캡스톤디자인

〈 Aniroomie - Final Report 〉

Team #2

20191127 김두홍

20214372 배수민

20216766 손유진

목차

---

1. 서비스 개요

'Aniroomie' 서비스는 대학교 기숙사에 거주하거나 거주하기를 희망하는 많은 학생들의 편의를 위한 애플리케이션이다.

'Aniroomie'라는 이름은 두 가지를 의미한다. 하나는 'Animal'과 'Roomie'의 합성어로, 여러 가지 동물 유형을 가진 사용자들의 룸메이트 매칭을 의미하며, 또 다른 하나는 'Any'와 'Roomie'의 합성어로, 어떤 사람이든 룸메이트 매칭 서비스의 사용자가 될 수 있음을 의미한다.

본 서비스는 룸메이트 매칭 서비스를 주력으로 하며, 성향 조사를 통한 동물 유형 알아보기, 비슷한 성향의 사용자들과 함께 참여할 수 있는 모임 추천 등을 추가 기능으로 한다.

2. 아이디어 제안 배경

물가 상승으로 인해 학교 주변에 적당한 가격의 자취방을 구하는 것이 어려워졌으며, 많은 사건/사고로 인해 안전 문제에 대한 관심이 증가하면서 기숙사를 선호하는 학생들이 증가하고 있다. 일반적인 원룸 자취와 비교하면 월세가 저렴하고, 강의실과 매우 가까우며, 캠퍼스 내에 위치하기 때문에 비교적 안전한 환경을 제공한다는 장점이 있다. 하지만, 기숙사는 공동생활 공간이기 때문에 타인과의 접촉이 빈번하게 발생하며, 이와 관련된 다양한 문제들이 존재한다. 그 중, 룸메이트와의 성격 또는 생활패턴의 차이로 인한 스트레스가 가장 큰 문제로 지적받고 있다.

룸메이트를 사전에 지정할 수 있는 시스템이 존재하지만, 원래 친구가 아닌 사람과 룸메이트 신청을 하기 위해서는 타 커뮤니티 등을 통해서 룸메이트가 될 사람을 찾아야 하는데, 이는 굉장히 많은 시간과 노력을 필요로 한다. 그래서 좀 더 편리하고 빠르게 룸메이트를 찾을 수 있도록 도와주는 추천 서비스를 고려하게 되었다.

3. 문제 상황 및 원인 분석

1) 기숙사 홈페이지의 룸메이트 매칭만으로는 사용자와 잘 맞는 룸메이트를 찾는 것이 어렵다.

→ 홈페이지에서는 흡연/비흡연, 아침형/저녁형, 외국인 선호/비선호 등의 간단한 몇 개의 정보만을 사용한다.

⇒ 더 많은 성향 정보를 기반으로 분석하여 추천하면 더 잘 맞는 룸메이트를 찾을 수 있을 것이다.

2) 에브리타임 등의 커뮤니티에서 룸메이트를 찾는 과정이 복잡하다.

→ 에브리타임 기숙사 게시판에 직접 글을 쓰거나 다른 사람의 글에 댓글/쪽지를 이용하여 룸메이트를 구할 수 있지만, 과정이 매우 번거롭고 서로 모든 성향 정보를 공유할 수 없기 룸메이트 만족도가 낮을 수 있다.

⇒ 추천 서비스를 이용하여 본인과 유사하거나 본인이 원하는 룸메이트 상에 부합하는 사용자를 추천받음으로써 룸메이트 만족도를 높일 수 있을 것이다.

