ARA Concept & Design

ARA Concepts

SAFE

1. 동행 Leader/Follower 로서의 **Data를 공개**
2. 서로의 동행 이력을 보면서 **신뢰감을 형성**할 수 있음
3. **서로 평가**가 가능. 하지만 공격적인 평가가 되지 않게 **제한적인 평가**
   -> 성향을 알려주는 정도로 작용

Simplicity

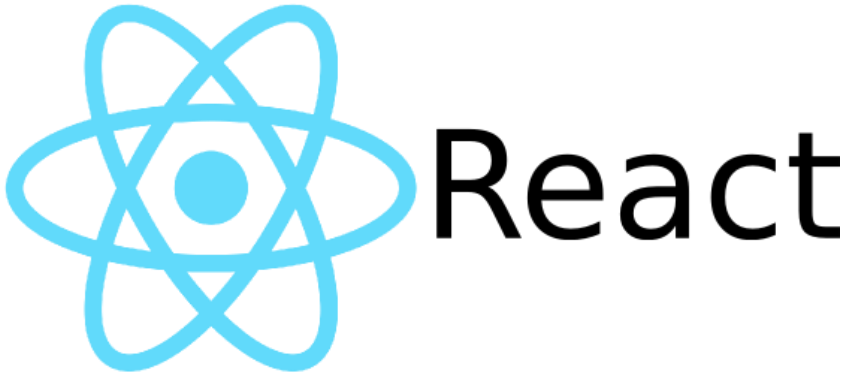
1. **쉬운 UI 및 UX**
2. 한눈에 들어오는 Interface & Information
3. 단순하고 **직관적인 Page Logic**

