팀2 : 오예스 최종 보고서

과목명	모바일프로그래밍	
지도교수	이창우 교수님	
학과	소프트웨어학부	
팀명	팀2: 오예스	
팀원	20203177 권해담	20213360 김혜은
	20212996 박수연	20213015 송규원
	20213027 유다영	
제출일자	2022.12.09	

[문제해결 및 공유사항]

1. 데이터 객체 전달

   사용자가 키워드로 영화를 검색하고, 이 중 하나를 선택하면 영화 상세 페이지로 이동하도록 구현되어 있다. 이때 선택한 영화의 데이터를 함께 전달하여 상세 페이지에서도 영화의 제목과 포스터를 그대로 볼 수 있도록 구현하려 했으나, 전체적인 구조를 Fragment를 사용하였기 때문에 트랜잭션을 통해 자동 전환이 발생하여 MovieMainData(영화 제목과 포스터 정보를 가진 데이터 객체) 객체를 상세 페이지 Fragment에 전달할 수 없는 문제가 발생하였다.

   Intent를 통해 Activity 간 데이터 객체 전달을 한 것처럼, Fragment 외부에서 Fragment로 데이터를 Bundle 객체 형식으로 전달할 수 있는 setArgument() 메소드를 사용하여 해당 문제를 해결하였다.

   Bundle 객체에 MovieMainData 객체를 넘겨주기 위해 키 값과 MovieMainData를 매개 변수로 갖는 bundle.putSerializable() 메소드를 사용하고, MovieMainData 클래스에서 Serializable 인터페이스를 구현하였다. 해당 내용은 recyclerView를 클릭하였을 때 발생하는 이벤트이므로, SearchFragmentAdapter.java의 setOnClickListener() 함수 내에서 구현해야 한다. HomeActivity.java의 onChangeFragment() 메소드를 호출하여 MovieMainData 객체를 매개 변수로 전달하고, onChangeFragment() 메소드 내에서는 상세 페이지 Fragment로 Bundle 객체를 전달해 주면서 반대로 키 값을 매개 변수로 갖는 getSerializabel() 메소드를 통해 영화 데이터 객체를 받아 사용 가능하게 하였다.

2. RecyclerView Adapter 파일에서 Fragment로의 화면 전환

   RecyclerView 내 특정 아이템 클릭 시 상세 페이지로 넘어가도록 구현하는 과정에서 어려움이 있었는데 Activiry를 이용한 Fragment to Fragment 전환 메소드를 사용하여 해당 문제를 해결하였다. Activity 객체를 Adapter를 호출할 때 같이 전달하여 Activity 내 구현해 둔 Fragment to Fragment 메소드에 접근해 Fragment로 화면 전환을 할 수 있도록 구현하였다.

   RecyclerView 각 아이템의 setOnClickListener() 메소드는 다른 여타 뷰들과 달리 Fragment.java 파일이 아니라, 무조건 Adapter 파일 내에서 관리하도록 되어 있어 문제가 발생하였다. Fragment로 화면 전환 시 HomeActivity.java 내 onChangeFragment() 메소드를 사용하여야 하고, 결과적으로 해당 메소드를 Adapter 파일 내에 setOnClickListener () 메소드 내부에 구현하여야 하는데, Adapter 파일은 HomeActivity.java의 객체

에 직접적으로 접근이 불가능하다. 따라서 HomeActivity.java에 직접적으로 접근이 가능한 Fragment에서 Adapter의 생성자가 호출이 될 때, 객체를 전달하여 onChangeFragment() 메소드를 사용하여 화면 전환이 가능하게끔 구현하였다.

3. Fragment 간 뒤로가기 구현

Fragment 간 작업이 진행되는 도중 뒤로가기 버튼을 눌렀을 때 그 전 Fragment를 띄워주는 것을 예상하였다. 하지만, 뒤로가기 버튼을 누르면 그 전 Fragment를 띄워주는 것이 아닌 Fragment를 다루던 이전 Activity로 돌아가는 현상을 발견하였다.

Fragment 간의 원활한 커뮤니케이션을 위해 Fragment Manager의 FragmentTransaction를 호출해서 addToBackStack() 메소드를 활용하여 해당 문제를 해결할 수 있었다. addToBackStack() 메소드는 stack에 열린 Fragment를 LIFO 형식으로 저장해서 이전 Fragment로 돌아가야 하는 이벤트가 발생하면, stack에서 POP을 해주는 메소드이다.

FragmentTransaction의 replace 메소드를 활용해서 remove, add 과정을 거치는 것이 아닌 add 메소드로 화면을 이동시켜 주는 것이다.