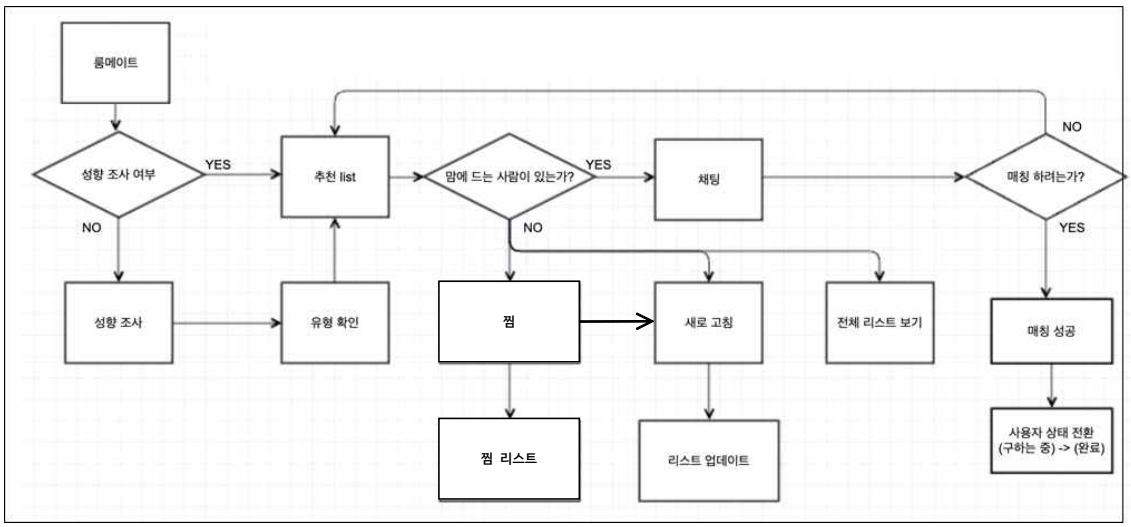
⇒ 본인의 성향 정보와 간단한 민감도 정보를 입력하면, 추천 알고리즘이 알아서 잘 맞을 만한 룸메이트를 추천해주기 때문에 매우 간단하게 룸메이트를 찾을 수 있을 것이다.

4. 다른 서비스와의 차이점

기존의 기숙사 홈페이지나 애플리케이션의 경우, 간단한 룸메이트의 선호 특징을 설정하여 호실을 배정받는 시스템이 존재한다. 하지만, 흡연/비흡연, 아침형/저녁형, 외국인 선호/비선호 등의 간단한 룸메이트의 특징만을 선택할 수 있으며, 이 정보만으로 사용자와 잘 맞는 룸메이트를 매칭하는 것은 매우 어렵다. 따라서, 더 많은 정보를 입력받아 사용자와 가장 잘 맞을 만한 룸메이트를 추천해주고 선택하도록 하는 룸메이트 매칭 시스템을 만드는 것이 본 서비스의 주요 목표이다.

룸메이트 추천 서비스를 구상하며 또 한 가지 생각한 것이 유사한 서비스를 제공하는 다른 웹/앱과 차별화된 사용자 인터페이스를 구성하는 것이다. 성격 유형 검사를 통해 본인의 MBTI 유형을 확인할 수 있는 것에서 영감을 얻어 본인의 성향 정보를 입력하면 성향에 따른 본인의 동물 유형을 확인할 수 있는 알고리즘을 추가하여 사용자들이 흥미를 느끼고 성향 조사에 더 성실하게 참여할 수 있도록 하고자 했다.

5. 메인 서비스 흐름도 (룸메이트 추천 기능)



6. 기능적 요구사항 분석

① 회원가입

- 사용자는 이름, 닉네임, 학교, 아이디, 비밀번호를 입력해 회원 가입을 해야 한다.
- 사용자는 회원가입 시 학교 이메일을 통해 해당 학교 학생임을 인증해야 한다.
- 사용자는 비밀번호를 한번 더 확인하여 회원 가입해야 한다.

② 로그인/로그아웃

- 사용자는 아이디와 비밀번호를 기입하여 로그인 해야 한다.
- 사용자는 로그아웃 해야 한다.

③ 성향조사

- 사용자는 룸메이트 창에 첫 접속 시 성향 조사 질문에 응답해야 한다.
- 사용자는 자신이 룸메이트를 구하고 있는 상태인지 설정해야 한다.
- 시스템은 사용자가 룸메이트를 구하고 있는 상태라면, 다른 사람의 추천 목록 페이지에 뜨게 해야 한다.
- 시스템은 사용자가 룸메이트를 구하지 않는 상태로 바뀌면, 다른 사람의 추천 목록 페이지에서 해당 사용자를 삭제해야 한다.
- 시스템은 사용자의 성향 조사에 따라 추천 목록 페이지를 제공해야 한다.
- 시스템은 추천 목록 페이지의 구성이 완료되면 사용자에게 알림을 해야 한다.
- 시스템은 사용자가 더 많은 목록 보기를 선택하면 더 많은 사용자들을 추천 목록 페이지에 추가해야 한다.
- 사용자는 추천 목록에서 다른 사용자를 선택해 채팅해야 한다.
- 사용자는 추천 목록에 마음에 드는 사람이 있다면 찜을 눌러 다시 확인할 수 있어야 한다.
- 사용자는 전체 목록에서 다른 사용자를 선택해 채팅해야 한다.

