environment Setting

구현 시작 전에 개발환경 초기 세팅을 했는데, Git 협업을 위해 만든 Repository 를 vscode 로컬환경에 클론하고, node 버전 확인 후 필요한 패키지를 설치해 주었다.



-typescript 로컬 전역 설치

```
npm install -g typescript
```

-ts 파일을 js 파일로 컴파일하기 위해 ts-node 설치

(tsc 는 대부분 Production 의 컴파일 단계에서 사용하며, 간단한 개발 단계에서의 컴파일은 build 파일이 필요없기 때문에 ts-node 를 사용)

```
> npm i -g ts-node
```

-npm 을 보완하기 위해 yarn 패키지 설치

```
> npm i -g yarn
```

-작업 경로에서 서버 사전 설정, package.json 파일 생성

```
> yarn init
```

```

package.json U
1  {
2    "name": "express-example",
3    "version": "1.0.0",
4    "main": "index.js",
5    "license": "MIT"
6  }

```

-express, nodemon(코드 변경 시, 서버 자동 재시작) 설치

```
yarn add express && yarn add -D @types/node @types/express nodemon
```

-tsconfig.json (TS->JS 로 컴파일하는 옵션 설정 가능) 생성

```
> tsc --init
```

-이후 package.json 에 실행 명령 script 추가

```

package.json U x tsconfig.json U
1  {
2    "name": "express-example",
3    "version": "1.0.0",
4    "main": "index.js",
5    "license": "MIT",
6    "scripts": {
7      "dev": "nodemon",
8      "build": "tsc && node dist"
9    },
10   "dependencies": {
11     "express": "^4.18.1"
12   },
13   "devDependencies": {
14     "@types/express": "^4.17.1"
15   }
16 }

```

[명령어 설명]
 yarn run dev → nodemon을 기반으로 서버를 실행
 yarn run build → TS 프로젝트를 JS로 빌드

-로컬 서버 구동 테스트 & 빌드 (이후 배포를 위해 필요)

```

yarn run dev
yarn run v1.22.18
$ nodemon
[nodemon] 2.0.20
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): src/**/*.env
[nodemon] watching extensions: js,ts,json
[nodemon] starting `ts-node --transpile-only ./src/index.ts`

#####
  Server listening on port: 3000
#####

```

```

> yarn run build
yarn run v1.22.18
$ tsc && node dist

#####
  Server listening on port: 3000
#####

```

초기 개발환경을 세팅 시 효율적인 팀 협업을 위한 Github 사용법과 전략을 설정했다.

🐙 git branch 전략

- main branch** : 배포 단위 branch
- develop branch** : 주요 개발 branch, main merge 전 거치는 branch
- feature branch** : 각자 이름 + 개발 branch

- 할 일 issue 등록 후 issue 번호로 branch 생성 후 작업
 - ex) feature/# issue num
- 해당 branch 작업 완료 후 PR 보내기
 - reviewer에 서로 tag후 code-review
 - comment 전 merge 불가!

branch 구조

```

- main
- develop
- ssong
  ├── #1
  └── #2

```

각자 맡은 파트 이외의 모듈 repository 에 커밋 또는 푸시를 하게 될 경우 위와 같은 공통적인 git branch 사용 전략을 사용했고, 코드 리뷰 시에 커밋을 구분하기 위해 다음과 같이 git message convention 을 설정했다.



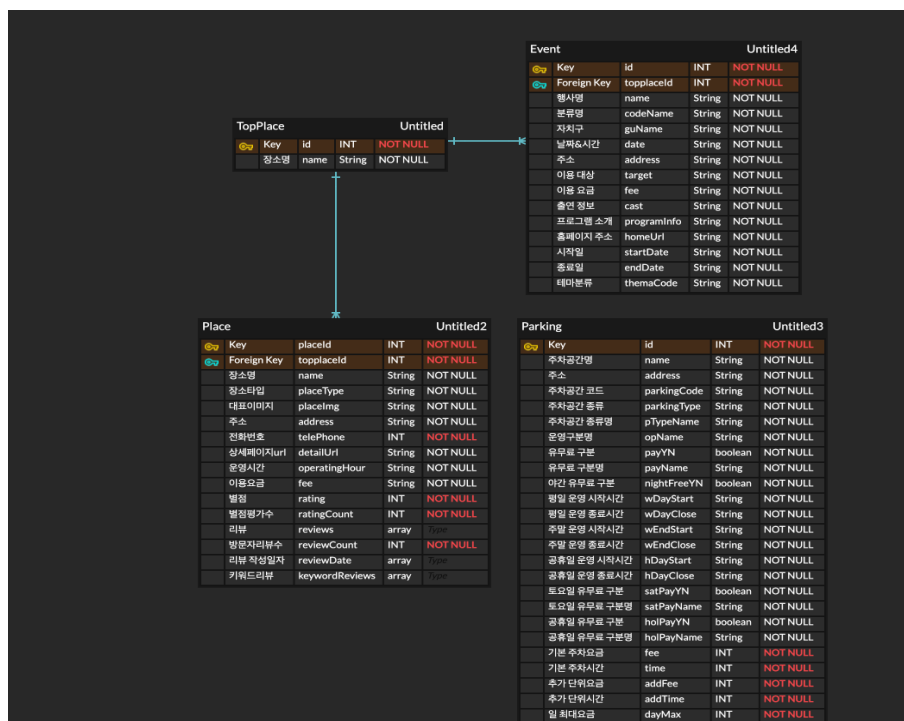
git commit message convention

ex) [FEAT] Add user api files

- [CHORE]: 코드 수정, 내부 파일 수정
- [FEAT]: ADD 이외의 부수적인 코드 추가, 라이브러리 추가, 새로운 파일 생성 시
- [ADD]: 새로운 기능 구현
- [FIX]: 버그, 오류 해결
- [DEL]: 쓸모없는 코드 삭제
- [DOCS]: README나 WIKI 등의 문서 개정
- [MOVE]: 프로젝트 내 파일이나 코드의 이동
- [RENAME]: 파일 이름의 변경
- [MERGE]: 다른브랜치를 merge하는 경우
- [STYLE]: 코드가 아닌 스타일 변경을 하는 경우
- [INIT]: Initial commit을 하는 경우

3.3.3 API Docs & DB Design

서버 개발환경을 설정하고, 크롤링 & 전처리한 데이터들 기반으로 데이터베이스 설계를 진행했다.



mongoDB 공식문서의 데이터 모델링 방식에 따라 mongoDB 자체는 schemaless지만 document 데이터 구조를 table과 schema를 활용한 ERD로 레퍼런스 모델링하였다.



각 table의 데이터 종류와 column은 다음과 같다.

- TopPlace: Place 테이블의 리뷰, 별점, 별점랭킹을 반영한 상위 100개 장소 저장.



Place 테이블에서 Flask Webhook 서버의 스코어 정렬 API 를 Express 웹 서버에서 호출하여 상위 100 개 장소를 추출하여 TopPlace 테이블에 저장한다.

TopPlace		Untitled		
	Key	id	INT	NOT NULL
	장소명	name	String	NOT NULL

- Place : 서울시 야영장, 유원지, 극장, 박물관, 미술관, 한옥체험시설, 휴게음식점, 작은도서관, 공공도서관 장소 정보를 저장

Place		Untitled2		
	Key	placeId	INT	NOT NULL
	Foreign Key	topplaceId	INT	NOT NULL
	장소명	name	String	NOT NULL
	장소타입	placeType	String	NOT NULL
	대표이미지	placeImg	String	NOT NULL
	주소	address	String	NOT NULL
	전화번호	telePhone	INT	NOT NULL
	상세페이지url	detailUrl	String	NOT NULL
	운영시간	operatingHour	String	NOT NULL
	이용요금	fee	String	NOT NULL
	별점	rating	INT	NOT NULL
	별점평가수	ratingCount	INT	NOT NULL
	리뷰	reviews	array	Type
	방문자리뷰수	reviewCount	INT	NOT NULL
	리뷰 작성일자	reviewDate	array	Type
	키워드리뷰	keywordReviews	array	Type

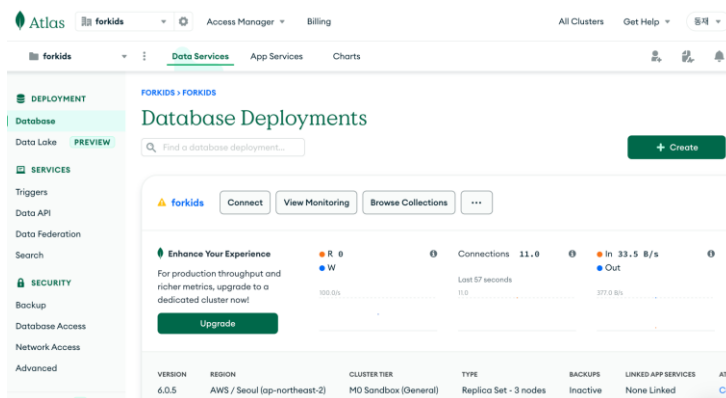
- Event : 큐레이션 캘린더 뷰에서 조회할 서울시 문화행사 정보를 저장.

Event		Untitled4		
	Key	id	INT	NOT NULL
	Foreign Key	topplaceId	INT	NOT NULL
	행사명	name	String	NOT NULL
	분류명	codeName	String	NOT NULL
	자치구	guName	String	NOT NULL
	날짜&시간	date	String	NOT NULL
	주소	address	String	NOT NULL
	이용 대상	target	String	NOT NULL
	이용 요금	fee	String	NOT NULL
	출연 정보	cast	String	NOT NULL
	프로그램 소개	programInfo	String	NOT NULL
	홈페이지 주소	homeUrl	String	NOT NULL
	시작일	startDate	String	NOT NULL
	종료일	endDate	String	NOT NULL
	테마분류	themaCode	String	NOT NULL

Parking : 서울시 주차공간 정보를 저장. Flask Webhook 서버의 거리계산 API로 사용자가 질문한 장소와 가장 가까운 주차공간을 조회할 수 있다.

Parking			Untitled3	
Key	id	INT	NOT NULL	
주차공간명	name	String	NOT NULL	
주소	address	String	NOT NULL	
주차공간 코드	parkingCode	String	NOT NULL	
주차공간 종류	parkingType	String	NOT NULL	
주차공간 종류명	pTypeName	String	NOT NULL	
운영구분명	opName	String	NOT NULL	
유무료 구분	payYN	boolean	NOT NULL	
유무료 구분명	payName	String	NOT NULL	
야간 유무료 구분	nightFreeYN	boolean	NOT NULL	
평일 운영 시작시간	wDayStart	String	NOT NULL	
평일 운영 종료시간	wDayClose	String	NOT NULL	
주말 운영 시작시간	wEndStart	String	NOT NULL	
주말 운영 종료시간	wEndClose	String	NOT NULL	
공휴일 운영 시작시간	hDayStart	String	NOT NULL	
공휴일 운영 종료시간	hDayClose	String	NOT NULL	
토요일 유무료 구분	satPayYN	boolean	NOT NULL	
토요일 유무료 구분명	satPayName	String	NOT NULL	
공휴일 유무료 구분	holPayYN	boolean	NOT NULL	
공휴일 유무료 구분명	holPayName	String	NOT NULL	
기본 주차요금	fee	INT	NOT NULL	
기본 주차시간	time	INT	NOT NULL	
추가 단위요금	addFee	INT	NOT NULL	
추가 단위시간	addTime	INT	NOT NULL	
일 최대요금	dayMax	INT	NOT NULL	

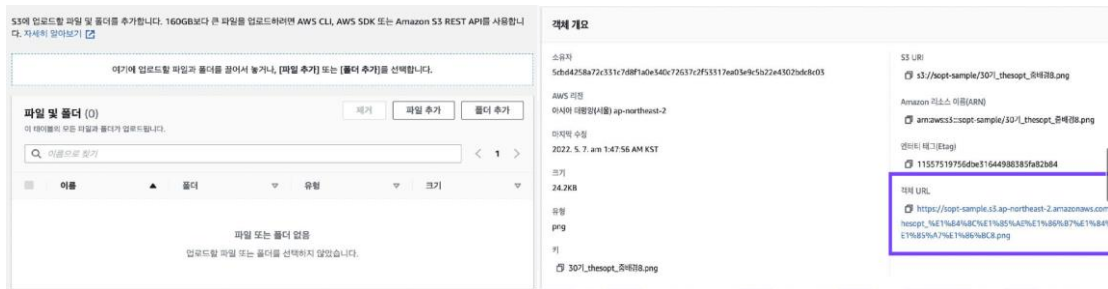
작성한 ERD 를 바탕으로 mongoDB 프로젝트를 생성하고 DB 를 연결해주었다.



mongoDB 내부의 데이터를 쉽게 볼수 있는 GUI인 MongoDB compass를 사용하여 Cluster와 connection을 완료했다. 이후 정제된 csv 데이터를 DB에 import했다.