4. 동적인 RecyclerView에 이미지 객체 전달

추천 알고리즘 결과값이 적용된 HomeFragment.java 내 추천 영화 RecyclerView는 사용자의 MBTI에 따라 추천 알고리즘의 결과값이 다르기 때문에 사용자에 동적인 RecyclerView를 구현하였다. 추천 영화에 쓰이는 RecyclerView는 우리가 구현한 어플리케이션의 여타 다른 RecyclerView들과 별도의 데이터 객체 파일을 생성하여 사용하였는데, RecyclerView에 이미지 객체를 전달할 때 타입 차이가 있을 것이라 예상하였기 때문이다.

착안 방법은 추천 알고리즘의 대상이 될 데이터 파일의 파이어베이스 항목 각각에 이미지 URL을 저장하여 필요시 다운로드 하는 것이다. 파이어베이스에서 이미지 URL을 가져와 이미지 Node 라이브러리로 띄워주는 방식인데, 이때 RecyclerView에 전달될 이미지 객체는 String 타입이어야 한다. 그러나 이미지 URL 저장 방식에서 어려움을 겪었고 메모리 용량을 최소화할 수 있는 방식이라고 생각하였는데 구현하지 못하였다.

차선책은 추천 알고리즘의 대상이 될 모든 데이터들의 이미지 파일을 drawable 디렉토리에 저장하는 것이다. 이를 R.drawable을 이용하여 전부 리스트에 저장하고, 인덱스를 통해 접근한다. 인덱스는 1대1 매칭이 가능하도록 추천 알고리즘 결과값에 해당하는 인덱스가 별도의 리스트에 저

장되며, 이때 RecyclerView에 전달될 이미지 객체는 int 타입이어야 한다.

추천 알고리즘의 대상이 되는 데이터 파일의 용량이 작고, 유동적이지 않기 때문에 가능한 구현 방법이었다. 추천 알고리즘의 대상이 되는 데이터 파일이 한정적이고 해당되는 모든 이미지를 integer 리스트에 저장하였기 때문에 하드코딩의 느낌이 강하고 코드 가독성 또한 떨어진다. 데이터베이스 서버와 연동하여 해당 문제를 해결할 수 있는 방법을 고안해볼 예정이다.

5. 추천 알고리즘 데이터 파일

원래 계획은 추천 알고리즘의 대상이 되는 데이터 파일을 파이어 베이스에 올리는 것이 아닌, CSV 파일을 제작하여 자바 내 File 클래스를 이용하여 읽어올 계획이었다.

그러나 두 가지 문제가 존재했는데, 첫 번째는 CSV 파일을 읽어오는 과정이다. 알고리즘 구현 당시 테스트를 위해 CSV 파일의 절대 경로를 입력하였는데, 안드로이드 스튜디오 실행 환경 상에서 이를 인식하지 못하였다. getResource(), getPath() 메소드를 이용하여 현재 작업 경로를 읽어오고 이를 String 타입 변수에 저장한 다음 데이터 파일 이름을 임의로 붙여주는 방식으로 상대 경로로 입력해 보았는데, 이클립스 실행 환경과 다르게 안드로이드는 상대 경로 또한 인식하지 못하였다. 이는 CSV 파일로 제작해 두었던 데이터 파일을 JSON 형식으로 다시 제작하여 이를 파이어베이스에 저장하는 것을 통해 해결하였다.

두 번째는 파이어베이스의 시간 지연 문제이다. 처음에는 파이어베이스를 통해 데이터 파일을 읽어오는 과정을 Fragment가 호출될 때마다 반복하였다. 읽어오는 데에는 문제가 없었으나 여기에 알고리즘 로직을 추가하니 데이터 파일을 파이어베이스에서 읽어오기 전에 알고리즘 로직이 먼저 동작하는 문제가 발생하였다. Fragment가 호출될 때마다 데이터 파일을 읽어오는 것은 시간이 오래 걸릴뿐더러 Fragment가 호출될 때마다 메모리 용량을 차지하여 비효율적일 것이라고 판단하였다. 결과적으로 파이어베이스를 통해 데이터를 읽어오는 과정을 사용자가 어플리케이션에 로그인 할 때 최초로 딱 한 번 수행되도록 로직 구현 위치를 변경하였다.

6. Fragment 중첩

어플리케이션 전반적으로 메모리 부담을 덜기 위해 Fragment를 사용하였다. HomeFragment.java에는 박스 오피스 순위와 추천 영화에 해당하는 총 두 개의 RecyclerView가 존재하는데, 작업자가 다르기 때문에 프로젝트 merge 시 충돌이 우려되어 별도의 Fragment에서 작업한 후, 최종으로 HomeFragment.java 위에 중첩으로 두 개의 Fragment를 올릴 생각이었

다. 중첩 Activity는 xml 코드로 간단하게 작업할 수 있는 반면 Fragment는 동적으로 중첩 처리가 필요한데, 해당 과정에서 어려움을 겪었다. 별도의 Fragment에서 구현을 한 후 이를 한 Fragment에 합치는 것으로 방식을 바꾸었고, RecyclerView 각각이 하나의 Fragment를 차지하는 것이 아닌 한 Fragment 안에 병렬로 배치되도록 구현하였다.