④ 채팅

- 사용자는 다른 사용자에게 메시지를 보내야 한다.
- 사용자는 다른 사용자에게 메시지를 받아야 한다.
- 사용자는 다른 사용자에게 메시지가 왔을 때, 알림을 받아야 한다.
- 사용자는 다른 사용자과 룸메이트 성사 버튼을 눌러야 한다.
- 시스템은 다른 사용자과 룸메이트 성사 버튼이 눌렸을 때 두 사용자 모두 룸메이트를 구하지 않는 상태로 변경해야 한다.
- 사용자는 채팅 방을 나갈 수 있다.

7. 서비스 기술 스택

웹 클라이언트에는 'React', 웹 서버에는 'Spring Boot'를 사용하였고, 룸메이트 추천에 필요한 정보들은 'MYSQL' 데이터베이스에 저장했습니다. 머신러닝을 활용한 추천 알고리즘에 필요한 AI 서버는 'FastAPI'로 개발했으며, 해당 서버는 'AWS Elastic Beanstalk'에 배포하였습니다.

8. 프론트엔드 구현 내용

① 브랜드 컬러 및 브랜드 캐릭터

브랜드 컬러 : #27334B

브랜드 캐릭터 : 루미(펭귄 캐릭터)

② 프론트엔드 세부 기술 스택

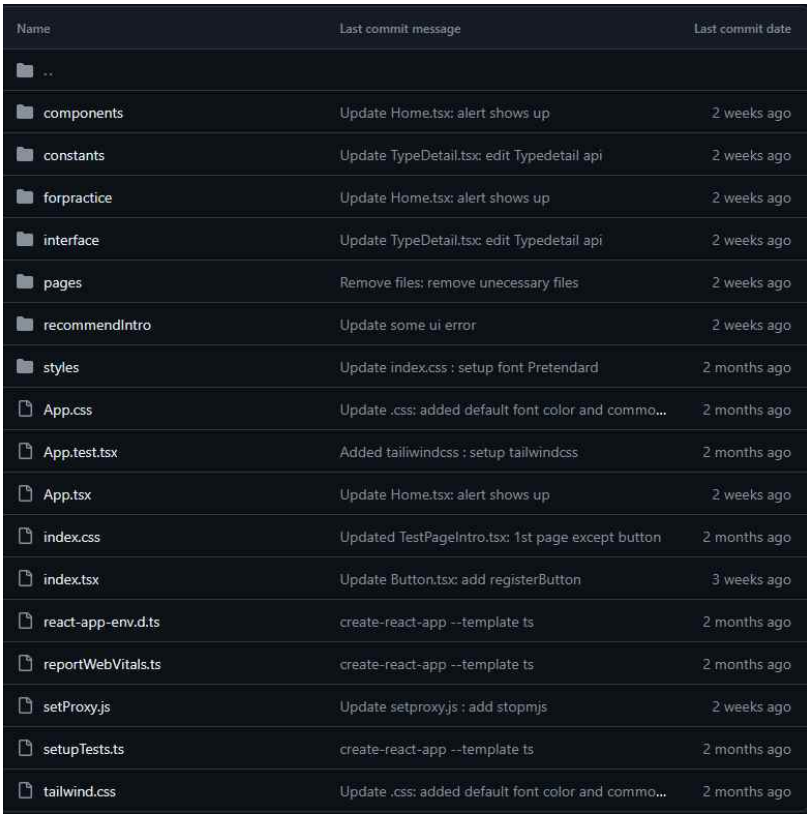
ReactJs, TailwindCSS

③ 세부 페이지 구조

가. 회원가입/로그인 페이지
나. 생활유형 테스트 페이지
다. 생활유형 결과 페이지
라. 동물사전 페이지
마. 홈 화면 페이지
바. 룸메이트 추천 페이지
사. 채팅 페이지
아. 기숙사 모임 페이지