장소와 문화행사 등의 이미지파일 관리를 위해 AWS S3 버킷도 생성해주었다.

S3 는 데이터를 버킷 내에 객체로 저장 가능하기 때문에 해당 클라우드 스토리지에 이미지 등의 데이터를 올려두고, 서버 내에서는 객체 url 만 보관하여 효율적으로 데이터를 저장 가능하다.



클라이언트가 어떤 데이터를 전달받고, 어떤 방식으로 request 해야 하는지, 어떤 형식으로 데이터가 넘어오는지 한눈에 확인하기 위해 API 명세서를 작성했다.

API Docs ...						
<input checked="" type="checkbox"/> 구현	Aa Desc	method	route	path	담당	
<input checked="" type="checkbox"/>	🏠 홈 화면 조회	GET	home	/home	김동재	
<input type="checkbox"/>						
<input checked="" type="checkbox"/>	🚩 장소 리뷰 & 별점 조회	GET	region	/region/review	김동재	
<input checked="" type="checkbox"/>	🚩 상위 100개 장소 DB에 저장	POST	region	/region/review	김동재	
<input checked="" type="checkbox"/>	🚩 현재 지역 핫플레이스 조회	GET	region	/region	김동재	
<input checked="" type="checkbox"/>	🚩 지역별 핫플레이스 조회	GET	region	/region/:guname	김동재	
<input type="checkbox"/>						
<input checked="" type="checkbox"/>	💬 장소 상세정보 조회	GET	chat	/chat/detailPlace/:name	김동재	
<input checked="" type="checkbox"/>	💬 주차공간 상세정보 조회	GET	chat	/chat/detailParking/:parkingId	김동재	
<input type="checkbox"/>						
<input checked="" type="checkbox"/>	📅 17 오늘의 체험 일정 조회	GET	calendar	/calendar/event/today	김동재	
<input checked="" type="checkbox"/>	📅 17 날짜별 체험 일정 조회	GET	calendar	/calendar/event/date	김동재	

해당 API 명세를 하나씩 살펴보면,

- 홈 화면 조회 : [GET] /home

API 설명

홈 화면을 조회하여 "사람들이 많이 찾은 Top 10" 장소를 보여준다.

Request-Header

Name	Desc
Content-Type	application/json

Response-Body

Name	Type	Desc
(tags)	array	top 10 배열
_id	int	장소 id
_name	string	장소 이름

200 OK

```
{ // 홈 화면 조회
  "status": 200,
  "success": true,
  "message": "홈 화면 조회 성공",
  "data": [
    {
      "id": 0,
      "name": "동국 키즈카페",
    },
    {
      "id": 1,
      "name": "동국 키즈키즈",
    },
    ...
  ]
}
```

404 Not Found

```
{
  "status": 404,
  "success": false,
  "message": "존재하지 않는 데이터입니다"
}
```

500 Internal Server Error

```
{
  "status": 500,
  "success": false,
  "message": "서버 내부 오류"
}
```

- 장소 리뷰 & 별점 조회 : [GET] /region/review

API 설명

장소 리뷰 정보, 별점 정보, 방문자수를 조회한다.

Request-Header

Name	Desc
Content-Type	application/json

Response-Body

Name	Type	Desc
(tags)	array	장소 배열
_id	int	장소 id
_topplaceId	string	상위 100 테이블 id
_facilityId	string	사업장 id
_name	string	장소 이름
_rating	string	별점
_ratingCount	number	별점 개수
_reviews []	array	리뷰 배열
_reviewCount	number	리뷰 개수
_reviewDate []	array	리뷰 날짜

200 OK

```
{ // 장소 리뷰 & 별점 조회
  "status": 200,
  "success": true,
  "message": "장소 리뷰 & 별점 조회 성공",
  "data": [
    {
      "id": 0,
      "topplaceId": 11,
```

```

        "facilityId": 111,
        "name": "동국 놀이방",
        "rating": "5점",
        "ratingCount": 25,
        "reviews": ["좋아요", "매우 좋아요", "...],
        "reviewCount": 125,
        "reviewDate": ["2021년 2월 17일", "2021년 2월 18일",...]
    },
    {
        "id": 1,
        "topplaceId": 12,
        "facilityId": 113,
        "name": "동국 키즈방",
        "rating": "6점",
        "ratingCount": 25,
        "reviews": ["좋아요", "매우 좋아요", "...],
        "reviewCount": 125,
        "reviewDate": ["2021년 2월 17일", "2021년 2월 18일",...]
    },...
]
}

```

404 Not Found

```

{
  "status": 404,
  "success": false,
  "message": "존재하지 않는 데이터입니다"
}

```

500 Internal Server Error

```

{
  "status": 500,
  "success": false,
  "message": "서버 내부 오류"
}

```

- 상위 100 개 장소 DB 에 저장 : **[POST] /region/review**

API 설명

스코어 정렬로 장소 리뷰를 상위 100개 정렬한 후 db에 저장합니다.

Request-Header

Name	Desc
Content-Type	application/json

Request-Body

Name	Type	Desc
(tags)	array	상위 100개 장소 배열
_id	int	장소 id
_name	string	장소 이름

Response-Body

Name	Type	Desc
(tags)	array	상위 100개 장소 배열
_id	int	장소 id
_name	string	장소 이름

201 CREATED

```
{ // 상위 100개 장소 DB에 저장
  "status": 201,
  "success": true,
  "message": "상위 100개 장소 저장 성공",
  "data": [
    {
      "id": 0,
      "name": "붕붕 놀이방",
    },
    {
      "id": 1,
      "name": "붕붕 키즈카페",
    },
    ...
  ]
}
```

500 Internal Server Error

```
{
  "status": 500,
  "success": false,
```



```
"message": "서버 내부 오류"
}
```

- 현재 지역 핫플레이스 조회 : [GET] /region

API 설명

핫플레이스 뷰를 조회하여 현재위치 기반 핫한 장소를 보여줍니다.

Request-Header

Name	Desc
Content-Type	application/json

Response-Body

Name	Type	Desc
(tags)	array	지역별 랭킹 배열
_id	int	장소 id
_name	string	장소 이름
_placeImg	string	장소 이미지
_address	string	장소 주소
_telePhone	number	장소 전화번호
_operatingHour	string	장소 운영시간

200 OK

```
{ // 현재 지역 핫플레이스 조회
  "status": 200,
  "success": true,
  "message": "현재 지역 핫플레이스 조회 성공",
  "data": [
    {
      "id": 0,
      "name": "동국 놀이방",
      "placeImg": "이미지",
      "address": "서울시 중구 필동",
      "telePhone": 01012345678,
      "operatingHour": "09:00-20:00"
    },
    {
```

```

        "id": 1,
        "name": "동국 키즈카페",
            "placelmg": "이미지",
            "address": "서울시 중구 필동",
            "telePhone": 01012345678,
            "operatingHour": "09:00-20:00"
        },...
    ]
}

```

404 Not Found

```

{
    "status": 404,
    "success": false,
    "message": "존재하지 않는 데이터입니다"
}

```

500 Internal Server Error

```

{
    "status": 500,
    "success": false,
    "message": "서버 내부 오류"
}

```

- 지역별 핫플레이스 조회 : **[GET] /region/:guname**

API 설명

특정 지역의 핫플레이스 뷰를 조회하여 핫한 장소를 보여줍니다.

Request-Header

Name	Desc
Content-Type	application/json

Response-Body

Name	Type	Desc
(tags)	array	지역별 랭킹 배열
_id	int	장소 id
_name	string	장소 이름

_placelmg	string	장소 이미지
_address	string	장소 주소
_telePhone	number	장소 전화번호
_operatingHour	string	장소 운영시간

200 OK

```
{ // 지역별 핫플레이스 조회
  "status": 200,
  "success": true,
  "message": "지역별 핫플레이스 조회 성공",
  "data": [
    {
      "id": 0,
      "name": "상봉 놀이방",
      "placelmg": "이미지",
      "address": "서울시 중랑구 상봉동",
      "telePhone": 01012345678,
      "operatingHour": "09:00-20:00"
    },
    {
      "id": 1,
      "name": "상봉 키즈카페",
      "placelmg": "이미지",
      "address": "서울시 중랑구 상봉동",
      "telePhone": 01012345678,
      "operatingHour": "09:00-20:00"
    },
    ...
  ]
}
```

404 Not Found

```
{
  "status": 404,
  "success": false,
  "message": "존재하지 않는 데이터입니다"
}
```

500 Internal Server Error

```
{
  "status": 500,
  "success": false,
  "message": "서버 내부 오류"
}
```

- 지역별 장소 정보 조회 : [GET] /chat/detailPlace/:name

API 설명

지역에 해당하는 장소 정보를 조회한다.

Request-Header

Name	Desc
Content-Type	application/json

Request-Body

Name	Type	Desc
facilityId	int	사업장 id
_id	int	장소 id

Response-Body

Name	Type	Desc
(tags)	array	장소 배열
_id	int	장소 id
_topplaceId	string	상위 100 테이블 id
_facilityId	string	사업장 id
_name	string	장소 이름
_placeType	string	장소 종류
_placeImg	string	장소 이미지
_address	string	장소 주소
_telePhone	number	장소 전화번호
_detailUrl	string	장소 홈페이지 url
_operatingHour	string	장소 운영시간
_fee	string	장소 요금

200 OK

```
{ // 지역별 장소 정보 조회
  "status": 200,
```

```

"success": true,
"message": "지역별 장소 정보 조회 성공",
"data": [
    {
        "id": 0,
        "topplaceId": 11,
        "facilityId": 111,
        "name": "동국 놀이방",
        "placeType": "휴게음식점",
        "placeImg": "이미지",
        "address": "서울시 중구 필동",
        "telePhone": 01012345678,
        "detailUrl": "장소 홈페이지 url",
        "operatingHour": "09:00-20:00",
        "fee": "10,000원"
    },
    {
        "id": 1,
        "topplaceId": 12,
        "facilityId": 113,
        "name": "동국 놀이방2",
        "placeType": "휴게음식점",
        "placeImg": "이미지",
        "address": "서울시 중구 필동",
        "telePhone": 01012345678,
        "detailUrl": "장소 홈페이지 url",
        "operatingHour": "09:00-20:00",
        "fee": "10,000원"
    },...
]
}

```

404 Not Found

```

{
    "status": 404,
    "success": false,
    "message": "존재하지 않는 데이터입니다"
}

```

500 Internal Server Error

```
{
  "status": 500,
  "success": false,
  "message": "서버 내부 오류"
}
```

- 지역별 주차공간 정보 조회 : [GET] /chat/detailParking/:parkingId

API 설명

지역에 해당하는 주차공간 정보를 조회한다.

