

**S EUNGKYU  
S ONGSUB  
A ND  
M OVIE  
F OR  
Y OU**

**팀장 : 유승규  
팀원 : 김송섭**

# 발표 순서



## Contents 1

팀장 및 팀원소개



## Contents 2

초기 계획과 구성 소개



## Contents 3

개발작 소개& 핵심기능 소개



## Contents 4

피드백&마무리

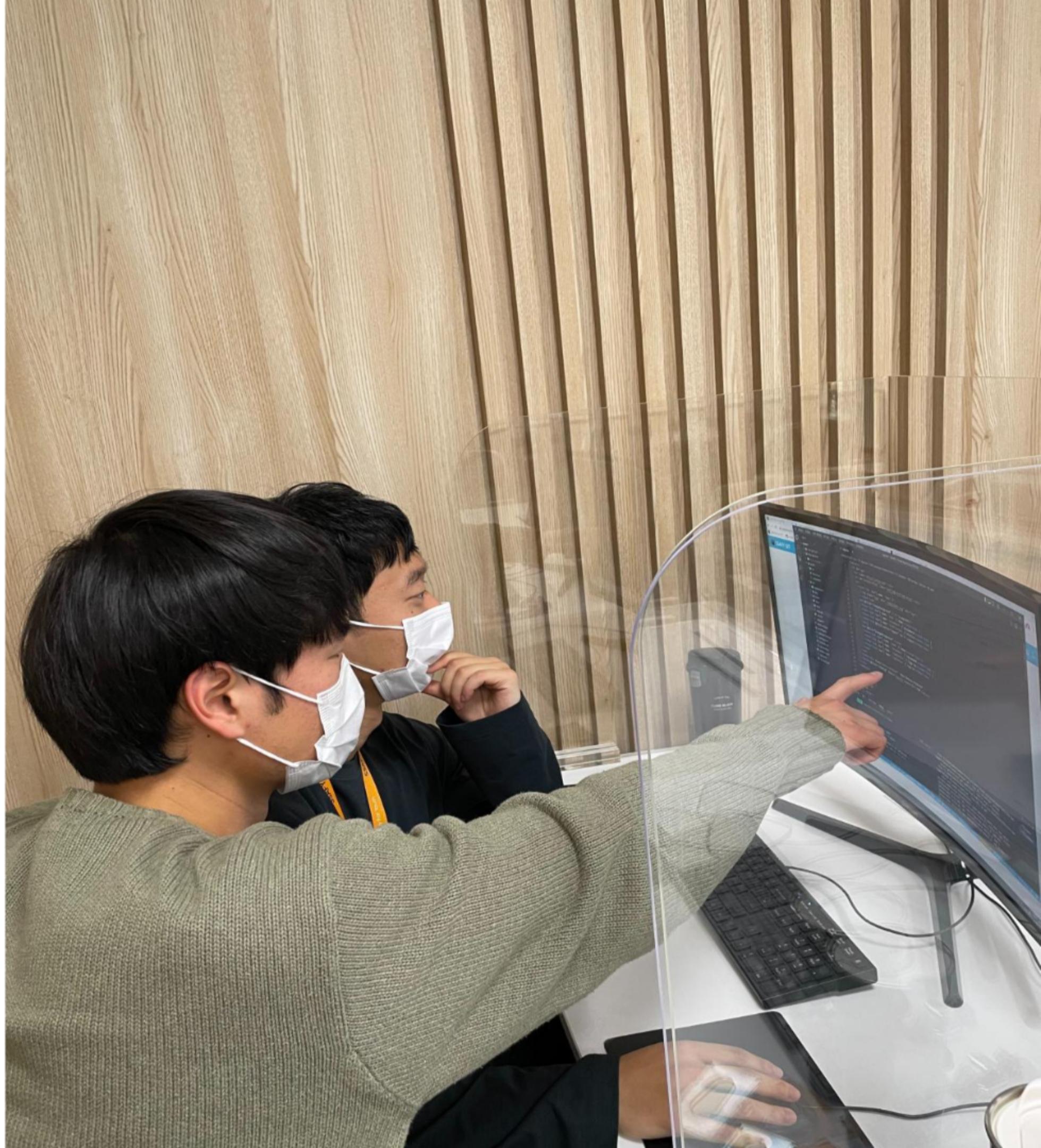
# 1

## Contents 1

팀장 및 팀원소개

팀장 : 유승규

팀원 : 김송섭



## 팀장

### 유승규



< 한전 원자력 연료 총무팀 인턴 20.07 >  
< 충남대학교 경제학과 학사 취득 22.02 >  
< 현 SSAFY 8기 >

사용자의 관점에서 개발하여 즐겁고  
다양한 경험을 제공하겠습니다.

- Vue Component 설계 및 routing
- 프로젝트 관리 및 UI 데이터 개선
- ERD 작성
- Django 기능 개선

## 팀원

### 김송섭



< GNU Systems Research Lab 인턴 수료 21.12 >  
< 국립경상대학교 항공우주 및 소프트웨어 공학과 학사 취득 22.02 >  
< 현 SSAFY 8기 >

개발 과정이 즐거워 사용자에게도 즐거움을  
전달 할수 있는 프로그램을 만들고 싶습니다.