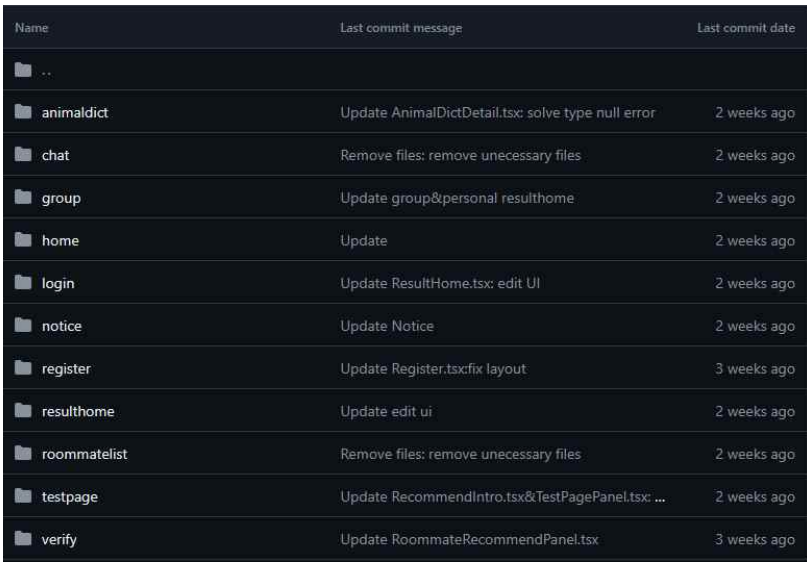
④ 파일 전체 구조

각 파트 별 코드는 pages에 넣어두었고, constants에는 동물의 유형이나 타입, 색을 정해두었고, components에는 재활용 가능한 컴포넌트들을 따로 관리하는 등 코드의 재활용을 위한 구조를 잘 구성하기 위해 노력했다.



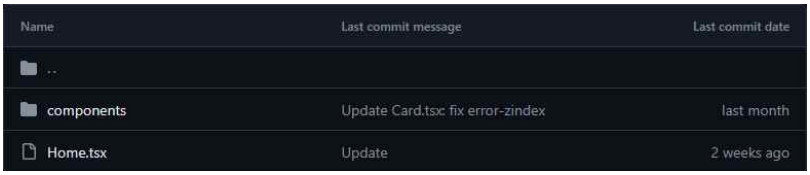
Name	Last commit message	Last commit date
..		
components	Update Home.tsx: alert shows up	2 weeks ago
constants	Update TypeDetail.tsx: edit Typedetail api	2 weeks ago
forpractice	Update Home.tsx: alert shows up	2 weeks ago
interface	Update TypeDetail.tsx: edit Typedetail api	2 weeks ago
pages	Remove files: remove unecessary files	2 weeks ago
recommendIntro	Update some ui error	2 weeks ago
styles	Update index.css : setup font Pretendard	2 months ago
App.css	Update .css: added default font color and commo...	2 months ago
App.test.tsx	Added tailiwindcss : setup tailwindcss	2 months ago
App.tsx	Update Home.tsx: alert shows up	2 weeks ago
index.css	Updated TestPageIntro.tsx: 1st page except button	2 months ago
index.tsx	Update Button.tsx: add registerButton	3 weeks ago
react-app-env.d.ts	create-react-app --template ts	2 months ago
reportWebVitals.ts	create-react-app --template ts	2 months ago
setProxy.js	Update setproxy.js : add stopmjs	2 weeks ago
setupTests.ts	create-react-app --template ts	2 months ago
tailwind.css	Update .css: added default font color and commo...	2 months ago

각 페이지의 코드가 있는 pages 파일 구조이며, 앞서 언급한 8개 파트의 페이지가 담겨있다.



Name	Last commit message	Last commit date
..		
animaldict	Update AnimalDictDetail.tsx: solve type null error	2 weeks ago
chat	Remove files: remove unecessary files	2 weeks ago
group	Update group&personal resulthome	2 weeks ago
home	Update	2 weeks ago
login	Update ResultHome.tsx: edit UI	2 weeks ago
notice	Update Notice	2 weeks ago
register	Update Register.tsx:fix layout	3 weeks ago
resulthome	Update edit ui	2 weeks ago
roommatelist	Remove files: remove unecessary files	2 weeks ago
testpage	Update RecommendIntro.tsx&TestPagePanel.tsx: ...	2 weeks ago
verify	Update RoommateRecommendPanel.tsx	3 weeks ago

파트 별로 페이지를 파고, 그 안에 대표 .tsx 파일을 두고, 여러 번 재활용되는 코드거나 하위 코드인 경우는 components에 두어 관리했다.