Request-Header

Name	Desc
Content-Type	application/json

Request-Body

Name	Type	Desc
address	String	주차공간 주소

Response-Body

Name	Type	Desc
(tags)	array	주차공간 배열
_id	int	주차공간 id
_name	string	주차공간 이름
_address	string	주차공간 주소
_parkingCode	string	주차공간 코드
_parkingType	string	주차공간 종류
_opName	string	주차공간 운영구분명
_payYN	boolean	유무료 구분
_nightFreeYN	boolean	야간 유무료 구분
_wDayStart	string	평일 운영 시작시간
_wDayClose	string	평일 운영 종료시간
_wEndStart	string	주말 운영 시작시간
_wEndClose	string	주말 운영 종료시간
_hDayStart	string	공휴일 운영 시작시간
_hDayClose	string	공휴일 운영 종료시간
_satPayYN	boolean	토요일 유무료 구분

_holPayYN	boolean	공휴일 유무료 구분
_fee	number	기본 주차요금
_time	number	기본 주차시간
_addFee	number	추가 단위요금
_addTime	number	추가 단위시간
_dayMax	number	일 최대요금

200 OK

```
{ // 지역별 주차공간 정보 조회
  "status": 200,
  "success": true,
  "message": "지역별 주차공간 정보 조회 성공",
  "data": [
    {
      "id": 0,
      "name": "동국 주차장",
      "address": "서울시 중구 필동",
      "parkingCode": "12321",
      "parkingType": "공터주차장",
      "opName": "운영~~",
      "payYN": true,
      "nightFreeYN": true,
      "wDayStart": "09:00",
      "wDayClose": "20:00",
      "wEndStart": "09:00",
      "wEndClose": "20:00",
      "hDayStart": "09:00",
      "hDayClose": "20:00",
      "satPayYN": false,
      "holPayYN": false,
      "fee": 10000,
      "time": 30,
      "addFee": 3000,
      "addTime": 10,
      "dayMax": 30000
    },
    {
      "id": 1,
      "name": "혜화 주차장",
      "address": "서울시 중구 필동",
```

```

        "parkingCode": "12321",
        "parkingType": "공터주차장",
        "opName": "운영~~",
        "payYN": true,
        "nightFreeYN": true,
        "wDayStart": "09:00",
        "wDayClose": "20:00",
        "wEndStart": "09:00",
        "wEndClose": "20:00",
        "hDayStart": "09:00",
        "hDayClose": "20:00",
        "satPayYN": false,
        "holPayYN": false,
        "fee": 10000,
        "time": 30,
        "addFee": 3000,
        "addTime": 10,
        "dayMax": 30000
    },...
]
}

```

404 Not Found

```

{
    "status": 404,
    "success": false,
    "message": "존재하지 않는 데이터입니다"
}

```

500 Internal Server Error

```

{
    "status": 500,
    "success": false,
    "message": "서버 내부 오류"
}

```

- 지역별 문화행사 정보 조회 : **[GET] /chat/event/:address**

API 설명

지역에 해당하는 문화행사 정보를 조회한다.

Request-Header

Name	Desc
Content-Type	application/json

Request-Body

Name	Type	Desc
address	String	문화행사 주소

Response-Body

Name	Type	Desc
(tags)	array	문화행사 배열
_id	int	문화행사 id
_topplaceId	string	상위 100 테이블 id
_name	string	문화행사 이름
_codeName	string	분류명
_guName	string	자치구
_date	string	날짜&시간
_address	string	문화행사 주소
_target	string	이용대상
_fee	number	이용요금
_cast	string	출연정보
_programInfo	string	프로그램소개
_homeUrl	string	홈페이지 주소
_startDate	string	행사 시작일
_endDate	string	행사 종료일
_themaCode	string	테마분류

200 OK

```
{ // 지역별 문화행사 정보 조회
  "status": 200,
  "success": true,
  "message": "지역별 문화행사 정보 조회 성공",
  "data": [
    {
      "id": 0,
      "topplaceId": 11,
      "facilityId": 111,
      "name": "동국 놀이방",
      "placeType": "휴게음식점",
```

```

        "placeImg": "이미지",
        "address": "서울시 중구 필동",
        "telePhone": 01012345678,
        "detailUrl": "장소 홈페이지 url",
        "operatingHour": "09:00-20:00",
        "fee": "10,000원"
    },
    {
        "id": 1,
        "topplaceId": 12,
        "facilityId": 113,
        "name": "동국 놀이방2",
        "placeType": "휴게음식점",
        "placeImg": "이미지",
        "address": "서울시 중구 필동",
        "telePhone": 01012345678,
        "detailUrl": "장소 홈페이지 url",
        "operatingHour": "09:00-20:00",
        "fee": "10,000원"
    },...
]
}

```

404 Not Found

```

{
    "status": 404,
    "success": false,
    "message": "존재하지 않는 데이터입니다"
}

```

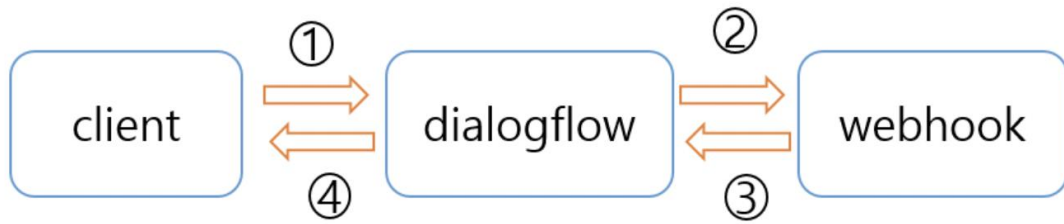
500 Internal Server Error

```

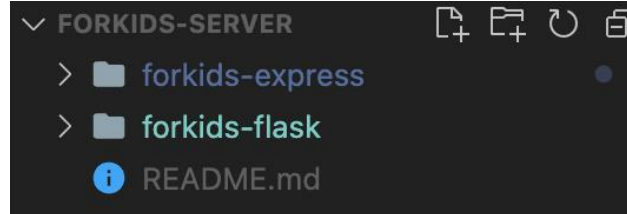
{
    "status": 500,
    "success": false,
    "message": "서버 내부 오류"
}

```

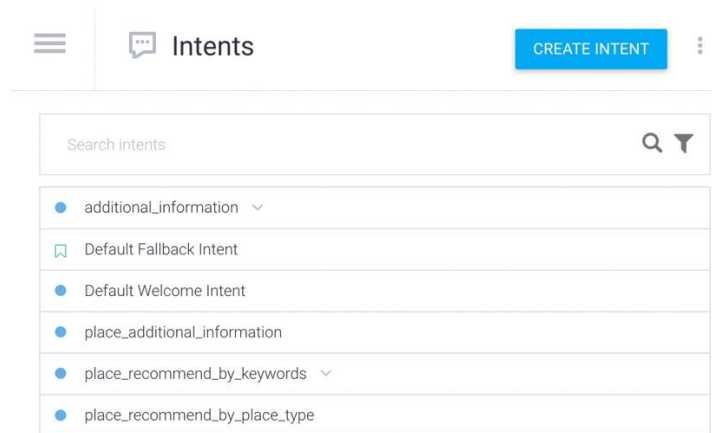
3.3.4 Flask Webhook Server



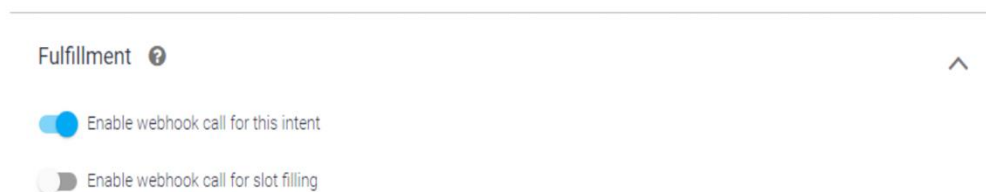
Webhook 은 모델에서 정해진 답변 이외의 사용자가 원하는 동적인 답변을 위의 플로우로 도출해낸다.



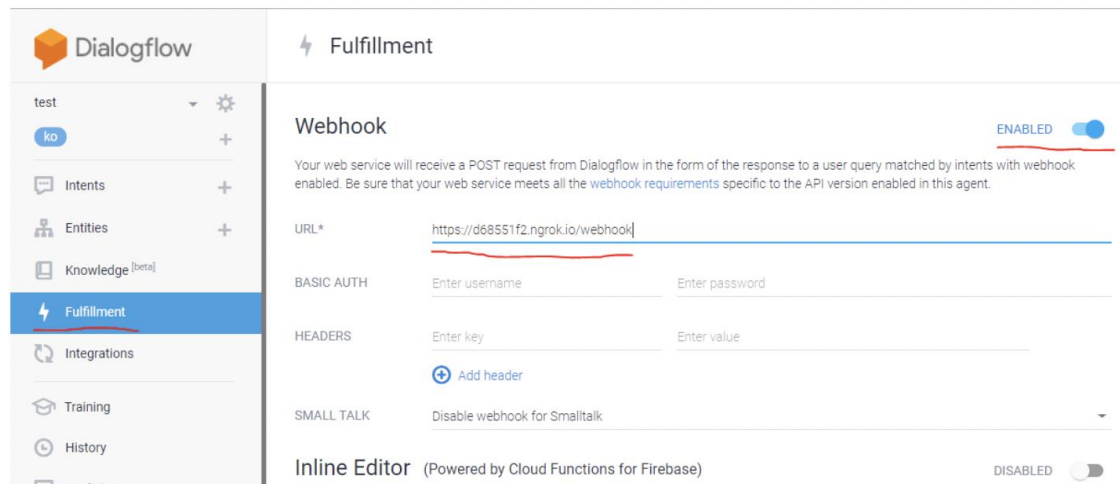
Flask 서버 작업공간을 만들어 flask 모듈을 설치하고



생성한 intent 에서 intents Fulfillment Enable 을 설정했다.



Fulfillment webhook 으로 Dialogflow 모델 API 를 호출할 수 있는 URL 을 생성했다.

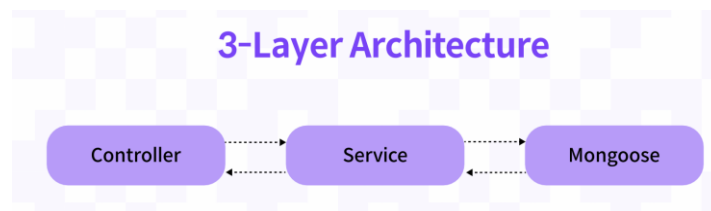


이후 Flask 서버에서 ngrok http 5000 을 사용하여 내부 ip 를 외부 ip 로 바꿔서 사용했다.
해당 서버에서 이후 데이터파트 모듈에서 설명될 알고리즘 API 들을 호출할 수 있다.

3.2.5 Node.js Web Server

validation 을 위한 npm 라이브러리인 express validator 를 사용했다.

Express 서버 설계는 3-Layer Architecture 로 다음과 같이 비즈니스 로직을 Route Controller 에서 분리하는 방식으로 설계하였다.



-Controller : Model 이 처리하기 전에 1 차 데이터 가공, 비즈니스 로직 수행 후 전달받은 데이터를 Response

구현 ex) HomeController

```

forkids-express > src > controllers > HomeController.ts
8  /**
9   * @route GET /home
10  * @desc Read MainPage
11  * @access Public
12  */
13  const getHome = async (req: Request, res: Response) => {
14    const error = validationResult(req);
15    if (!error.isEmpty()) {
16      return res
17        .status(statusCode.BAD_REQUEST)
18        .send(util.fail(statusCode.BAD_REQUEST, message.BAD_REQUEST));
19    }
20
21    try {
22      const data = await HomeService.getHome();
23      if (!data) {
24        return res
25          .status(statusCode.NOT_FOUND)
26          .send(util.fail(statusCode.NOT_FOUND, message.NOT_FOUND));
27      }
28
29      res
30        .status(statusCode.OK)
31        .send(util.success(statusCode.OK, message.READ_HOME_SUCCESS, data));
32    } catch (error) {
33      console.log(error);
34      res
35        .status(statusCode.INTERNAL_SERVER_ERROR)
36        .send(
37          util.fail(

```

- Service : Controller 에서 받아온 데이터를 가지고 가공하여 DB 에 접근해서 결과값을 받아옴.