- UI/UX
- Django 서버 관리 및 유지보수
- 데이터 정제
- Django DB - Vue 연동
- CSS 구성 및 개선

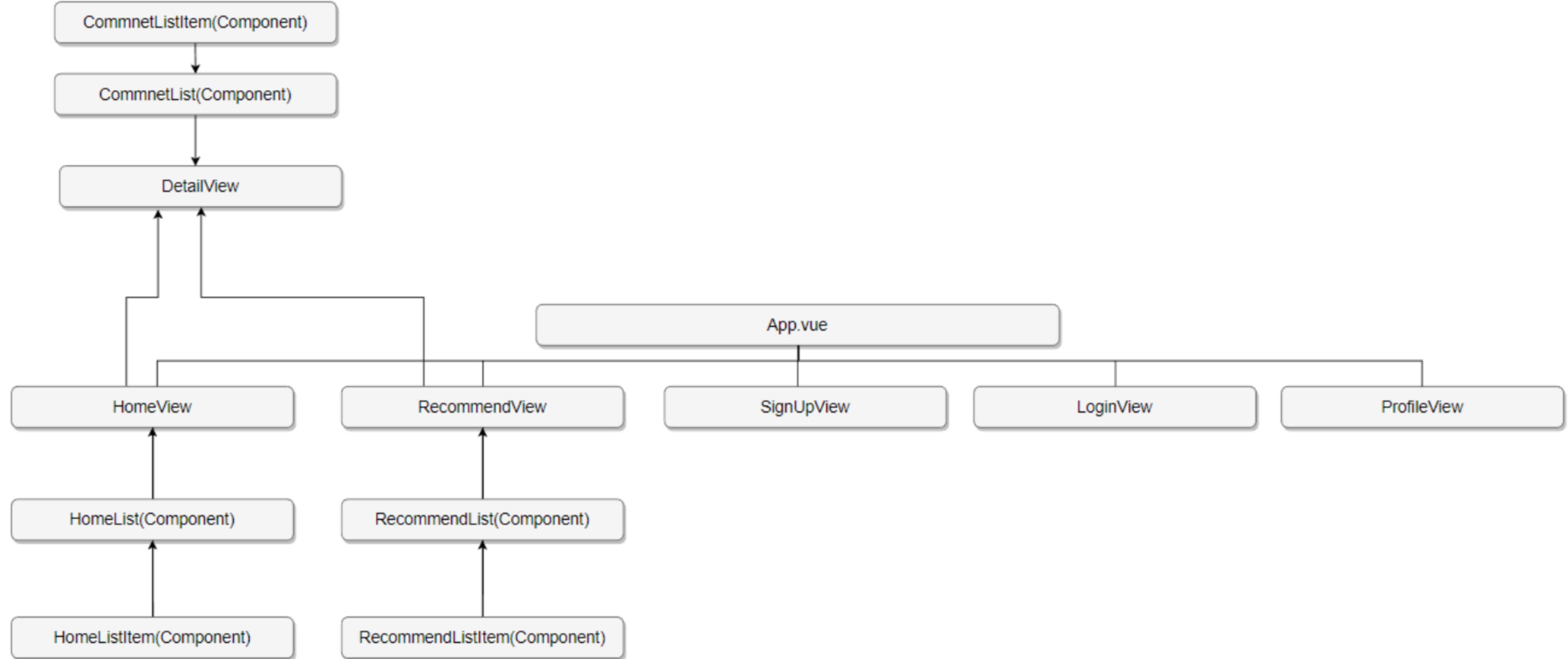