9. 백엔드 구현 내용

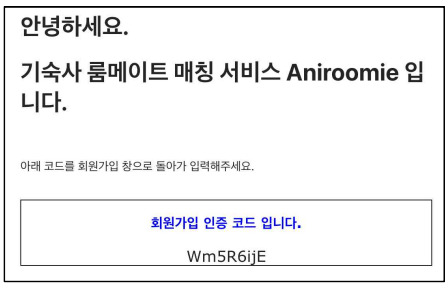
① 백엔드 세부 기술 스택

웹 서버 : Spring Boot
데이터베이스 : MYSQL
배포 : Elastic Beanstalk

② 회원가입 기능 구현

가. cau 이메일 인증 기능

중앙대 재학생만 가입 가능하도록 하기 위해 @cau.ac.kr 이메일로인증코드를 보내는 기능을 구현하였다. Google smtp server를 이용하였다. 회원이 이메일 인증하기를 누르면 해당 중앙대 이메일로 다음과 같은 메일이 전송된다.



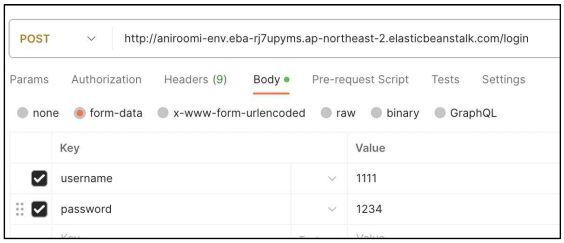
사용자가 해당 코드를 입력하면 클라이언트 단에서 일치하는지 확인한 후, 가입 절차를 마저 진행한다.

나. 회원가입 기능

사용자가 자신의 정보를 모두 기입한 후 가입하기를 누르면, 해당 api로 사용자 정보가 서버에 전송되고, 데이터베이스에 저장된다. 이때 password는 해시 함수를 통해 암호화 후 저장된다.

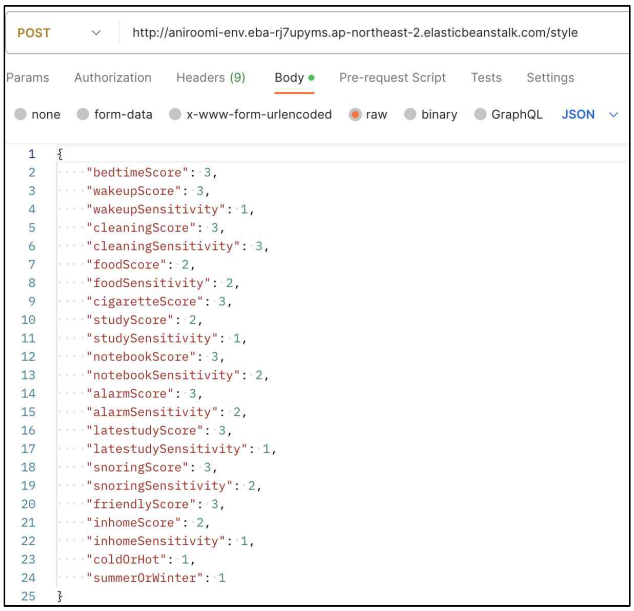
③ 로그인 기능 구현

Spring security를 이용해 Session 형식의 form 로그인 방식으로 구현하였다.



④ 룸메이트 성향 조사 기능 구현

성향 조사를 위한 '/style' api를 구현하여 클라이언트 단에서 성향 조사 결과를 받아오도록 구현하였다.



그리고 서버단에서 받아온 결과에 따른 사용자의 동물을 지정한 후 사용자의 성향 조사와 동물 유형을 각각 데이터베이스에 저장한다.

⑤ 동물 유형 및 유저 특성 반환 기능 구현

클라이언트 단에서 사용자 본인 혹은 다른 사용자의 동물과 성향 정보를 요청하면, 해당 정보들을 계산 및 처리한 후 보내주는 api들을 구현하였다.