7. 데이터베이스 구조 문제

데이터베이스의 데이터 구조는 userUid 안에 email, mbti, name, title이 저장되어 있고, title 안에는 review, date, posterUrl, rating을 저장해 두었다. 영화의 개수가 유한한 것이 아니기 때문에 userUid를 통해 데이터베이스에 접근해야 하는데, key값을 email, mbti, name을 If문을 통해 없애고 title만 남도록 완전히 정제해야 했다.

title 내에서 poster의 경우 bitmap 타입으로 저장해야 했는데, 데이터베이스에 bitmap으로 직접적으로 저장할 수 없기 때문에, 데이터베이스에 저장할 때는 String, 가져올 때는 bitmap으로 바꾸어 주는 별도의 함수를 구현하였다.

8. 파이어베이스 시간 지연 문제

Firebase authentication을 통해 회원가입을 하고 여기서 userUid를 받아 key값으로 realtimeDatabase를 만들어야 했다. 하지만 userUid를 받고 해결하는데 있어 시간지연 문제가 발생하였다. 이를 thread.sleep을 사용하여 값을 받아오는 시간을 지연시키는 것을 통해 해결하였다.

현재는 어플리케이션의 효율을 떨어뜨리는 쪽으로 구현되어 있지만, 수업시간에 다른 조들의 발표를 들으며 동기, 비동기 개념을 통해 좀 더 깔끔하게 구현할 수 있음을 알았다. 동기, 비동기 개념을 활용한 방식을 다시 한번 고안해 볼 예정이다.

9. Fragment 간 데이터 전달

(1) 영화 목록 화면(Search Fragment)에서 영화 상세 화면 (MoiveDetailFragment)으로 클릭한 영화에 대한 데이터 전달

(2) 감상평 목록 화면(Review Fragment)에서 감상평 상세 화면 (ReviewDetailFragment)으로 클릭한 감상평에 대한 데이터 전달

(3) 영화 상세 화면(MovieDetailFragment)에서 감상평 상세 화면 (ReviewDetialFragment)으로 감상평을 보거나 작성하러 가기 위해 데이터 전달

초기에는 String 타입의 영화 이름을 전달하여 이동한 Fragment에서 감상평 혹은 영화 데이터의 개수만큼 반복문을 돌려 같은 영화 제목을 가진 데이터만큼 보여주는 것으로 구현을 했다. 하지만 Fragment를 이동할 때마다 반복문을 사용하면 시간 복잡도가 O(n)만큼 걸릴 뿐만 아니라 효율적이지 못한 코드라는 판단을 하였다. 또한, 상황 (3)의 경우에서 감상평 데이터 유무에 따라 보여지는 레이아웃이 다른데 Fragment 이동 전에 이 유무를 따져서 아예 클래스 객체 자체를 전달할 필요가 있다고 생각했다.

따라서 감상평(ReviewMainData.java)과 영화 (MovieMainData.java) 클래스에 Serializable 인터페이스를 상속하고, 해당 영화에 대한 감상평이 있다면 영화 homeActivity.java의 onFragmentChange() 메소드를 통해 Bundle 객체에 “아이템”이라는 키와 영화/ 감상평 클래스 객체를 전달시켰다. Serializable 클래스를 사용한 이유는 Bundle객체에서 전달하고 받을 때 직렬화 가능한 객체를 넣어야 하기 때문이었다. 이동한 프래그먼트(ReviewDetailFragment)의 자바 파일에서 Serializable 타입의 item 변수를 선언하여 영화 데이터를 전달 받았을 때는 다시 각각의 영화/ 감상평 클래스 객체로 다운캐스팅 처리하여 각 경우에 맞게 레이아웃이 보여지게 구현하였다.

10. 선택 값 유무에 따른 버튼 활성화

회원가입 화면에서 입력 값이 아예 없을 때의 비활성화 기능은 잘 구현되었지만, 입력 값을 썼다가 지운 경우에도 버튼이 활성화 되어있는 오류가 발생하였다.

이는 addTextChangedListener의 afterTextChanged() 메소드를 오버라이딩하는 함수와 입력 값을 확인하는 setIbRegisterEnableDisable() 메소드를 동시에 호출하고, 항상 이 메소드를 호출해 확인할 수 있도록 수정하였다.