# 2

## Contents 2

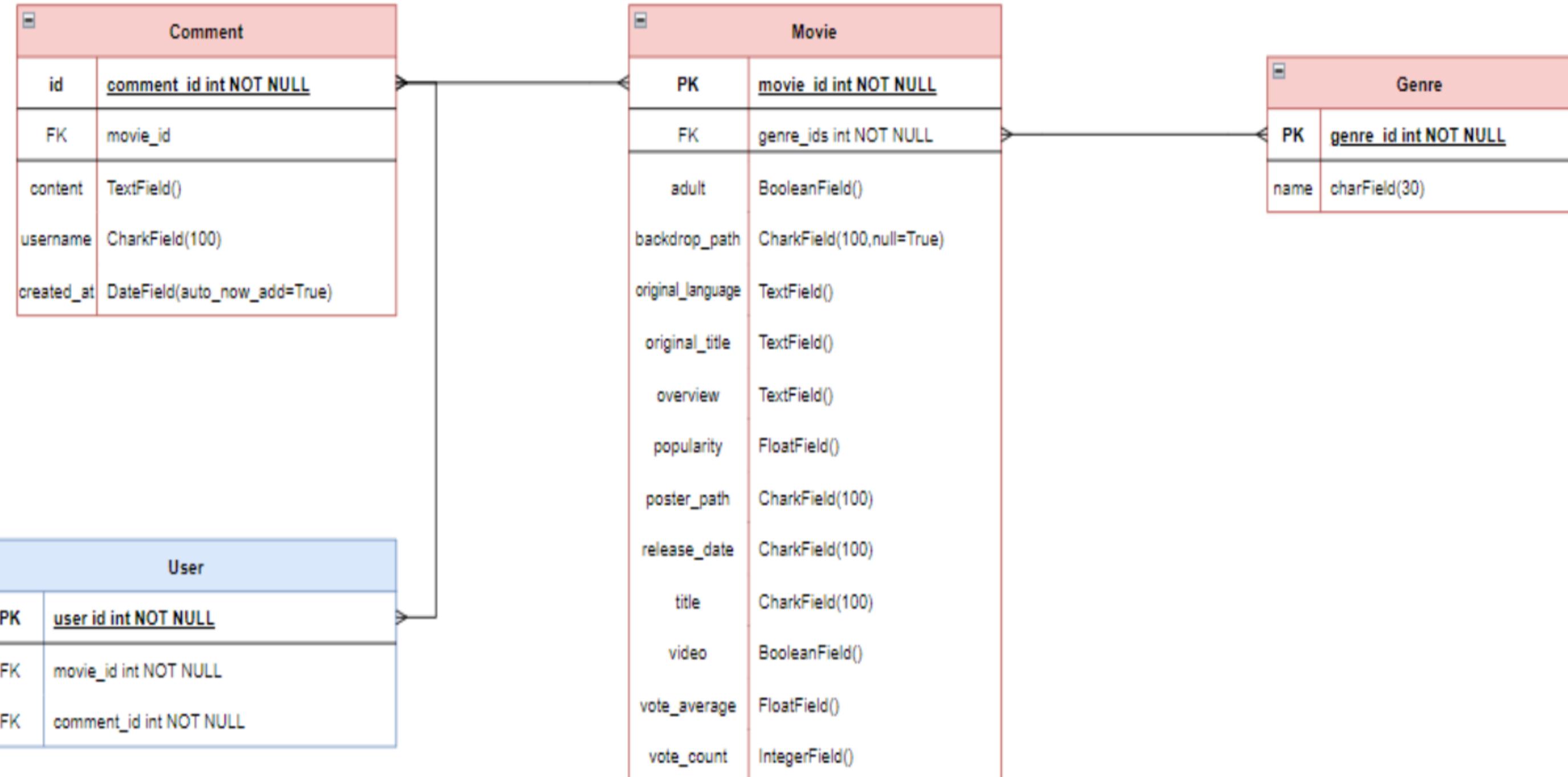
초기계획과 구성 소개