해당 api들은 자신의 동물 정보를 볼 때나 추천된 사용자의 정보를 보고 싶을 때 사용된다. 사용자와 다른 사람일 때의 api가 다른 이유는, 접속한 해당 사용자의 id는 이미 알고 있어서 굳이 서버 단으로 넘겨주지 않아도 확인이 돼서 다르게 처리했다.

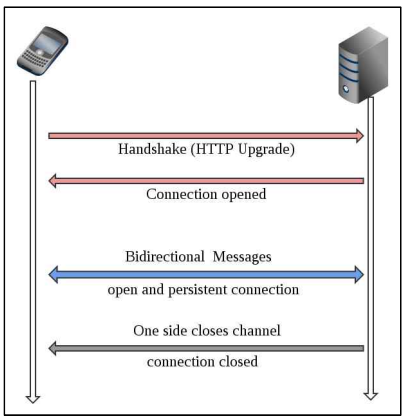
⑥ 룸메이트 추천 기능 구현

룸메이트 추천은 AI 서버(Fast api)와의 상호작용을 통해 이루어진다.
Fast api를 사용하여 다음 6가지 기능을 구현하였다.

- 사용자가 새로 룸메이트를 찾는 상태가 되었을 때 해당 사용자의 성향 조사 결과와 id를 전송하는 기능
- 사용자가 더이상 룸메이트를 찾는 상태가 아닐 때 AI 서버에 알려주는 기능
- 사용자 두 명의 매칭이 성사되었을 때 알려주는 기능
- 클라이언트 단에서 추천 리스트를 요청했을 때 사용자의 추천 리스트를 받아와 card 형식으로 가공하여 (추천 사용자들의 학과, 나이, 동물 정보 등을 담아) 반환하는 기능
- 클라이언트 단에서 추천 리스트의 새로 고침을 요청했을 때 새로 고침된 추천 리스트를 받아와 card 형식으로 반환하는 기능
- 사용자가 '찜하기'를 눌렀을 때 AI 서버에 알려주는 기능

⑦ 채팅 기능 구현

채팅은 websocket을 사용해 실시간 양방향 통신을 구현하였다.



← [websocket architecture]

사용자가 추천 리스트에서 원하는 사람에게 채팅하기를 누르면 서버에서 해당 채팅방이 이미 존재하는지 확인한 후, 없으면 새로운 chatroom을 개설하고 있다면 해당 chat 방으로 이동한다. 그 후 데이터베이스에 저장되어있는 채팅 내용을 불러온다. 사용자가 채팅을 하면 해당 채팅은 바로 다른 상대방에게 전달되고 동시에 데이터베이스에 timestamp와 함께 저장된다. 이후 채팅방에서 '매칭하기'를 누르면 양쪽 사용자가 모두 눌렀는지 확인한 후, 모두 눌렀다면 해당 사용자들의 룸메이트 구하는 status를 false로 변경하고 ai 서버에 매칭 id쌍을 전송한다.

10. AI 구현 내용

① AI 세부 기술 스택

Fast API (Python)

② 클러스터링 구현

다양한 성향을 가진 사용자들이 존재하기 때문에 1차적으로 성향이 비슷한 사용자끼리 매칭되는 것이 가장 중요하다고 생각했고, 사용자의 여러 성향 수치를 다차원 공간상의 데이터 포인트로 하여 클러스터링했다. 이후 유사도 기반 추천 시에 해당 사용자와 같은 클러스터의 사용자를 우선적으로 추천하게 된다.

```
def clustering(u_data, f_name):

    result = []

    with open(f_name, 'r', encoding='utf-8') as f:
        dataset = json.load(f)

    clustering_dataset = fsc.clustering_modify(dataset)

    df = pd.DataFrame(clustering_dataset)
    IDs = df['userId'].tolist()
    df = df.drop('userId', axis=1)

    kmeans = KMeans(n_clusters=3, random_state=10, n_init=10)
    kmeans.fit(df)

    for i, label in enumerate(kmeans.labels_):
        temp = {"userId": IDs[i], "cluster": str(label)}
        result.append(temp)
```