11. Fragment 호출 시 데이터 증가 문제

특정 Fragment(감상평 목록 화면, 홈 화면)로 이동할 때마다 데이터들이 계속 늘어나는 오류가 발생하였다. 강의 자료를 찾아보거나 프론트 팀 내 코드 비교를 통해 이는 Fragment 생성주기를 간과하고 있어 생긴 오류라는 것을 확인할 수 있었다. 감상평 데이터가 담긴 arrayList를 onCreateView() 밖에서 선언하여 초기 데이터의 추가적인 증식을 막았습니다.

12. 아이템 삭제 기능

감상평 목록에서 편집 모드일 때 삭제 기능을 구현하는 과정에서 연달아 있는 감상평 아이템을 삭제하면, 삭제된 아이템의 다음 아이템이 삭제되지 않는 오류가 발생하였다. 이는 아이템이 삭제되어 인덱스들이 앞으로 1씩 줄어들어 바로 다음 아이템으로 건너뛴 것으로 인해 발생한 오류였음을 파악했다. remove() 메소드를 사용한 코드 다음에 i-- 처리를 해주어 같은 인덱스에 해당하는 아이템의 체크박스 유무를 다시 한번 체크하여 오류를 해결하였다.

[목표 선정 피드백]

▼ 제품 요구사항 문서 URL

https://www.notion.so/PRD-a549bbb8df034fb0989a9227a0068cf3

[중간 점검 및 피드백]

1. 차별성

   초기 어플리케이션의 구현 계획은 영화 감상평을 작성하여 사용자가 필요 시 찾아볼 수 있는 수준에서 그쳤다. 다른 어플리케이션과 차별성을 갖추기 위해 메인 홈화면에 박스 오피스 순위를 API 연동과 웹 크롤링을 통해 보여주고, 그 하단에는 사용자에게 추천 영화를 보여준다. 하단에 사용자에게 보여지는 추천 영화의 가장 핵심적인 부분은 MBTI를 활용했다는 점이다. 이는 회원가입 시 사용자에게 MBTI 정보를 입력하도록 하여 해당 정보를 기반으로 사용자에게 추천 영화를 보여주며, 팀 명에서 유래되어 착안된 부분이다.

   추천 알고리즘은 영화를 장르별로 분류하고 이를 MBTI 항목 별로 매칭한 것이 아닌, 영화마다 MBTI 각 항목이 매칭되어 있어 사용자의 MBTI 정보를 받아와 사용자 MBTI 항목과 3개 이상 겹치는 영화를 추천해주는 간단한 알고리즘을 구현하였다.

2. 데이터베이스 구축

   사용자의 로그인 및 회원가입 활동을 용이하게 하기 위하여 인증 기능이 내장되어 있는 파이어베이스를 사용하였다. 파이어베이스는 IOP를 통해 서비스 인증 정보를 관리하기 때문에 하나의 계정으로 다수의 서비스 이용이 가능하다. 따라서 사용자는 로그인 및 회원가입 활동을 쉽게 할 수 있다. 또한 NoSql 클라우드 데이터베이스에 JSON 형태로 데이터를 저장하고 클라이언트에 실시간으로 동기화하기 때문에 데이터가 바뀔 때마다

실시간으로 업데이트를 송수신 할 수 있다. 소켓 기반 서버를 만들어서 통신하는 것보다 코드 양이 줄어들기 때문에 효율적인 측면에서 이점이 존재한다. 추가적으로 웹 및 서버로의 확장을 고려하여 SQLite가 아닌 파이어베이스를 이용하여 데이터베이스를 구축하였다.

데이터베이스에는 회원가입 시 입력한 사용자의 기본 정보, MBTI, 사용자가 작성한 감상평 목록이 저장되어 있다. 감상평의 경우 추가하거나 필요시 불러올 수 있으며, 추천 알고리즘의 대상이 되는 데이터 파일과 MBTI 정보 또한 저장되어 있어 기본적으로 알고리즘 로직 작동이 가능하게 한다. API를 연동하여 박스 오피스 순위를 보여주는 기능에도 파이어베이스의 기능이 일부 사용된다.

3. 크롤링의 위법성

자동화된 프로그램으로 웹사이트나 앱에 접속하여 각종 정보를 기계적으로 복제한 뒤 이를 별도의 서버에 저장하는 기술인 크롤링 행위에 대해 두 가지 법률적 쟁점이 존재한다. 첫 번째는 정보통신망법 위반, 두 번째는 저작권법 위반 문제이다. 이는 로봇배제표준, 즉 'robot.txt'을 확인하여 허용된 정보만을 크롤링하거나 애초에 타 정보 시스템에서 활용 가능하도록 고안된 Open API 사용을 권고한다.