구현 ex) HomeService

```

import TopPlace from '../models/TopPlace';
import { TopPlaceResponseDto } from '../interfaces/topplace/TopPlace';

const getHome = async (): Promise<TopPlaceResponseDto[] | null> => {
  try {
    const topplaces = await TopPlace.find().limit(10);

    if (!topplaces) {
      return null;
    }

    const data = topplaces.map(topplace => ({
      id: topplace.id,
      name: topplace.name,
    }));

    return data;
  } catch (error) {
    console.log(error);
    throw error;
  }
};

export default {
  getHome,
};

```

- Router : 엔드포인트로 들어온 Request 를 Controller 내부 함수가 처리하도록 연결

구현 ex) HomeRouter

```
forkids-express > src > routes > HomeRouter.ts
1 import { Router } from 'express';
2 import HomeController from '../controllers/HomeController';
3
4 const router: Router = Router();
5
6 router.get('/', HomeController.getHome);
7
8 export default router;
```

- 계층 간 데이터 교환을 위해 Interface의 DTO를 사용

✓ Request <-> Controller <-> Service Layer <-> Response 의 데이터 교환

- API Request/Response 시에 mongoose ODM 사용하여 객체 데이터와 DB document 를 매핑

3.2.6 Client-Server HTTP

AWS EC2 의 배포를 위해 HTTP, HTTPS 접속을 허용시키고 인스턴스를 시작했다.



인스턴스 ssh 명령어를 복사, 터미널에 key file 위치로 이동하여 EC2 에 접속한다. 이때, chmod 400 명령어를 사용하여 key file 을 보호해준다.

```
Warning: Permanently added 'ec2-3-39-222-248.ap-northeast-2.compute.amazonaws.com' (ED25519) to the list of known hosts.
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 5.4.0-1071-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of Thu May  5 17:50:32 UTC 2022

System load:  0.0          Processes:    93
Usage of /:   15.5% of 7.69GB   Users logged in:  0
Memory usage: 18%          IP address for eth0: 172.31.1.244
Swap usage:   0%

0 updates can be applied immediately.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-1-244:~$
```

* 이후 node, nvm, yarn을 차례로 설치해준다

forkids 서버 프로젝트가 있는 git repository를 ubuntu 서버에 clone 받아준다. 아래와 같이 clone 받은 뒤 실행 스크립트인 yarn run dev를 입력하면 배포가 완료된다.

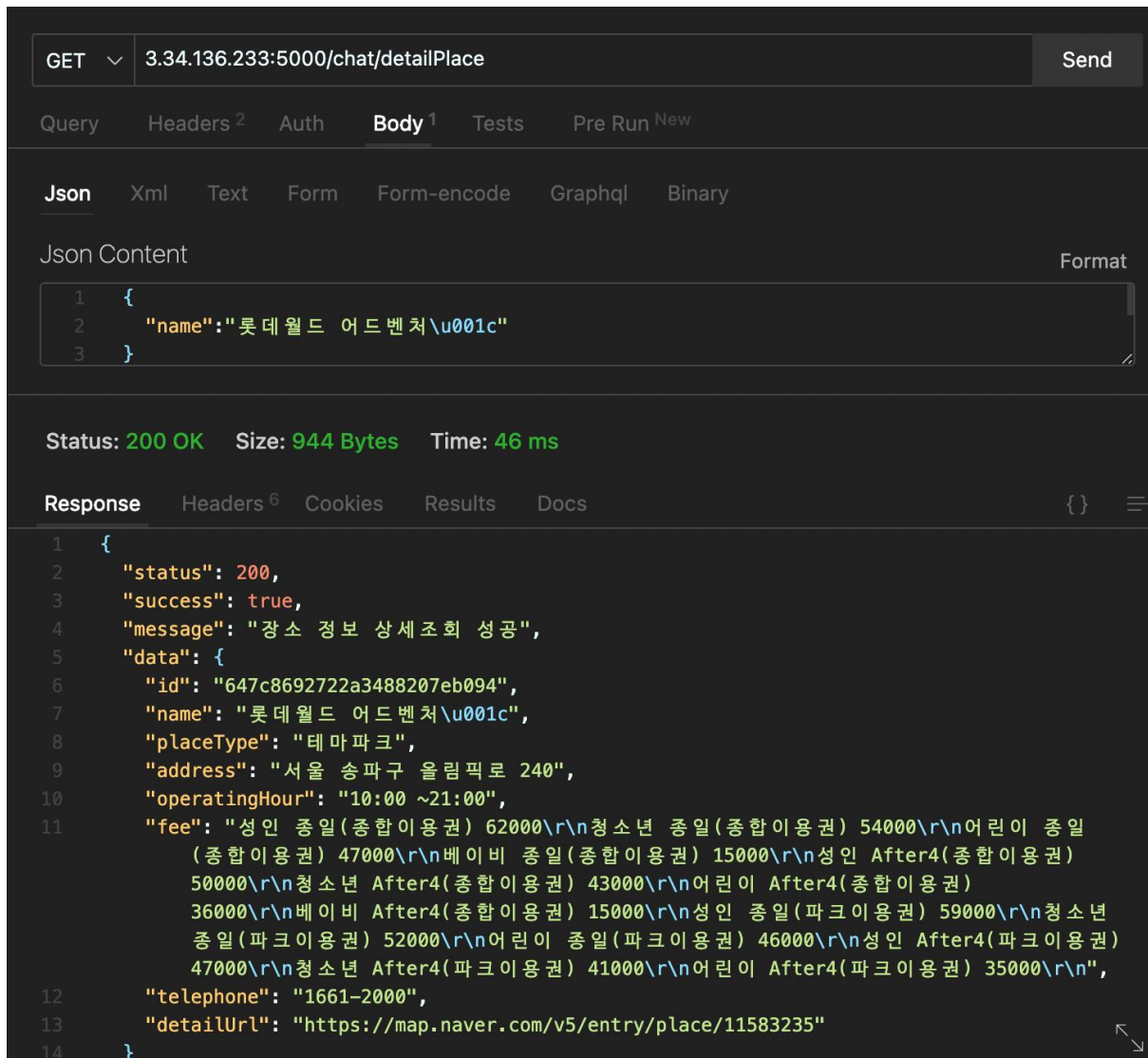
```
ubuntu@ip-172-31-1-244:~$ git clone https://github.com
Cloning into 'Jobchae'...
remote: Enumerating objects: 169, done.
remote: Counting objects: 100% (169/169), done.
remote: Compressing objects: 100% (119/119), done.
remote: Total 169 (delta 55), reused 140 (delta 37), p
Receiving objects: 100% (169/169), 55.12 KiB | 11.02 M
Resolving deltas: 100% (55/55), done.
```

배포 이후 EC2 접속과 상관 없이 서버를 실행하기 위해 Node 애플리케이션을 쉽게 관리할 수 있게 하는 pm2를 사용하여 무중단 배포를 했다.

```
ubuntu@ip-172-31-34-134:~/forkids-server/forkids-express$ pm2 start dist
[PM2] Applying action restartProcessId on app [dist](ids: [ 0 ])
[PM2] [dist](0) ✓
[PM2] Process successfully started
```

id	name	mode	⌵	status	cpu	memory
0	dist	fork	1	online	0%	19.6mb

Ubuntu 내에서 서버 빌드 후에 dist를 pm2로 시작해 주면, EC2에 접속해 있지 않아도 통신이 가능하다.



3.3 Front-end

JavaScript 기반 React Native 프레임워크를 사용하여 개발의 편의성을 노리고, 짧은 개발기간 내에 목표한 기능 구현이 가능하도록 하였다. 백엔드와 데이터분석 파트 간의 협업을 위해 git 으로 버전을 관리하였으며, 통신 API 로 axios 를 사용하였다.

3.3.1 앱 화면 구성 및 플로우

0) OverView

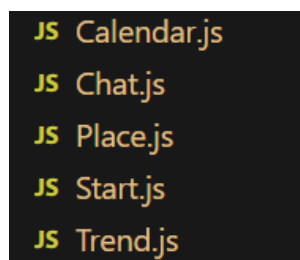
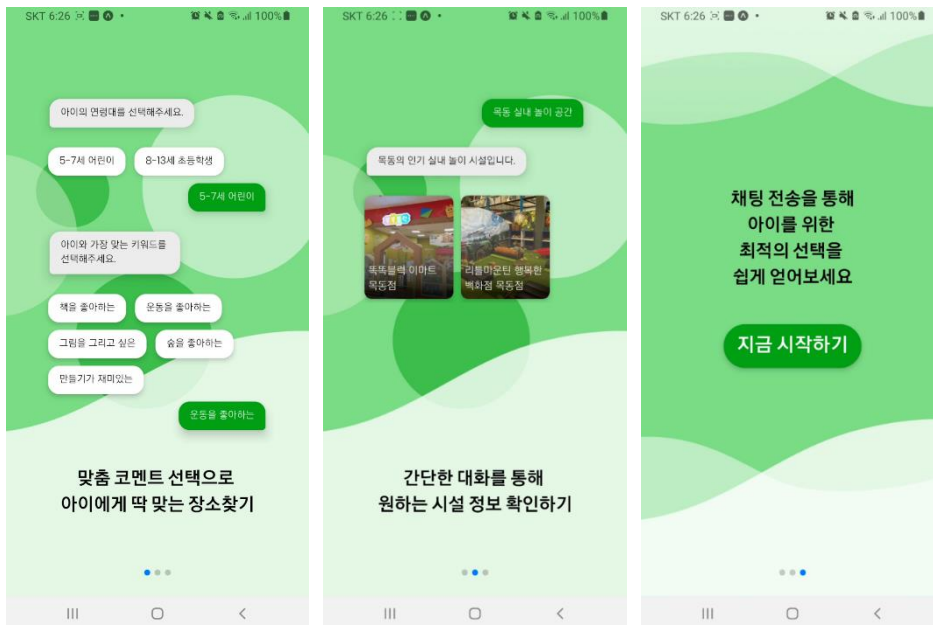


그림 1 "./screens"


JS name	Feature	Used Component
Start	초기 시작 시 튜토리얼 화면을 보여준다. Swiper을 이용하여 3개의 뷰를 보여주고, 이 화면을 보고 난 후 사용자가 Chat 화면으로 넘어가도록 한다.	<Swiper> - 기존 내장 <MessageBubble> <QuickReplyContainer> <PlaceContainer>
Trend	다른 사용자들이 가장 좋은 후기와 별점을 준 장소 리스트를 보여준다. 클릭 시 해당 장소의 정보를 보여주는 모달 창이 띄워진다.	<TrendBlock>
Chat	Nav의 디폴트 화면. 사용자가 채팅을 통해 챗봇의 답변을 얻을 수 있도록 하는 화면이다. 시설 정보, 주차 정보, 근처 카페 정보 얻기와 장소추천받기 플로우 기능을 제공한다.	<MessageBubble> <RecommendContainer> <QucikReplyContainer> <PlaceContainer>
Place	지역별 추천 장소 리스트를 보여주는 화면. 장소를 구 별로 선택하여 해당 구 내의 인기 장소 리스트를 보여준다. 클릭시 Trend의 장소 블록과 동일하게 상세 정보 모달 창을 보여준다.	<PlaceBlock> <SelectLoca>
Calendar	날짜별로 운영하는 문화 행사정보를 보여주는 화면. 날짜를 선택 시 해당 날짜에 진행되는 문화 행사 리스트를 보여주며, 해당 행사를 클릭 시 관련 사이트 url로 가게 된다.	<CalendarPicker> - Librar y <EventBlock>

1) Start

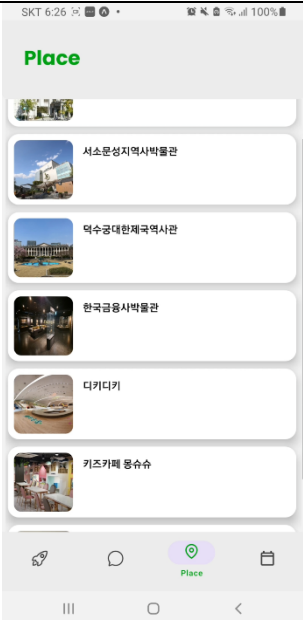


사용자에게 챗의 활용을 위한 가이드 텍스트 리스트를 보여주는 화면이다. 또한 팀에서 제공하는 서비스가 목표로 하는 편의를 사용자에게 보여주어 해당 앱에서 채팅을 적극적으로 활용할 수 있도록 유도한다.