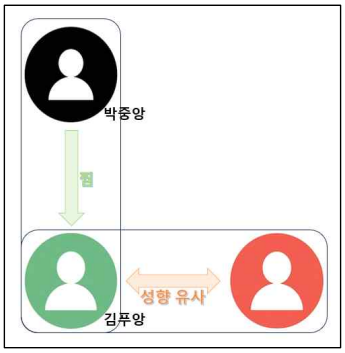
③ 유사도 측정 및 순위 알고리즘 구현

성향 조사 시에 두 개의 질문을 제시한다. 한 개는 사용자의 성향 수치를 파악하기 위한 질문이고, 다른 한 개는 사용자의 성향에 대한 민감도를 파악하기 위한 질문이다. 만약 해당 성향에 대한 민감도가 높다면, 룸메이트의 해당 성향을 중요하게 생각한다는 뜻이기 때문에 유사도 측정 시에 해당 성향에 가중치를 두어 계산했다.

```
def sensitive_score_similarity_ranking(u_data, f_name):

    result = []

    with open(f_name, 'r', encoding='utf-8') as f:
        dataset = json.load(f)

    dataset, user_data = fse.modify(dataset, u_data)
    del dataset[-1]
    dataset.append(user_data)
    dataset = check_cluster(u_data, f_name)

    df = pd.DataFrame(dataset)
    n_array = df.to_numpy()

    IDs = n_array[:, 0]
    features = n_array[:, 1:]

    euclidean_distances_result = euclidean_distances(features, features)
    ranking = np.argsort(euclidean_distances_result)[-1]
    ranking = ranking.tolist()

    for rank in ranking:
        result.append(IDs[rank])
    if u_data["userId"] in result:
        result.remove(u_data["userId"])

    return result
```

④ 찜 기반 협업필터링 구현

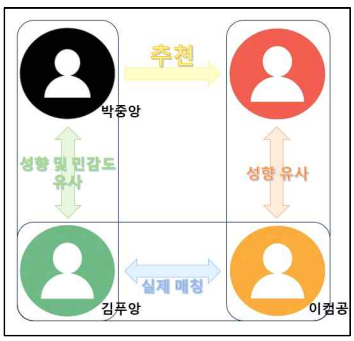
박중앙이 김푸앙을 찜한다면, 김푸앙과 성향 수치가 유사한 다른 사용자를 추천할 수 있는 알고리즘이다. 성향만으로 유사도 측정 및 순위를 매기는 함수를 따로 만들어 이용한다.




```
def wishlist_filtering_recommend(u_data, f_name):

    res, result = [], []
    wishlist = get_wishlist_data(u_data)

    for wish in wishlist:
        res.append(score_similarity_ranking(get_userdata(wish, f_name), f_name)[0:20])

    for i in range(20):
        for j in range(len(res)):
            temp = res[j][i]
            if temp not in result and temp != u_data["userId"]:
                result.append(res[j][i])

    return result
```

⑤ 실제 매칭 데이터 기반 협업필터링 구현

박중앙이 김푸앙과 성향 수치 및 민감도가 유사하다면, 김푸앙과 실제 매칭된 이컴공의 성향 정보를 이용하여 이컴공과 성향 수치가 유사한 다른 사용자를 추천할 수 있는 알고리즘이다. 성향과 민감도 모두 사용하여 유사도 측정 및 순위를 매기는 함수를 추가적으로 이용한다.




```
def result_filtering_recommend(u_data, f_name):

    res, result = [], []
    matching = get_matching_data()
    similar = total_similarity_ranking(u_data, f_name)[0:5]

    for sim in similar:
        for match in matching:
            if sim == match["userId"]:
                res.append(score_similarity_ranking(get_userdata(match["matchingId"], f_name), f_name)[0:20])
                break
            elif sim == match["matchingId"]:
                res.append(score_similarity_ranking(get_userdata(match["userId"], f_name), f_name)[0:20])
                break

    for i in range(20):
        for j in range(len(res)):
            temp = res[j][i]
            if temp not in result and temp != u_data["userId"]:
                result.append(res[j][i])

    return result
```