9/24 아이디어 회의 (비대면)

회의 인원 👩🏻‍💻

- 총원 5 : 참여 5 불참 0

회의 안건 📚

1. 정기 회의 날짜 협의
2. 팀 주제 협의
3. 선정된 아이디어 구체화
4. 세부 결정 사항

1. **정기 회의 날짜 협의**

   매주 월요일 대면 회의

   장소 : 해동KL (도서관)

   시간 : 오후 6시 ~ 안건 종료 시

   매주 토요일 비대면 회의

   구글 미팅 사용

   시간 : 오후 9시 or 10시

2. **팀 주제 협의**

   왓챠 클론코딩, 국민팅/국팅 - 같은 대학교 사람들끼리 이어주는 앱, 심리 테스트 앱, 가상대학 기능 추가 버전, 쇼핑몰 가격 비교, 옷장 어플 클론코딩 등등…

3. **선정된 아이디어 구체화**

   선정된 아이디어 : AI 기반의 옷 코디 추천 서비스

   - 로그인/ 회원가입
   - 하단 카테고리 총 4개 (내 옷장, 추천 서비스, OOTD, 마이페이지)

a. 내 옷장 - 내가 가진 옷 사진을 찍어서 올려두는 곳, 카테고리 나뉨 (상의, 하의, 원피스, 신발.. 등등 추가 가능성 있음)

b. 추천 서비스 - 상단에 위치기반 실시간 날씨 정보 제공, 메인은 who/ where

   who- 교수님, 애인, 친구, 가족, 혼자

   where - 학교, 장례식, 결혼식, 운동, 발표, 외식, 직장

c. OOTD - 다이어리 느낌

d. 마이페이지 - 개인정보 조회 [닉네임, 이메일(아이디 겸용), 비밀번호] +변경, 로그아웃

4. **세부 결정 사항**

   협업 도구 : Notion

   버전 관리 시스템 : Github

   사용 언어 : Java

다음 회의까지 준비해야 할 것 ✅

필요한 기술 스택 조사 해오기.

09/26 아이디어 회의 (대면)

회의 인원 👩‍💻

- 총원 5 : 참여 4 불참 1 (사유 불참)

회의 안건 📚

1. 팀 주제 변경 및 대회 관련 논의
2. 개발 역할 분담
3. 10/05 발표 준비

1. **팀 주제 변경 및 대회 관련 논의**

   원래는 국회에서 주최하는 국회 open API 활용 경진대회를 계획 중에 있었으나, 팀원들이 추구하는 개발 방향과 맞지 않아 나가지 않는 것으로 결정.

   대회 조건 (국회 관련 open API를 필수적으로 사용하는 것)이 팀의 발전 가능성을 저해하며 다양한 사고의 확장에 방해 요소가 될 것이라 판단하여 팀원 모두의 합의 하에 결정된 사항임.

   9/24 비대면 줌 회의를 통해 팀 주제를 'AI 기반의 옷 코디 추천 서비스'로 결정하였으나, 팀원 개개인의 기술 스택에 비해 높은 수준의 개발이 요구된다고 느껴 실현 불가능할 것이라 판단.

   팀원 모두가 인공지능 요소가 포함된 안드로이드 앱 개발을 원해 해당 기술이 포함될 수 있는 다른 주제를 모색함. >> 국민대학교 학생 전용 만남 주선 어플리케이션

   아이디어 구체화 과정

   - 회원가입 - 메일 주소 입력 (도메인 국민대 웹메일로 제한) > 닉네임 > 성별 > 단과대 및 학번 (비공개 가능) > 해시태그 (나를 표현할 수 있는 것/ 내가 원하는 사람 유형 키워드, 총 두번 선택 안 하기 가능)
   - 홈화면 - 미팅 / 밥 / 수업 / 술 / 등교 및 하교 > 미팅은 매칭 시스템 > 그 외 게시판화 > 개인채팅 (미팅도 게시판도 있어야 함)

2. **개발 역할 분담**

   프론트엔드 - 김혜은, 박수연

   백엔드 - 권해담, 송규원, 유다영

      이미지프로세싱, DB 모델링, 추천 알고리즘 설계 (웹 크롤링..?)

3. **10/05 발표 준비**

   발표자 : 김혜은 (by. 사다리 타기)

   PPT 및 대본 준비 : 발표자 외 인원 전부 > 9/28까지 준비해서 발표자에게 줄 것.

다음 회의까지 준비해야 할 것 ✅

프로젝트에서 사용할 기술 스택/수강할 인강 조사 해오기.