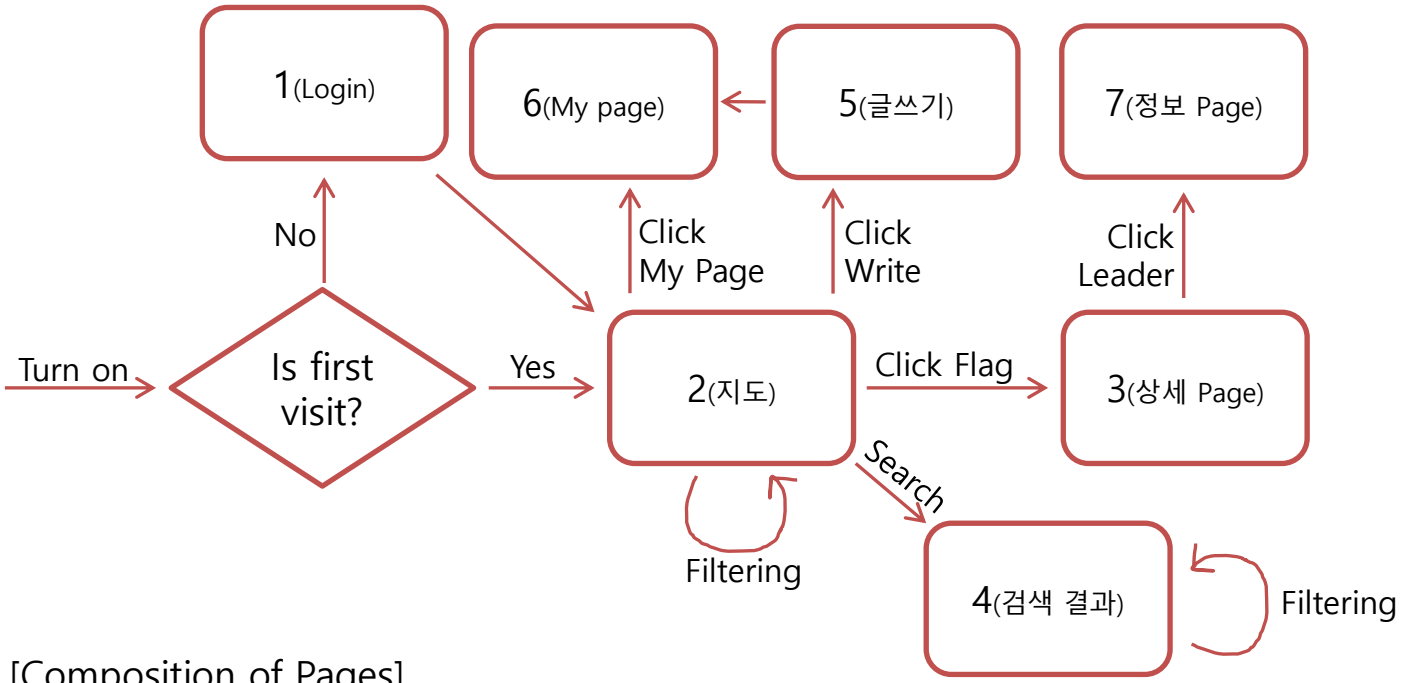
Suitability

[Next Generation Item]

1. **Big data기반**으로 사용자가 동행을 선택하고 구성하는데 도움을 준다.
    - **동행 장소** 추천
    - 함께 하면 만족도 높은 **동행자 성향** 추천(자신의 성향 대비)
    - 여행지 별 만족도 높은 **동행 Spec** 추천
2. 위의 내용을 위해 데이터베이스 설계 시 데이터 세분화 필요

Front-end


React



[Composition of Pages]

1. **로그인 및 회원가입**(Naver, Kakao, Facebook등 연동 가능)
2. 내 위치 정보 기반으로 주변 **동행 모집 정보 확인**
   (에어비앤비, 직방 등과 같이 지도 위에 간단한 정보를 띄워주는 방식)
3. **동행 모집 상세 Page**: 지도 or 검색을 통해 선택하여 확인 가능한 Page
4. 동행 모집 키워드 **검색 및 필터링**
5. 동행 모집 **글쓰기**
6. **My Page**(내 성향, 정보, 내가 쓴 글, 동행이력 등)
7. 다른 사람 성향 및 **정보 Page**

Back-end

Language & Framework

VS

[Pros]
성능이 좋음, API 만드는데 좋음
동시 요청 관리 용이

[Cons]
비동기식 프로그래밍의 어려움
Single Thread, **Callback 지옥**

[Pros]
Security, **DB 연동 쉬움**
개발 속도 빠름, 자료 많음

[Cons]
크기가 큼, 작고 간단한 앱에는 부적합
Framework에 대한 완전한 이해 필요

[Criteria for choosing Laguage & Framework]

- **CRUD(Create, Read, Update, Delete)**위주의 Service: python + Django
- **Realtime Service**, API, Customize가 중요: Node.js + Express.js

[Conclusion]

- ARA는 **Realtime Service가 필요하지 않음**
- **개발이 쉽고 확장성이 높은** Python 기반의 Django 프레임워크를 사용하는 것을 추천