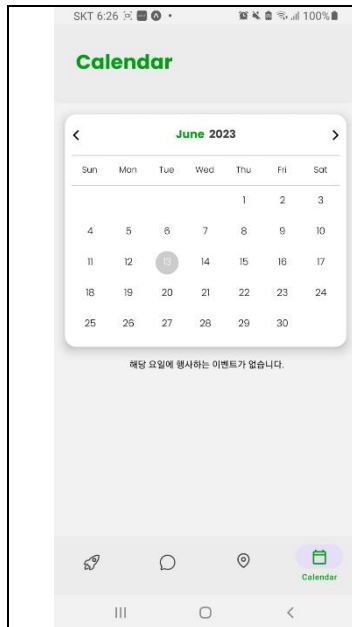
2) Trend

	<p>Trend 화면에서는 전체 사용자들이 매긴 별점, 후기 등을 참조하여 인기있는 장소 리스트를 사용자에게 보여준다. '/home'으로 GET요청을 통해 받은 장소 리스트는 <TrendBlock> 컴포넌트를 사용하며 각 컴포넌트 내의 place는 name, location, runningHour, telephone 과 같은 기본적인 정보를 내제하고 있다. 해당 컴포넌트를 클릭 시 <PlaceModal>이 화면에 띄워지도록 설계하였다.</p>
---	---

3) Place

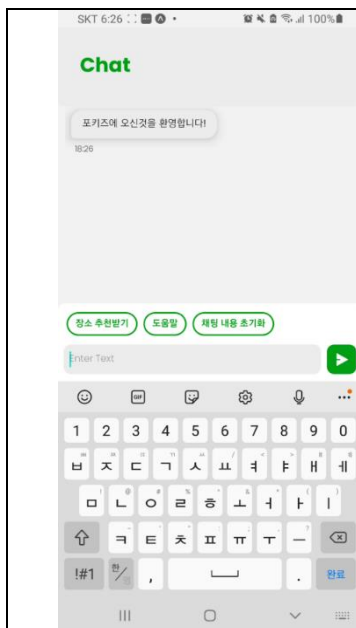
	<p>Place 화면에서는 사용자의 위치 기반 또는 설정한 위치를 기반으로 하여 구 별로 장소를 필터링에 보여준다. '/region'에서 받은 장소 리스트는 기본적으로 현재 위치 기반으로 장소가 필터링 되며, 상단의 Place 버튼을 클릭하여 장소를 설정하면 '/region/:guname'으로 데이터 요청을 하여 <PlaceBlock> 컴포넌트를 통해 리스트 내의 아이템들의 정보를 보여준다. Trend 뷰와 마찬가지로 블록을 클릭시 <PlaceModal> 컴포넌트를 통해 상세 정보를 사용자가 얻을 수 있도록 설계하였다.</p>
--	---

4) Calendar





Calendar 화면에서는 <CalendarPicker>에서 선택한 날짜를 'calendar/event'와 함께 request body에 포함시켜 데이터를 요청하여 받은 문화행사를 하단의 <EventBlock> 컴포넌트를 포함한 리스트 뷰에 보여준다. <EventBlock>은 name, location, url 이 포함된다. 블록 클릭시 연결된 url로 자주 사용하는 브라우저를 통해 창이 열리게 된다.

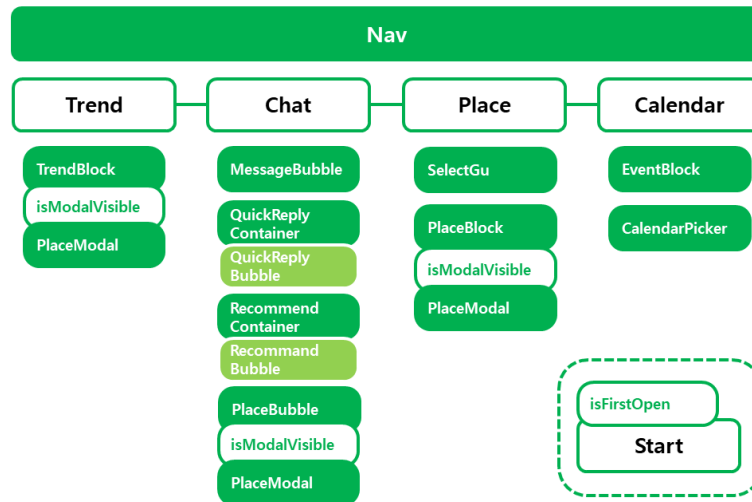
5) Chat



Chat 1) 기본 화면 시작 시 환영 메시지가 화면에 띄워진다. <RecommendContainer>로 가장 주요 기능인 '장소 추천받기'와 사용자의 텍스트 입력 가이드를 주는 '도움말', 채팅 내용을 초기화 할 수 있는 '채팅 내용 초기화'가 들어있으며, 하단의 inputText에 챗을 입력하여 챗봇에게 특정 정보를 요구할 수 있다. 챗봇의 답변은 Webpack을 이용하여 Dialogflow의 답변을 가져와 messages 리스트에 반영되며 메시지는 render Messages()를 호출하여 뷰에 출력되도록 한다.





	<p>Chat 2) '장소 추천받기'를 챗봇에 전달하였을 때의 플로우다. 아이의 연령대와, 아이에게 어울리는 키워드를 선택하도록 사용자에게 <QuickReplyContainer>로 <QucikReplyBubble>들을 messages에 추가하여 렌더링한다. 추후에 키워드 리스트를 수정할 수 있으며, 추가 QuickReplyContainer을 넣어 키워드를 추가로 선택할 수 있도록 플로우를 수정할 수 있다. 선택한 키워드는 백엔드를 거쳐 분석 알고리즘의 데이터로 쓰이고, 알고리즘의 결과물을 화면에 출력한다.</p>
	<p>Chat 3) 알고리즘을 통한 추천 리스트는 <PlaceContainer>을 통해 <PlaceBubble>이 messages값을 통해 렌더링되며, name과 image, 모달창을 위한 상세 정보들이 포함되어 messages에 추가된다. 장소 버블을 클릭하면 다른 뷰에서와 동일하게 상세 정보 모달 창이 화면에 띄워지게 된다. '도움말' 클릭 시 서비스가 기대하는 사용자의 텍스트 입력 가이드를 보여준다.</p>


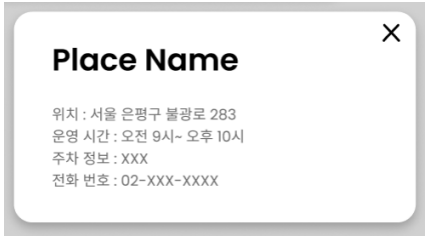
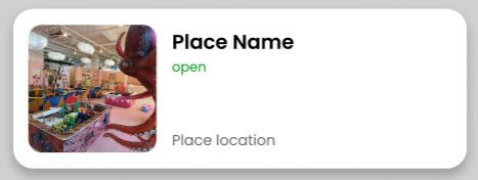
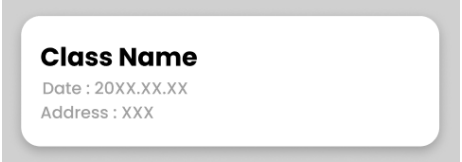

6) Key features



3.3.3 File Sets

1) Components

Componet	Props
	<pre>{ //TrendBlock id : String, name : String, image : String, }</pre>
	<pre>{ //MessageBubble text : String, user : Boolean, isSent : Boolean, isQuickReply? : Boolean, date : getCurrentTime(), options : Array[] isPlaceImage? : Boolean, places : Array[] }</pre>
	<pre>{ //PlaceBubble id : String image : url, name : String, }</pre>
	<pre>{ //QuickReplyBubble key : Integer,</pre>

	<pre> name : String, onOptionPress : handleQuickReply } </pre>
	<pre> { //RecommandBubble key : Integer, name : String, } </pre>
	<pre> { //PlaceModal name : String, address : String, runninghour: String, parking : String, telephone : String, onModalPress : handleModalVisible(), } </pre>
	<pre> { //PlaceBlock id : String, name : String, address : String, } </pre>
	<pre> { //EventBlock id : String, name : String, startDate : String, address : String } </pre>
	: bottom tab navigation

2) Assets

- fonts

UI 디자인에 필요한 폰트를 지정한다.

Poppins_bold

Poppins_extrabold

Poppins_regular

Poppins_semibold

Prompt_bold

폰트를 앱에 사용하기 위해 App.js에서 fontsLoaded로 폰트 로드 상태를 관리하였다.

```
const [fontsLoaded, setFontsLoaded] = useState(false);
const loadFonts = async () => {
  await Font.loadAsync({
    Poppins_regular: require("./assets/fonts/Poppins_regular.ttf"),
    Poppins_bold: require("./assets/fonts/Poppins_bold.ttf"),
    Poppins_semibold: require("./assets/fonts/Poppins_semibold.ttf"),
    Poppins_extrabold: require("./assets/fonts/Poppins_extrabold.ttf"),
    Prompt_bold: require("./assets/fonts/Prompt_bold.ttf"),
  });
  setFontsLoaded(true);
};
```

- png files

디자인에 필요한 이미지 파일들과 splash, icon 이미지를 포함한다.

- dummy : 디버그 테스트를 위한 이미지 데이터

3.3.4 RESTful HTTP Client

axios를 기반으로 한 RESTful api를 통해 서버에서 데이터를 얻었다.

```
const getData = async () => {
  axios
    .get(`${baseUrl}region/${guchoice}`)
    .then((response) => {
      const responseData = response.data;
      setPlaces(responseData.data);
    })
    .catch((error) => {
      // 에러 처리
      console.error(error);
    });
};
```

3.4Data

3.4.1. Data Collection (예진)

고객의 리뷰 데이터를 얻기 위해, 수집한 장소의 사업장명을 네이버지도에 검색하여 각 장소의 상세 페이지로 이동했다. 상세 페이지에서는 해당 장소의 방문자 리뷰, 방문자 리뷰 수, 키워드 리뷰 정보와

장소타입, 운영시간, 이용료, 별점, 별점 평가 수, 상세 장소 페이지 URL, 대표 이미지 등 부가 정보를 확인할 수 있다. 각 장소 페이지마다 이러한 정보를 크롤링하여 수집했다.

추가로, 네이버에 '서울 아이와 함께', '서울 가족여행', '서울 주말 나들이' 키워드를 네이버지도에 검색하여 각 장소의 상세 페이지로 이동했다. 각 장소 페이지마다 위와 동일한 과정을 거쳐 칼럼별로 크롤링하여 데이터를 수집했다.

이 전체 과정을 통해 장소에 대한 기본 정보와 리뷰, 부가 정보를 포함한 데이터를 수집했다. 이러한 데이터는 추후 챗봇 시스템에서 적합한 장소를 추천하는 데 활용될 것이다.

1 차 수정: 'detail_url' 칼럼에 사업장 홈페이지 url 을 저장하는 부분을 네이버지도 해당 장소 상세페이지 url 을 저장하도록 수정했다.

2 차 수정: 크롤러 실행 시 이전에 수집한 데이터와 중복하여 결과가 저장되는 오류가 발생하였다.

3 차 수정: 대표이미지가 수집되지 않고, 정확하지 않은 전화번호가 수집되는 오류가 발생하였다.

4 차 수정: IP 주소를 지정하여 관리자 권한으로 접근 하는 방식으로 변경하였다.

3.4.2. Data Preprocessing(서영, 예진)

사용한 라이브러리:

데이터 전처리를 위해 다음 파이썬 라이브러리를 사용했다.