10/01 아이디어 회의 (비대면)

회의 인원 👩🏻‍💻

- 총원 5 : 참여 4 불참 1 (사유불참)

회의 안건 📚

1. **팀 주제 변경**

1. **팀 주제 변경**

   팀원 개인의 개발 스택 역량, 실현 가능성, 개발 방향성 변경 등의 이유에 따른 팀 주제 변경.

   일기장 서비스 제공, 환경 보호 차원에 도움이 되는 기록 어플리케이션, 직접 작성한 영화 감상평 어플리케이션, 코딩 일기장, 알고리즘 공부 일기장 등 기록 서비스를 제공하는 무언가

   >> 결정된 사항 : 영화 감상평을 남겨 기록해 둘 수 있는 서비스 제공 어플리케이션 개발

다음 회의까지 준비해야 할 것 ✅

변경된 팀 주제 구체화 사항 (탑재 되어야 할 기능 등)

10/01 프론트엔드 회의 (대면)

회의 인원 👩🏻‍💻

- 총원 2 : 참여 2 불참 0

회의 안건 📚

1. 페이지 분담
2. UI 디자인

1. **페이지 분담**
    - 로그인 전 화면 - 회원가입/로그인 선택
    - 로그인 - 메일주소(아이디 대신), 비밀번호, 로그인버튼
    - 회원가입 -
      a. 웹메일 확인 - 메일 입력칸, 보내기 버튼, 인증번호 입력칸, 옆에 제한시간
      b. 내정보 입력 - 닉네임, 프사,성별, 단과대, 학번, 공개/비공개 라디오버튼. 에
      c. 내 태그 선택
      d. 이상형 태그 선택 - 해시태그 형식
      (1) 키 - ~150, 150~160, 160~170, 170~180, 180~
      (2) 성격 -
      (3) 얼굴상 -
      (4) 관심사 - 맛집탐방, 전시회 관람, 반려동물과 산책, 동네 산책, 쇼핑, 여행, 영화 감상, 음악 감상, PC방, 음주, 카페, 스포츠 경기관람, 솔직한 대화, 코딩, 만화, 코노,
      (5) MBTI - 16가지
      (6) 스타일 - 빈티지, 캐주얼, 스트릿, 모던, 댄디, 스포티, 레트로, 톰보이, 펑크
    - 홈 - 서비스 선택
    - 게시판
    - 매칭 화면
    - 마이페이지 - 회원 정보 수정,
      a. 회원 정보 수정 (에타처럼 카테고리)
      (1) 기본 정보 수정 화면 - 닉네임 변경, 비밀번호 변경, 프사, 단과대 및 학번 공개.

비공개 수정
(2) 태그 수정 화면 - 수정 확인 버튼

- 채팅
  a. 채팅 목록 - 상대방 이름, 마지막 대화 한 줄(가능하면), 현활표시처럼 새로운 채팅 알림(동그라미), 시간 순 정렬
  b. 채팅 화면 - 채팅은 알아서, 부가기능 - 사진 전송
- 프로필 - 게시판,채팅화면에 필요
- 모든 화면 통일 기능 - 하단바, 뒤로가기

2. **UI 디자인**

- 어플 로고 (여러개 생각) - 폰트 - 여기어때 잘난체
- 앱 아이콘
- 하단바 아이콘 - 채팅, 홈, 마이페이지

10/04 아이디어 회의 (대면)

회의 인원 👩🏻‍💻

- 총원 5 : 참여 5 불참 0

회의 안건 📚

1. 변경된 팀 주제 구체화 및 역할 분담
2. 어플 이름 정하기
3. 10/05 발표 준비

1. **변경된 팀 주제 구체화 및 역할 분담**

   변경된 팀 주제 : 영화 리뷰 및 기록 할 수 있는 서비스 제공

   >> 영화 정보 제공, 평점 등록 기능 (별점 혹은 아이콘) - 통계 제공 기능, 감상평 등록 기능, 로그인 (소셜 로그인 연동), 내용 검색 기능

   https://www.kobis.or.kr/kobisopenapi/homepg/apiservice/searchServiceInfo.do

   https://www.data.go.kr/data/3076402/openapi.do?recommendDataYn=Y

   API 끌어와서 영화 정보 제공

   역할 분담

   프론트엔드 : 김혜은, 박수연

   벡엔드 : 권해담, 송규원, 유다영

2. **어플 이름 정하기**

   수시로 단체 카카오톡 방에 의견 내기

3. **10/05 발표 준비**

   대본 및 피피티 오후까지 다시 제작해서 노션에 업로드

발표자는 위 자료 기반으로 연습해 주세요

다음 회의까지 준비해야 할 것 ✅

프로젝트에 필요한 기술 스택 조사, 프로젝트 세부 일정 어떻게 짤 지, 프론트/벡 팀 별 회의 다음 회의 전까지 진행 (세부 역할 분담 다음 회의 때 브리핑), 프로젝트와 관련하여 개인 공부 계획