# 컴포넌트 구조

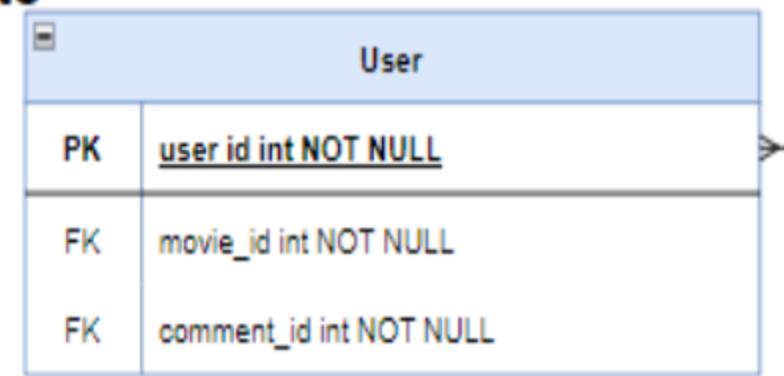


# ERD 구조

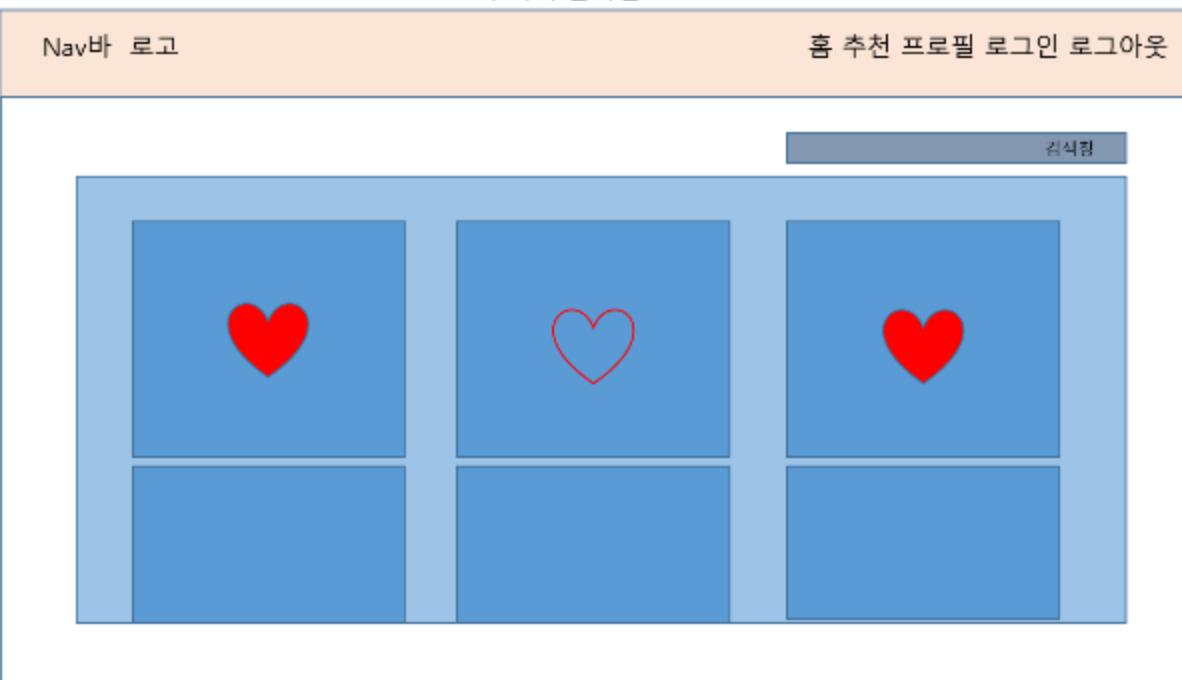