⑥ 백엔드-AI 연동을 위한 API 구현

가. 추천 리스트 반환 API

input : {"ID": str, "_score": int, "_sensitivity": int, ...}

output : {"ID1": str, "ID2": str, ...}

나. 추천 리스트 유사도 반환 API

input: {"ID": str}

output: {"similarity1": float, "similarity2": float, ...}

다. 추천 리스트 업데이트 및 반환 API

input: {"ID": str}

output: {"ID1": str, "ID2": str, ...}

라. 찜 목록 업데이트 API

input: {"ID": str, "wishID": str}

output: {}

마. 매칭 데이터 업데이트 API

input: {"ID": str, "matchingID": str}

output: {}

바. 룸메이트 구하는 상태 업데이트 API

input: {"ID": str, "state": bool}

output: {}

11. 개발 일정

4주차 ~ 16주차 개발 계획

진행 단계	수행 내용	주 기능 구현						부 기능 구현			부족한 부분 보완		발표	리뷰
		4주 (9/22~9/28)	5주 (9/29~10/5)	6주 (10/6~10/12)	7주 (10/13~10/19)	8주 (10/20~10/26)	9주 (10/27~11/2)	10주 (11/3~11/9)	11주 (11/10~11/16)	12주 (11/17~11/23)	13주 (11/24~11/30)	14주 (12/1~12/7)	15주 (12/8~12/14)	16주 (12/15~12/21)
		교수님 피드백		멘토링 피드백		중간고사	교수님 피드백	멘토링 피드백		중간 대모	멘토링 피드백		최종 대모	보고서 제출
기획	요구사항 분석													
	아이디어 제안													
	UI 디자인													
	아이디어 수정													
설계	전체 아키텍처 설계													
	데이터베이스 설계													
개발 (백)	성향조사 기능													
	추천 리스트 기능													
	채팅 기능													
	기숙사 합격 인증 기능													
	모임 기능													
개발 (프론트)	UI디자인													
	성향조사 테스트 UI													
	성향조사 결과 UI													
	룸메이트 매칭 UI													
	로그인 UI													
	모임 UI													
	UI-백 연동 테스트													
개발 (AI)	클러스터링 구현													
	유사도 측정 및 추천 기능													
	협업 필터링 구현													
피드백	배포													
	유저 피드백													
	최종 보고서 작성 및 발표													

계획에서는 11주차, 12주차에 기능 구현을 거의 끝마치고, 성능 향상 및 유저 피드백을 통해 서비스를 개선하려고 했는데, 생각보다 기능 구현의 난이도가 어렵거나 연동 오류가 발생하는 경우가 간혹 있어서 생각보다 기능 구현이 조금 늦어졌다.

12. 기대 효과

① 간편하게 룸메이트 추천받기

- 기존 에브리타임 등의 커뮤니티에서 룸메이트를 찾을 때보다 훨씬 간편하게 룸메이트를 찾을 수 있다.
- 어느 정도 정해진 질문과 유형이 있기 때문에 특정 성향에 대한 누락을 걱정할 필요 없이 질문에 따라 사용자의 성향과 민감도를 쉽게 수치화할 수 있다.

② 나와 잘 맞는 룸메이트 추천받기

- 기숙사 홈페이지나 커뮤니티를 이용하는 것보다 훨씬 사용자와 잘 맞는 룸메이트를 추천받을 수 있다.
- 본인과 성향이 유사한 사용자뿐만 아니라, 협업 필터링을 통한 다양한 사용자를 추천받을 수 있다.

③ 나와 잘 맞는 모임 추천받기

- 룸메이트 추천을 위한 성향 조사 결과를 이용하여 사용자와 성향이 유사한 사람들이 만들었거나 유사한 사람들이 많이 속해있는 모임을 추천받을 수 있다.

④ 룸메이트 성향 조사에 성실하게 답하기

- 사용자는 룸메이트 추천을 위한 성향 조사 질문의 수가 많다고 느낄 수 있지만, 본인의 동물 유형을 확인할 수 있게 만듦으로써, 좀 더 흥미를 느끼고 정확한 결과를 위해 조사에 성실하게 임하게 할 수 있다.

13. 향후 계획

① 기숙사생들에게 필요한 기능 추가

- 초기 계획에 있었지만, 구현 과정에서 우선순위가 뒤로 밀린 공지사항 알림, 식단표 알림 기능 등을 추가할 계획이다.

② 타겟층 확장

- 현재 중앙대 기숙사생들을 타겟으로 하여 개발을 진행했지만, 많은 대학 기숙사에서 사용할 수 있도록 각 대학의 특징들을 반영할 수 있도록 업데이트할 계획이다.

③ 베타 테스트 및 대학 기숙사와 협력 추진

- 어느 정도 완성된 서비스를 배포하여 사용자들의 피드백을 수용하고 성능을 향상시켜 실제 대학 기숙사에서 사용할 수 있도록 만들 계획이다.