10/08 프론트엔드 회의 (대면)

회의 인원 👩🏻‍💻

- 총원 2 : 참여 2 불참 0

회의 안건 📚

👁️ 기능 및 페이지

10/08 벡엔드 회의 (비대면)

회의 인원 👩🏻‍💻

- 총원 3 : 참여 3 불참 0

회의 안건 📚

1. 벡엔드 상세 역할 분담
2. PRD 구상

1. **벡엔드 상세 역할 분담**

   API 연동 - 해담

   추천 알고리즘 구축 - 규원

   DB 모델링 - 다영

2. **PRD 구상**

   프론트 회의록 참고하여 PRD 구상 후 프론트 팀에게 해당 사항 협의

   https://www.notion.so/PRD-a549bbb8df034fb0989a9227a0068cf3

   +) PRD 구상 및 일정 협의 후 카카오톡으로 어플리케이션 이름 결정 됨 >> **“MVTI”**

10/10 아이디어 회의 (대면)

회의 인원 👩🏻‍💻

- 총원 5 : 참여 5 불참 0

회의 안건 📚

1. 버전 관리
2. 시험 기간 팀 프로젝트 진행 방식
3. 전달 사항

1. **버전 관리**

   깃허브 협업 레파지토리 생성, 브랜치 사용법 익히기

2. **시험 기간 팀 프로젝트 진행 방식**
   - 별도의 아이디어 회의 (전체 회의) X
   - 팀별 회의 필요시 진행, 노션에 기록 남겨주기
   - 11월부터 본격적인 스터디 및 협업 진행 예정

3. **전달 사항**

   프론트 전달 사항 : UI 및 기초 Layout 제작 구상 해두기

   백 전달 사항 : 자신이 맡은 부분 기술 스택 조사, 인터넷 강의 참고할 것 있으면 조사

10/31 아이디어 회의 (대면)

회의 인원 👩‍💻

- 총원 5 : 참여 5 불참 0

회의 안건 📚

1. **전달 사항**
    - 시험 끝, 회의 재개 및 본격적인 스터디 & 협업 준비 → 참고할 인강 개별 준비
    - 프론트엔드/ 백엔드 팀별 진행 상황 수시로 업데이트 해주기 (회의록 남겨주세요!)
    - 깃허브 개별 브랜치 만들기, 커밋, 머지 등등… 연습 해보기
    - 각자 맡은 부분 기본적인 구상은 다음 회의까지 어느정도 갖추어져 있어야 함
    - 10주차에 교수님과 줌으로 있을 진행 현황 리뷰에서 자문 구할 부분, 피드백 받고 싶은 부분 정리하여 팀장에게 전달

11/02 프론트엔드 회의 (대면)

회의 인원 👩🏻‍💻

- 총원 2 : 참여 2 불참 0

회의 안건 📚

1. 페이지 분담
2. 세부 결정 사항

1. **페이지 분담**
    - 수연: 로그인, 회원가입-MBTI, 감상평 상세, 목록
    - 혜은: (회원가입-아이디/비번/비번확인), (영화 상세), (목록), (마이페이지)

2. **세부 결정 사항**

    11/5까지 각자 Figma 이용하여 페이지 디자인 마무리 후 회의 예정

11/06 프론트엔드 회의 (대면)

회의 인원 👩🏻‍💼

- 총원 2 : 참여 2 불참 0

회의 안건 📚

1. **UI 디자인**

   Figma 이용하여 UI 디자인 제작

https://www.figma.com/file/fG36loqczE1nT9Sj7at5MH/MVTI?node-id=0%3A1

11/07 프론트엔드 회의 (대면)

회의 인원 👩🏻‍💻

- 총원 2 : 참여 2 불참 0

회의 안건 📚

1. 디자인 진행 상황
2. 세부 결정 사항

1. **디자인 진행 상황**

   디자인 수정 완료

2. **세부 결정 사항**

   11/14일까지 xml 화면 구성해오기

11/07 백엔드 회의 (대면)

회의 인원 👩🏻‍💻

- 총원 3 : 참여 3 불참 0

회의 안건 📚

1. **결정 사항**

   해담 : 연동할 API 사이트 선별, 최신 자료 (User 관점에서 고려하기), 저작권 문제 등과 관련하여 Open API 사용을 권장

   다영 : 파이어 베이스 사용법 익히고 도움이 필요한 자료 준비 팀원들에게 도움 요청

   규원 : 모델 학습, 머신러닝 혹은 딥러닝 적용 방향 생각하고 인강 참고

11/14 프론트엔드 회의 (대면)