- numpy
- pandas
- gensim 의 word2vec
- konlpy 의 Okt

```
combine_df.columns = ['name','place_type','place_img','address','telephone','detail_url','operating_hours','fee','rating','rating_count','reviews','review_date','review_count','keyword_reviews']
```

데이터 형식화: 수집한 데이터를 ['name', 'place_type', 'place_img', 'address', 'telephone', 'detail_url', 'operating_hours', 'fee', 'rating', 'rating_count', 'reviews', 'review_date', 'review_count', 'keyword_reviews'] 열 형식에 맞춰 데이터프레임으로 저장했다.

```
# 키워드 리뷰 없는 행 제거

print(combine_df['keyword_reviews'].isnull().sum())

combine_df.dropna(subset=['keyword_reviews'], inplace=True)

# 키워드 리뷰 '-'인 행 제거

print((combine_df['keyword_reviews'] == '-').sum())

combine_df = combine_df[combine_df['keyword_reviews'] != '-']
```



```

# 가게 이름 중복 제거

combine_df.drop_duplicates(subset='name', keep='first', inplace=True)

# 주소 서울 아닌 행 제거

combine_df = combine_df[combine_df['address'].str.startswith('서울')]

# 전체 장소 타입 확인

unique_place_types = combine_df['place_type'].unique()

unique_place_types

# 관련 없는 장소 타입 제거

remove_types = ['요리주점', '피부,체형관리', '귀금속,시계', '관리업', '결혼예물', '필라테스', 'KT', '자동차정비,수리', '안경원', 'PC방', '순대,순댓국', '미용실', '천연화장품', '이자카야', '반찬가게', '스터디카페', '네일아트, 네일샵', '바(BAR)', '스크린골프장', '금제품전문', '와인']

combine_df = combine_df[~combine_df['place_type'].isin(remove_types)]

```

```

array(['테마파크', '박물관', '아쿠아리움', '동물원', '시민공원', '워터파크', '관람,체험', '요트',
      '동물카페', '피자', '카페,디저트', '편의점', '베이커리', '교습학원,교습소', '샌드위치', '카페',
      '김밥', '33떡볶이', '만화방', '종합분식', '도넛', '베트남음식', '셀프,대여스튜디오', '호두과자',
      '과일,주스전문점', '육류,고기요리', '케이크전문', '덮밥', '와플', '아이스크림', '종합도소매', '빙수',
      '브런치', '테이크아웃커피', '가방,핸드백', '다이어트,샐러드', '애견카페', '일식당', '가공식품',
      '치킨,닭강정', '떡볶이', '차', '국수', '푸드트럭', '과자,사탕,초코렛', '만두', '햄버거', '한식',
      '바나프레소', '이용원', '과일', '떡,한과', '슈퍼,마트', '공방', '토스트', '포장마차', '분식',
      '라면', '맥주,호프', '헬스장', '도서,음반,문구', '차,커피', '돈가스', '중식당', '크레페',
      '원예,화훼농원', '호떡', '다방', '종합패션', '베이글', '패션', '보드카페', '꽃집,꽃배달',
      '인도음식', '아이스링크', '인테리어소품',
      (중간생략)
      '곱창,막창,양', '가구', '호텔', '양꼬치', '침구,커튼', '수입의류', '미술,공예품', '유아,아동용품',
      '침구류,직물', '건축,토목', '아경면선', '오므라이스', '스포츠용품', '소고기구이', '시계', '시장',

```

```
'양말', '스포츠,오락', '체험마을', '전망대', '테마공원', '자연공원', '영,유아,아동'],  
dtype=object)
```

의미없는 데이터 및 이상치 제거: 먼저, 리뷰 데이터 값이 비어있거나 0, -, null 등으로 표시된 행을 제거했다. 이때, 리뷰가 아닌 부가 정보가 비어있는 경우에는 해당 값을 0으로 채워주었다. 주소가 서울이 아닌 잘못 수집된 데이터도 제거했다. 추가로, 전체 장소 타입 중 관련 없는 장소 타입에 해당하는 행을 제거했다.

```
train_data = combine_df['keyword_reviews']  
  
tokenized_data=[]  
  
for sentence in train_data:  
    keywords = sentence.split('WrWn')  
    tokenized_data.append(keywords)  
  
combine_df['tag'] = tokenized_data  
  
split_data=[]  
  
for sentence in place_df['keyword_reviews']:  
    keywords = sentence.split('WrWn')  
    split_data.append(keywords)  
  
place_df['tag'] = split_data  
  
model = Word2Vec(sentences=tokenized_data, vector_size=100, window = 5, min_count = 5, workers = 4, sg = 0)
```

훈련 데이터셋 구성: 전처리가 완료된 약 5,000 여 개의 장소 데이터를 훈련 데이터셋으로 사용했다. 각 장소의 키워드 리뷰를 키워드 별로 분할하여 word2vec 모델에 학습시켰다.

```
unique_place_types = p_df['place_type'].unique()
```

```
array(['테마파크', '박물관', '아쿠아리움', '동물원', '시민공원', '워터파크', '관람,체험', '요트',  
      '동물카페', '스포츠,오락', '키즈카페,실내놀이터', '캠핑,야영장', '식물원,수목원', '체험마을', '전  
      망대',  
      '테마공원', '자연공원', '영,유아,아동'], dtype=object)
```

추천 장소 데이터 필터링: 추천 장소 데이터에는 아이와의 놀이 장소로 적합하지 않은 PC 방, 요리주점, 스터디카페, 유흥업소, 음식점, 카페 등의 장소 타입을 제거한 데이터를 사용했다.

Adding latitude and longitude

장소를 추천한 후, 해당 장소에 대해 가까운 거리에 있는 주차장과 카페를 추천해주는 기능을 구현하기 위해 p_data, top100, parking_data, cafe_data 데이터셋에 위도와 경도 컬럼을 추가하였다. geopy.geocoders 패키지를 이용해서 각각의 도로명 주소를 지오코딩하여 위도와 경도 좌표로 변환하였고, 변환한 좌표를 리스트로 만들어 위도는 latitude, 경도는 longitude 의 이름으로 데이터프레임의 컬럼으로 추가하였다.

```
# import package

from geopy.geocoders import Nominatim

geo_local = Nominatim(user_agent='South Korea')

#function

def geocoding(address):

    try:

        geo = geo_local.geocode(address)

        x_y = [geo.latitude, geo.longitude]

        return x_y

    except:

        return [0,0]

latitude = []

longitude =[]

for i in address:

    latitude.append(geocoding(i)[0])

    longitude.append(geocoding(i)[1])
```

```
# Adding columns of latitude and longitude
```

```
example_data = pd.DataFrame({'name': df_example['name'], 'place_img': df_example['place_img'], 'address': df_example['address'], 'telephone': df_example['telephone'], 'detail_url': df_example['detail_url'], 'latitude': latitude, 'longitude': longitude})
```

3.4.3. Dataset

p_data: 서울에 위치한 장소 목록

- p_data.csv 파일에 포함되어 있다.
- 서울 지역에 위치한 다양한 장소의 정보를 담고 있다.
- 데이터 포맷: CSV
- 컬럼: name, place_type, place_img, address, telephone, detail_url, operating_hours, fee, rating, rating_count, reviews, review_date, review_count, keyword_reviews, tag, target_age, latitude, longitude

p_data는 추천할 장소 데이터를 모아둔 데이터셋으로 총 133 개의 장소에 대한 데이터가 담겨있다. 네이버에서 '서울 아이와 함께'와 '서울 키즈카페'를 검색했을 때 나오는 네이버맵 크롤링 결과 데이터와 서울 열린 데이터 광장의 '서울시 일반야영장업 인허가 정보', '서울시 유원시설업(기타) 인허가 정보' 데이터를 병합하고 전처리하여 생성하였다. 추천 장소를 서울시로 제한하였기 때문에 서울시의 데이터만 수집하였고, 챗봇을 통해 나이와 키워드를 바탕으로 장소를 추천할 때 사용하는 데이터셋이다. 아래는 p_data의 샘플 데이터 10 개를 캡처한 그림과 칼럼 이름에 대한 내용을 담은 표이다.

	name	place_type	address	telephone	detail_url	operating_hours	fee	rating	rating_count	reviews	review_date	review_count	keyword_reviews	tag	target_age	latitude	longitude
0	롯데월드 (테마파크)	서울 송파	1661-2000	https://map	10:00 ~ 21:00	성인 종일		4.47	35269	•티켓은 두 4.27.목 3.31.금		37693	놀이기가 다양해요	[놀이기가 다양해요]	37.5111	127.0982	
1	서울어린이0테마파크	서울 광진	02-450-9311	https://map	10:00 ~ 18:30	공원입장료		4.57	3950	어른인 나! 5.11.목 4.30.일		5205	아이와 가기 좋아요	[아이와 가기 좋아요]	37.53403	127.0678	
2	국립중앙박물관	서울 용산	02-2077-900	https://map	10:00 ~ 18:00	관람료 0		4.53	672	국립중앙2.6.월 3.3.금 3.		787	유익해요	전시 구성이 [유익해 high]	37.52395	126.9803	
3	코엑스아쿠아리움	서울 강남	0507-1435-7	https://map	10:00 ~ 20:00	성인₩4000		4.5	34266	옥토넷 야! 4.20.목 4.23.일		37951	볼거리가 많아요	[볼거리가 많아요]	37.51182	127.0591	
4	전쟁기념관박물관	서울 용산	02-709-3114	https://map	09:30 ~ 18:00	관람료 0		4.42	851	아이가 배! 12.11.22.화 22		781	유익해요	주차가 [유익해 high]	37.53542	126.9787	
5	롯데월드 (아쿠아리움)	서울 송파	1661-2000	https://map	10:00 ~ 20:00	어른 및 장		4.37	25921	다양한 물! 5.14.일 5.5.금		28102	볼거리가 많아요	관리가 [볼거리가 많아요]	37.51425	127.1052	
6	서대문형무소박물관	서울 서대	02-360-8590	https://map	09:30 ~ 17:00	일반 3000		4.48	233	유익하고 5.13.토 2.23.토		656	유익해요	전시 구성이 [유익해 high]	37.57454	126.9563	
7	서울역사박물관	서울 종로	02-724-0274	https://map	09:00 ~ 18:00	입장료 0		4.48	80	모형전시 5.10.수 4.30.일		19	유익해요	전시 구성이 [유익해 high]	37.57048	126.9705	
8	아쿠아플라넷아쿠아리움	서울 영등	1833-7001	https://map	10:00 ~ 19:00	아쿠아플라		4.18	8724	아쿠아리움 5.13.토 5.5.금		11415	공원이 재밌어요	주차가 [공원이 재밌어요]	37.51985	126.9403	
9	국립고궁박물관	서울 종로	02-3701-750	https://map	10:00 ~ 18:00	관람료 0		4.44	77	코로나 지! 3.12.일 4.3.월		38	유익해요	전시 구성이 [유익해 high]	37.57658	126.9748	

Column name	Description	Column name	Description
name	장소명	place_type	장소 종류
address	주소	telephone	전화번호
detail_url	네이버 플레이스 url	operating_hours	운영시간
fee	요금	rating	평점 평균
rating_count	평점 수	reviews	리뷰
review_date	리뷰 날짜	review_count	리뷰 수
keyword_reviews	키워드 리뷰	tag	장소 추천에 사용할 키워드
target_age	high: 8 세~13 세 low: 5 세~7 세	latitude	위도

longitude	경도	place_img	이미지 경로
-----------	----	-----------	--------

top100: 인기장소 100 개

- top100.csv 파일에 포함되어 있다.
- 인기있는 상위 100 개 장소의 정보를 담고 있다.
- 데이터 포맷: CSV
- 컬럼: name, place_type, address, rating, rating_count, review_count

top100 은 p_data 에서 별점, 방문자수, 리뷰 수를 기준으로 인기 장소 100 개를 나열한 데이터셋으로 컬럼명은 p_data 와 동일하다. top100 뷰 페이지에서 인기 장소를 나열해서 보여줄 때와 챗봇을 통해서 사용자가 언급한 장소 종류에 맞게 장소를 추천할 때 사용하는 데이터셋이다. 아래는 top100 의 샘플 데이터 10 개를 캡처한 그림과 칼럼 이름에 대한 내용을 담은 표이다.