Database

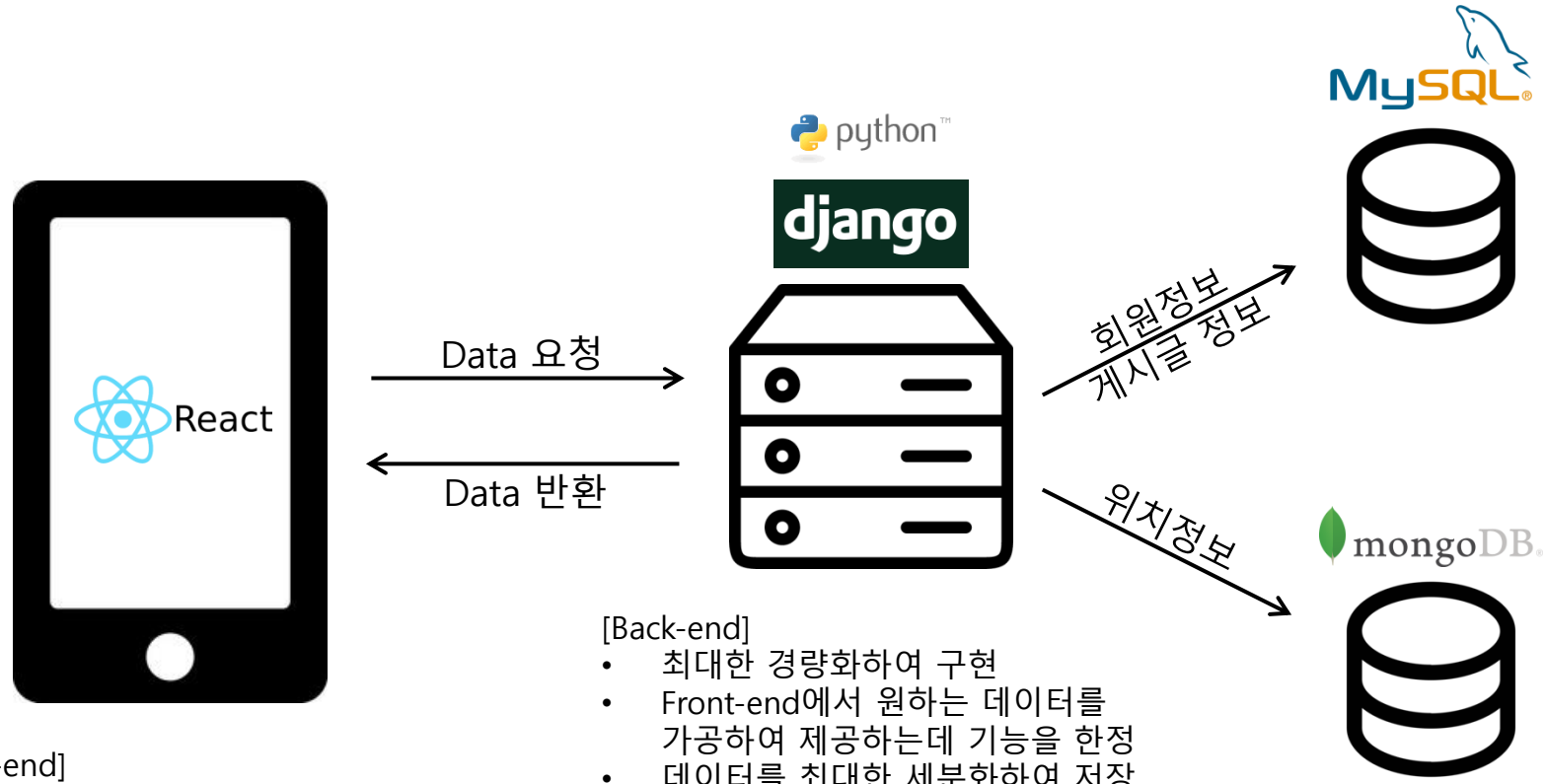
- **기본적인 DB설계 및 회원 관리 용이**
- SQL만 효율적으로 구현한다면 웬만큼 큰 서버 커버 가능
- 자료 많음

- **2차원 Index(2d)**가 탑재되어 위치 정보 처리에 유리함
- **KNN 쿼리**를 통해 위치 주변에서 **거리 순서로 정렬한 N개의 결과**를 효율적으로 도출

[Conclusion]

- **ARA는 위치정보 기반**으로 데이터를 제공하므로 MongoDB를 사용하면 좋음
- 기본적인 데이터는 MySQL로 관리하되 MongoDB를 이용한 **위치정보 Searching의 성능을 높이는 방법** 추천

Whole Architecture



[Front-end]
- 화면을 보여주는 모든 작업 처리
- 최대한 단순하고 직관적인 디자인
- 너무 많은 페이지가 있어서는 안됨

[Back-end]
- 최대한 경량화하여 구현
- Front-end에서 원하는 데이터를 가공하여 제공하는데 기능을 한정
- 데이터를 최대한 세분화하여 저장할 수 있게 설계

The end