회의 인원 👩🏻‍💻

- 총원 2 : 참여 2 불참 0

회의 안건 📚

1. **결정 사항**

   xml 화면 확인 후 11/21까지 기능 구현

11/14 백엔드 회의 (대면)

회의 인원 👩🏻‍💻

- 총원 3 : 참여 3 불참 0

회의 안건 📚

1. **결정 사항**

   모델 학습에 어려움 존재

   → MBTI 16개 각 특성을 파악하기 힘들고 장르로 분류하기에도 각 MBTI와 매칭하기 어렵다는 점

   위 사항을 고려하여 MBTI 별 장르 기반 알고리즘 구현이 아닌 단순 구현 방식으로 변경 예정

   → 영화마다 MBTI 항목 4개를 매칭하여 사용자의 MBTI와 2개 or 3개 이상 겹치면 추천

   → 필요한 데이터 파일 준비하여 csv 파일 읽어오는 방식으로 구현 예정

   API 연동 필요한 프래그먼트 또는 액티비티 파악

   → 박스오피스 순위 노출에 쓰이므로 주기 결정이 핵심 (너무 길어도, 짧아도 문제가 될 것)

   데이터 베이스 필요한 프래그먼트 또는 액티비티 파악

   → 로그인/ 회원가입 기능 외에도 저장하고 불러올 부분이 필요하다면 함께 저장하는 방향으로

11/21 아이디어 회의 (대면)

회의 인원 👩🏻‍💻

- 총원 5 : 참여 5 불참 0

회의 안건 📚

1. 진행 방향 협의
2. 전달 사항

1. **진행 방향 협의**
    a. 액티비티 및 프래그먼트 UI 설계 완료
    b. 화면 전환 및 로그인/ 회원가입 기능 설계
    c. API 연동하여 필요한 부분에 삽입
    d. 데이터 라벨링 선별 필요 그 후 추천 알고리즘 설계
    e. 세부 기능 구현
    f. 충돌 및 에러 잡기 (Branch Merge)

2. **전달 사항**
    - 진행 속도 진전 필요, 스터디 횟수 줄이고 협업 횟수 늘리기 (대면 만남 횟수 증가 필요)
    - 깃허브 사용법 아직 익히지 않았다면 익히기
    - 프론트 UI 기본 프래그먼트 + 네비게이션바텀 사용 (메모리 부담 줄음)
    - 팀 중간 발표와 진행현황 리뷰 시 받았던 교수님 피드백 수시로 참고하고 반영하기

11/27 아이디어 회의 (비대면)

회의 인원 👩🏻‍💻

- 총원 5 : 참여 5 불참 0

회의 안건 📚

1. 11/28 발표 자료 준비
2. 11/30 발표 준비

1. **11/28 발표 자료 준비**

   개인 발표 자료 등록이기 때문에 PPT 디자인 통일성 필요 → 회의 후 미리캔버스에서 찾아서 보내기

   각자 발표 내용 겹치지 않게 발표하기로 정한 개별 소스 카톡방에 남겨주기

   자신이 어떤 것을 했고, 구현하며 어떤 어려움이 있었는지 또 이를 어떻게 해결하려 노력하였는지 그 과정을 위주로 기술할 것

   회의 마치고 팀장에게 학번/ 사진/ MBTI 보내주세요 -!

2. **11/30 발표 준비**

   발표 순서 팀장 → 프론트엔드 → 백엔드

   4분 넘기지 않게 연습 많이 하고 발표 전 날 되도록이면 만나서 서로 피드백 할 수 있는 부분 해주기

   프론트 내 발표 순서 : 수연 → 혜은

   백 내 발표 순서 : 해담 → 다영

12/01 아이디어 회의 (대면)

회의 인원 👩🏻‍💻

- 총원 5 : 참여 5 불참 0

회의 안건 📚

1. 팀 등록 자료 제작 분배
2. 전달 사항

1. **팀 등록 자료 제작 분배**
    - 회의록과 목표 선정 피드백, 중간점검 및 피드백 제작하여 취합 - **규원**
    - 문제 해결, 공유 사항 개별적으로 작성하여 팀장에게 보내면 취합하여 제출할 것
    - Readme.txt 제작 - **해담 혜은 수연**
    - 시연 영상 제작 및 편집 - **다영 규원**
    - 소스코드 - 이번 주까지 완성하기

2. **전달 사항**

    다 함께 노력했기에 발표까지 잘 마무리 할 수 있었다고 생각, 고생 많았습니다 :)!!

    이번주까지 구현 마무리 할 수 있게 조금만 더 다 함께 힘 써주고, 팀 구분 없이 안 되는 부분 서로 돕기