	name	place_type	address	telephone	detail_url	operating_hours	fee	rating	rating_count	reviews	review_date	review_count	keyword_reviews	tag	target_age	latitude	longitude
0	롯데월드 (테마파크)	서울 송파	1661-2000	https://map.na	10:00 ~ 21:00	성인 ₩	4.47	35269	4.27.목 3.31.금	37693	놀이기가 다양한 [놀이]	0	37.5111	127.0982			
1	코엑스아쿠아리움	서울 강남	0507-1435	https://map.na	10:00 ~ 20:00	성인 ₩	4.5	34266	목토넷 아 [4.20.목 4.23.일	37951	볼거리가 많아요관 [볼거]	0	37.51182	127.0591			
2	롯데월드 (아쿠아리움)	서울 송파	1661-2000	https://map.na	10:00 ~ 20:00	어른 ₩	4.37	25921	다양한 물: 5.14.일 5.5.금 4	28102	볼거리가 많아요관 [볼거]	0	37.51425	127.1052			
3	서울스카이 전망대	서울 송파	02-1661-20	https://map.na	10:30 ~ 22:00	어른 (단	4.38	8208	열배도 신: 5.3.수 3.29.수 4	13636	뷰가 좋아요 야경이 [뷰가]	0	37.51425	127.1052			
4	아쿠아플라넷 아쿠아리움	서울 영등	1833-7001	https://map.na	10:00 ~ 19:00	아쿠아	4.18	8724	아쿠아리움 5.13.토 5.5.금 4	11415	공원이 재밌어요 주 [공원]	0	37.51985	126.9403			
5	송파파크8 워터파크	서울 송파	02-1600-06	https://map.na	10:00 ~ 18:00	주말 우	4.16	6684	사우나와 [4.30.일 5.13.토	8310	시설이 깔끔해요 주 [시설]	0	37.48212	127.1239			
6	타요키즈 키즈카페, 실나	서울 동작	0507-1397	https://map.na	11:00 ~ 20:00	어린이	4.38	5756	우리아들 [4.15.토 4.23.일	6372	놀거리가 많아요 공 [놀거]	low	37.49161	126.9247			
7	다이나믹테마파크	서울 종로	02-2034-06	https://map.na	10:00 ~ 18:00	다이나	4.19	5666	비오는 여 [5.6.토 5.1.월 5.:	6517	아이와 가기 좋아도 [아이]	0	37.57247	126.9873			
8	서울어린이테마파크	서울 광진	02-450-931	https://map.na	10:00 ~ 18:30	공원 입	4.57	3950	어린이 나 [5.11.목 4.30.일	5205	아이와 가기 좋아도 [아이]	0	37.53403	127.0678			
9	SeaLaLa 워터파크	서울 영등	1522-9661	https://map.na	07:00 ~ 21:00	워터 파	4.22	4058	도심 한복 [5.7.일 4.16.일 2	4898	시설이 깔끔해요 주 [시설]	0	37.5183	126.9035			

Column name	Description	Column name	Description
name	장소명	place_type	장소 종류
address	주소	telephone	전화번호
detail_url	네이버 플레이스 경로	operating_hours	운영시간
fee	요금	rating	평점 평균
rating_count	평점 수	reviews	리뷰
review_date	리뷰 날짜	review_count	리뷰 수
keyword_reviews	키워드 리뷰	tag	장소 추천에 사용할 키워드
target_age	high: 8 세 ~ 13 세 low: 5 세 ~ 7 세	latitude	위도
longitude	경도	place_img	이미지 경로

parking_data

parking_data 는 서울시 주차장 데이터로 구성된 데이터셋으로 총 1141 개의 주차장에 대한 데이터를 포함한다. 서울 열린데이터광장의 '서울시 공영주차장 안내 정보' 데이터셋을 전처리하여 사용하였다. 장소 근처의 주차장을 추천할 때 사용하는 데이터셋이다. 아래는 parking_data 의 샘플 데이터 10 개를 캡처한 그림이다. 칼럼 이름이 대부분 한글로 되어있으며 영어로 된 칼럼도 직관적이어서 그 내용을 파악하기 쉽기 때문에 칼럼 내용을 작성은 생략하였다.

cultural_data 는 서울시 문화행사 데이터로 구성된 데이터셋으로 총 3453 개의 데이터를 포함한다. 서울 열린데이터광장의 '서울시 문화행사 정보' 데이터셋을 전처리하여 사용하였으며, 캘린더 뷰에서 날짜 별 문화행사 데이터를 보여줄 때 사용하는 데이터셋이다. 아래는 cultural_data 의 샘플 데이터 10 개를 캡처한 그림과 칼럼 이름에 대한 내용을 담은 표이다.

	CLASSNM	AREANM	SVCNM	PLACENM	USETGTINFO	FEE	PERFORMERNM	DTLCONT	SVCURL	IMGURL	RCPTBGNDT	SVCOPNBGNDT	SVCOPNENDDT
0	클래식	마포구	M 아티스마포아트센터 8세이상관람가		VIP석 55, 피아니스트 김도현				https://www.https://culti		2023-03-15	2023-12-05 0:00	2023-12-05 0:00
1	영화	서초구	[서초문화] 서초문화재단전체관람가		전석무료(사전예약)		빌리, 왜 발		https://www.https://culti		2023-05-08	2023-10-10 0:00	2023-10-10 0:00
2	콘서트	강동구	Dear Nex 강동아트센터 14세이상 관람가		R석 66,000원 S석 44,000원				http://www.https://culti		2023-03-02	2023-09-28 0:00	2023-09-28 0:00
3	콘서트	광진구	음유시인 나무아트센터 주우 공지		R석 60,000원 / S석 40,000원				https://www.https://culti		2023-05-18	2023-09-09 0:00	2023-09-09 0:00
4	영화	서초구	[서초문화] 서초문화재단전체관람가		전석무료(사전예약)		알프스에서		https://www.https://culti		2023-05-08	2023-09-05 0:00	2023-09-05 0:00
5	콘서트	마포구	Summer J 마포아트센터 8세 이상		VIP석 77, 카즈미 타테이시		한국과 일본		https://www.https://culti		2023-05-31	2023-08-26 0:00	2023-08-26 0:00
6	축제-문화/예술	광진구	축제 [피크어린]이대공공시민 누구나						https://www.https://culti		2023-05-18	2023-08-26 0:00	2023-08-26 0:00
7	클래식	마포구	2023 M 소마포아트센터 8세이상관람가		VIP석 55,000원, R석 44,000원, S석 33,0				https://www.https://culti		2023-05-09	2023-08-23 0:00	2023-08-23 0:00
8	국악	중구	가족인형극 서울남산국악전체관람가		워크숍 10,000원 / 공연 20,000원				https://www.https://culti		2023-05-18	2023-08-03 0:00	2023-08-05 0:00
9	무용	마포구	2023 해설 마포아트센터 6세(2018년 출생)		전석 1만원				https://www.https://culti		2023-05-23	2023-08-02 0:00	2023-08-02 0:00

Column name	Description	Column name	Description
CLASSNM	서비스 분류	AREANM	지역명
SVCNM	서비스 이름	PLACENM	장소명
USETGTINFO	서비스 대상	FEE	요금
PERFROMERNM	출연자 이름	DTLCONT	상세내용
SVCURL	바로가기 URL	IMGURL	이미지 경로
RCPTBGNDT	접수시작일시	SVCOPNBGNDT	서비스개시시작일시
SVCOPNENDDT	서비스개시종료일시		

test_data: 모델 학습용 데이터

- test_data.csv 파일에 포함되어 있다.
- 모델 학습을 위한 데이터로 활용된다.
- 데이터 포맷: CSV
- 컬럼: name, place_type, place_img, address, telephone, detail_url, operating_hours, fee, rating, rating_count, reviews, review_date, review_count, keyword_reviews

3.5. Algorithm

3.5.1. Keyword Extraction

3.5.2. Keywords Based Recommendation Algorithm(예진)

아래는 'recommend_places' 함수의 코드이다.

```
def recommend_places(input_string, target_age=None):

    similar_words = model.wv.most_similar(input_string, topn=3)
```

```

similar_words = [word[0] for word in similar_words]

if target_age:

    filtered_df = place_df[(place_df['tag'].apply(lambda x: any(word in x for word in similar_words))) & ((place_df['target_age'] == target_age) | (place_df['target_age'].isnull()))]

else:

    filtered_df = place_df[place_df['tag'].apply(lambda x: any(word in x for word in similar_words))]

filtered_df = filtered_df.sort_values(['rating', 'rating_count', 'review_count'], ascending=[False, False, False]).head(5)

return filtered_df

```

이 함수는 사용자가 입력한 문자열(input_string)과 대상 연령(target_age)을 기반으로 장소를 추천하는 알고리즘을 구현한 것이다.

위의 코드에서 알고리즘은 다음과 같이 동작한다.

- 입력 문자열(input_string)과 유사한 단어를 찾기 위해 word2vec 모델을 사용한다.
model.wv.most_similar 메서드를 통해 입력 문자열과 유사한 상위 3 개의 단어를 선택한다.
- 대상 연령(target_age)이 주어진 경우, 선택된 유사 단어들이 장소 데이터의 'tag' 열에 포함되고, 대상 연령과 일치하거나 대상 연령이 누락된 경우의 데이터를 필터링한다.
- 대상 연령이 주어지지 않은 경우, 선택된 유사 단어들이 장소 데이터의 'tag' 열에 포함되는 데이터를 필터링한다.
- 필터링된 데이터는 'rating', 'rating_count', 'review_count' 열을 기준으로 내림차순 정렬되고, 상위 5 개의 장소를 선택한다.
- 최종 결과로 선택된 장소 데이터를 반환한다.

3.5.3. Distance Based Recommendation Algorithm

장소명을 입력으로 받고, 해당 장소의 위도와 경도를 Haversine 공식에 대입하여 직선거리를 도출해 가까운 거리에 위치한 순서대로 3 개의 장소명을 반환한다. 입력하는 장소는 사용자가 입력한 장소이고, 반환하는 장소는 주차장 혹은 카페에 해당한다. 이 알고리즘의 코드는 다음과 같다.


```

# import package
import pandas as pd
import numpy as np
from geopy.geocoders import Nominatim
import pandas
from haversine import haversine

# data import(local based)
place_df = pd.read_csv('p_data.csv', index_col=0)
parking_df = pd.read_csv('parking_data.csv', index_col=0)

# function
def distance_calculation(place_name):

    distances = []
    x = place_df[place_df['name'] == place_name]['latitude'].values[0]
    y = place_df[place_df['name'] == place_name]['longitude'].values[0]

    for name, lat, long in zip(parking_df['name'], parking_df['latitude'], parking_df
['longitude']):
        place = (x, y)
        parking = (lat, long)
        distance = haversine(place, parking, unit='km')
        distances.append((distance, name))

    distances.sort() # Sort distances in ascending order

```

3.6. Dialogflow

3.6.1. Entity

설정된 엔티티는 다음 표와 같다. Dialogflow 학습 시에 꼭 필요한 단어나 사용자의 대화 중 필히 인식해야 하는 단어를 엔티티로 설정하였고, 사용자의 언어를 유연하게 받아들일 수 있도록 유의어와 동의어를 수기로 입력하였다.

Entity name	Description
additional_information	추가적인 제공할 정보와 관련한 개체들이 담겨있다. 예를 들면, '주차 정보'와 '카페' 등의 단어들이 포함된다.
children_age	맞춤형 장소 추천에서 사용되는 아이의 나이와 관련한 개체들이 담겨있다. 예를 들면, '초등학생', '어린이' 등의 단어들이 포함된다.
keywords	맞춤형 장소 추천에서 사용되는 키워드 개체들이 담겨있다. 예를 들면, '컨셉이 독특한', '운동을 좋아하는' 등의 단어들이 포함된다.
location	지역에 관한 개체들이 담겨있다. 예를 들면, '서울', '강동구' 등의 단어들이 포함된다.
person_type	아이와 어른에 관한 개체들이 담겨있다.
place_name	장소명 개체들이 담겨있다. 예를 들면, '롯데월드', '서울어린이대공원', 등의 단어들이 포함된다.
place_type	장소 종류 개체들이 담겨있다. 예를 들면, '캠핑장', '박물관' 등의 단어들이 포함된다.

2.6.2. Intent

Intent Design

처음 설계 시 인텐트 설정은 아래 표와 같다. 키워드 맞춤형 장소 추천 기능, 장소에 관한 부가 정보 제공 기능, 그 외의 부가 정보 제공 기능으로 세 가지의 인텐트를 구성하였지만, 설계한 인텐트로 구현하였을 때, 인텐트가 구체화되어 있지 않아 각각의 기능을 수행하지 못하였다.

Intent name	Description
Default Welcome Intent	사용자가 챗봇 대화에 처음 입장했을 때 기본적인 인사말을 제공한다.
Default Fallback Intent	챗봇 대화 도중 사용자의 말을 이해하지 못했을 때의 답변을 제공한다.
place_recommend	사용자가 선택한 키워드에 따라 추천된 장소 정보를 제공한다.


place_recommend-custom	place_recommend의 follow-up intent 로 place_recommend에서 추천한 장소에 관한 주차 정보, 근처 식당 등의 추가적인 정보를 제공한다.
Others	장소 추천 외 날씨 등의 정보를 물어 보았을 때에 대한 답변을 제공한다.

Finalized Intent

최종 확정된 인텐트는 아래 표와 같다. 챗봇 대화의 처음 입장 시 제공하는 인사말과 사용자의 말을 이해하지 못하였을 때 제공하는 답변 형식은 설계와 같으며, 그 외의 인텐드가 구체화되어 기능에 따라 4 가지의 인텐트로 구성되고 필요에 의해 fall-back intent 를 설정하였다.

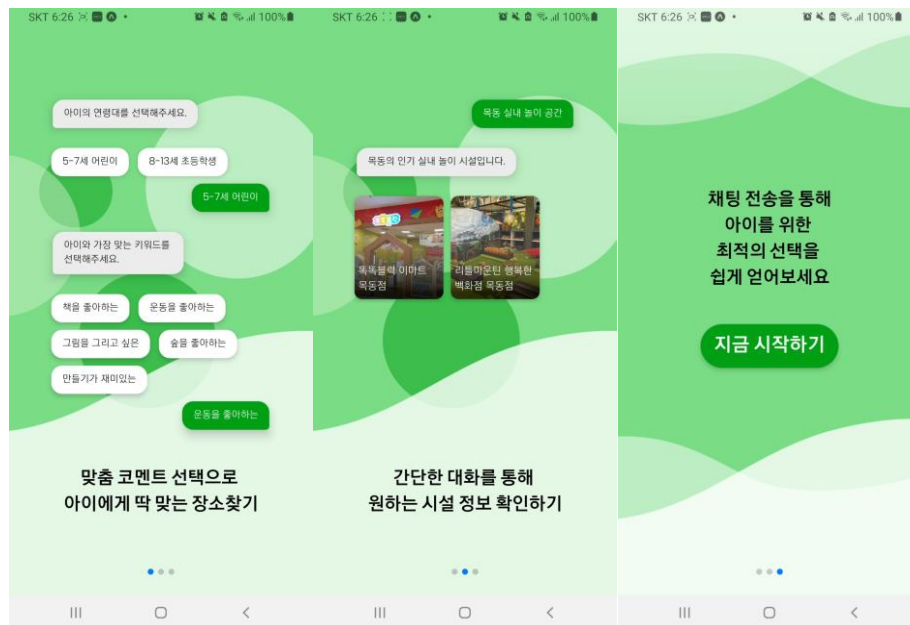
Intent name	Description
Default Welcome Intent	사용자가 챗봇 대화에 처음 입장했을 때 기본적인 인사말을 제공한다.
Default Fallback Intent	챗봇 대화 도중 사용자의 말을 이해하지 못했을 때의 답변을 제공한다.
place_recommend_by_keywords place_recommend_by_keywords - age place_recommend_by_keywords - keywords	사용자가 선택한 키워드에 따라 추천된 장소를 제공하는 intent로 사용자가 장소를 추천 받고 싶을 때 호출된다. place_recommend_by_keywords - age와 place_recommend_by_keywords - keywords는 place_recommend_by_keywords의 follow up intent로 각각 아이의 나이와 성향 키워드를 입력받는다.
place_recommend_by_type	사용자가 원하는 장소 유형에 해당하는 인기 장소 목록을 제공하는 intent로 사용자가 장소 유형을 언급했을 때 호출된다.
place_additional_information	날씨나 주차정보 등 장소에 대한 추가적인 정보를 제공하는 intent로 추천받은 장소가 있을 시에 호출된다.
additional_information additional_information - place	날씨나 주차정보 등 장소에 대한 추가적인 정보를 제공하는 intent로 사용자가 장소 없이 추가적인 정보만을 물어봤을 때 호출된다. additional_information - place은 additional_information의 follow up intent로 해당 intent에서 사용자에게 가고자 하는 장소에 대한 정보를 입력받은 후에 장소에 맞는 정보를 제공한다.

3.6.3. Dialogflow Results

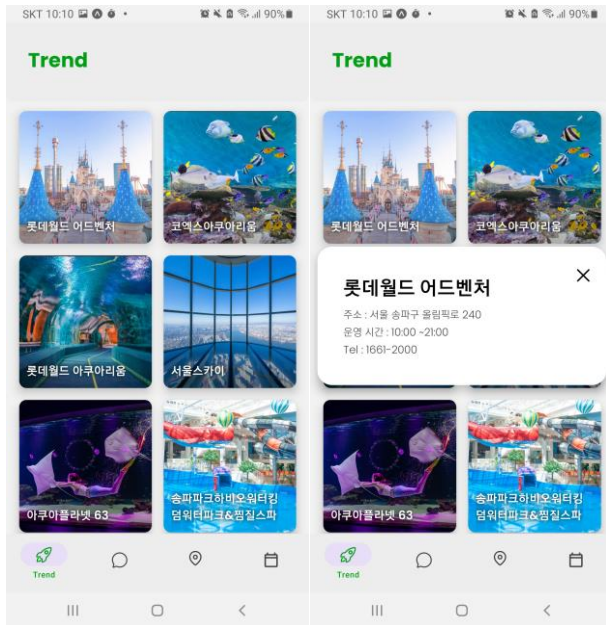
 <div>forkids</div> <div>여 의해 작동됩니다 </div> <div>안녕</div> <div>안녕하세요. 아이들을 위한 맞춤형 장소 추천 챗봇, Forkids 입니다. Forkids를 통해 다양한 놀이시설 정보를 확인해보세요.</div> <div>장소 추천해줘</div> <div>자녀의 연령대를 선택해주세요. [5-7세 어린이][8-13세 초등학생]</div> <div>뭔가를 요청하세요 </div>	 <div>forkids</div> <div>여 의해 작동됩니다 </div> <div>초등학생</div> <div>아이가 초등학생이사군요! 다음으로는 아이 혹은 장소와 관련된 키워드를 선택해주세요. (예: 운동, 그림, 동물/박물관, 미술관 등등)</div> <div>미술을 좋아하는</div> <div>'미술을 좋아하는' 아이의 연령과 성향에 어울리는 곳을 추천해드릴게요!</div> <div>뭔가를 요청하세요 </div>	 <div>forkids</div> <div>여 의해 작동됩니다 </div> <div>캠핑장 추천해줘</div> <div>캠핑장에 관한 장소 추천드리겠습니다.</div> <div>뭔가를 요청하세요 </div>
맞춤형 장소 추천 결과		장소 종류별 추천 결과

엔티티와 인텐트를 학습시켜 나온 결과는 아래와 같다. 서버와 연결 전 Dialogflow 내에서 학습시킨 결과이기 때문에 데이터는 나타내고 있지 않지만, 목표하는 플로우대로 잘 흘러가는 것을 확인할 수 있다.

4 Result

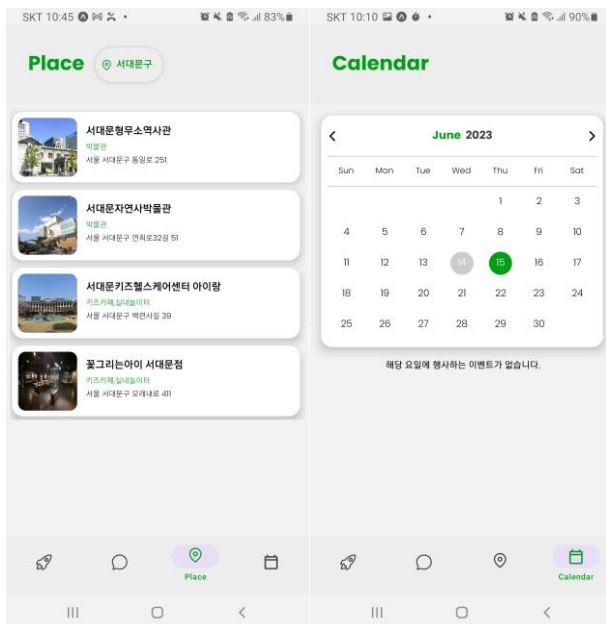


앱 처음 시작시 나오는 튜토리얼 화면 '지금 시작하기' 버튼을 눌러 Chat화면으로 넘어간다.



<Trend>

트렌드 화면에서는 데이터베이스에서 인기 있는 장소들을 보여준다. 장소 블록을 클릭하면 장소 이름과 주소, 운영시간과 전화번호를 얻을 수 있다. 추가적으로 장소의 분류라벨과 주차장의 정보가 포함되어 있으면 그 정보 또한 모달 창에서 확인할 수 있다.

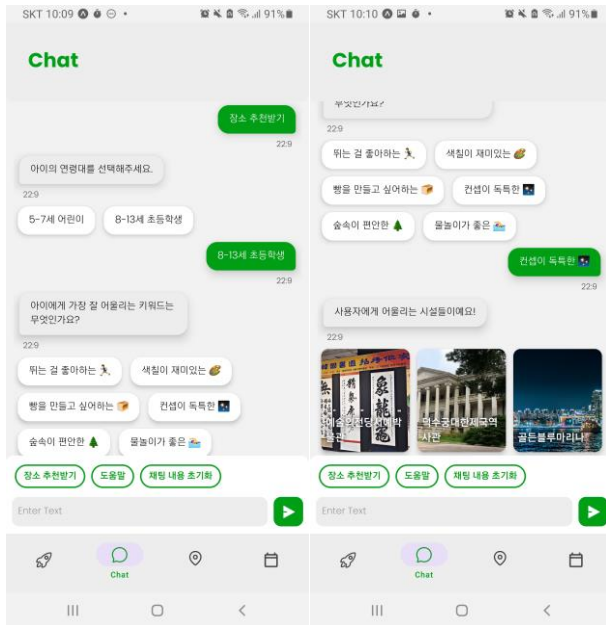


<Place>

상단의 장소를 구 단위로 선택하면 해당 구 내의 장소 리스트를 얻을 수 있다. 블록 클릭 시 placeblock과 동일하게 모달 창을 통해 상세 정보를 얻을 수 있다.

<Calendar>

Calendar 달력에서 날짜를 클릭하여 선택할 수 있으며 선택한 날짜에 운영하는 문화행사 프로그램 리스트를 확인할 수 있으며 해당 프로그램 블록을 클릭하면 관련 url 링크로 이동한다.



<Chat>

Chat 화면에서 미리 제안되는 예시 텍스트를 클릭하거나 직접 텍스트를 입력하여 챗봇에 시설정보를 요청하면, 해당 장소의 정보를 PlaceBubble 또는 텍스트를 통해 제공한다. PlaceBubble은 클릭 시 시설에 대한 자세한 정보를 얻을 수 있다.