## movies



## accounts



### 초기 기획 홈화면



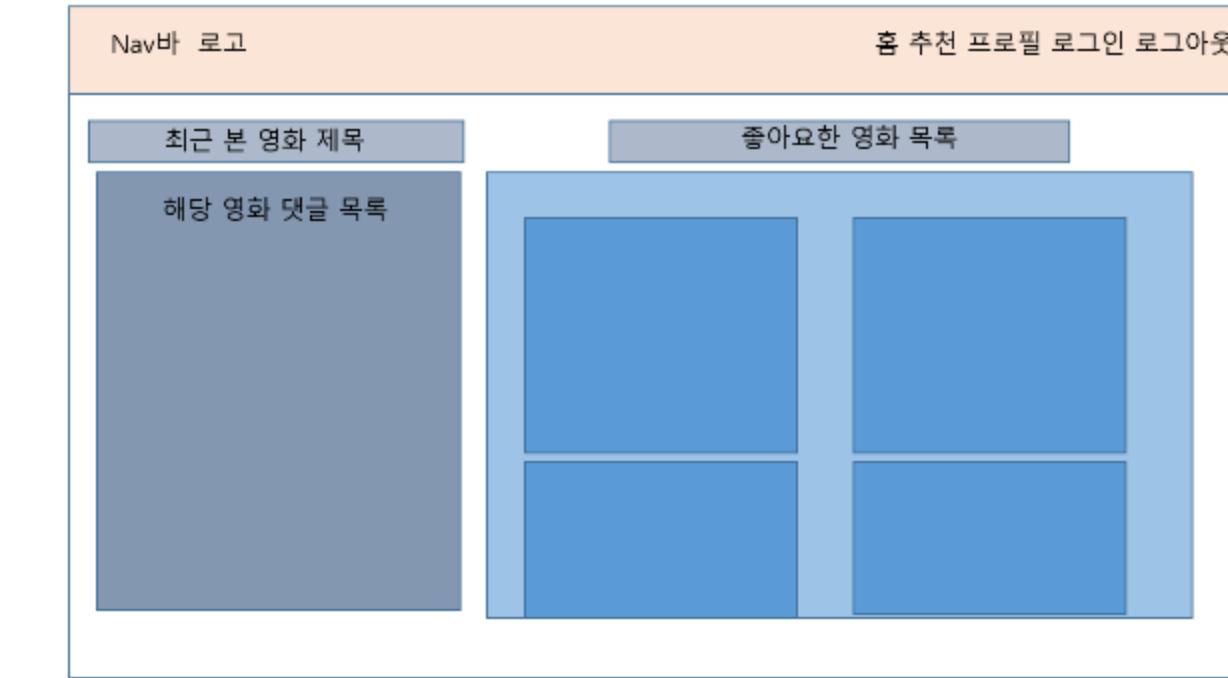
### 초기 기획 디테일화면



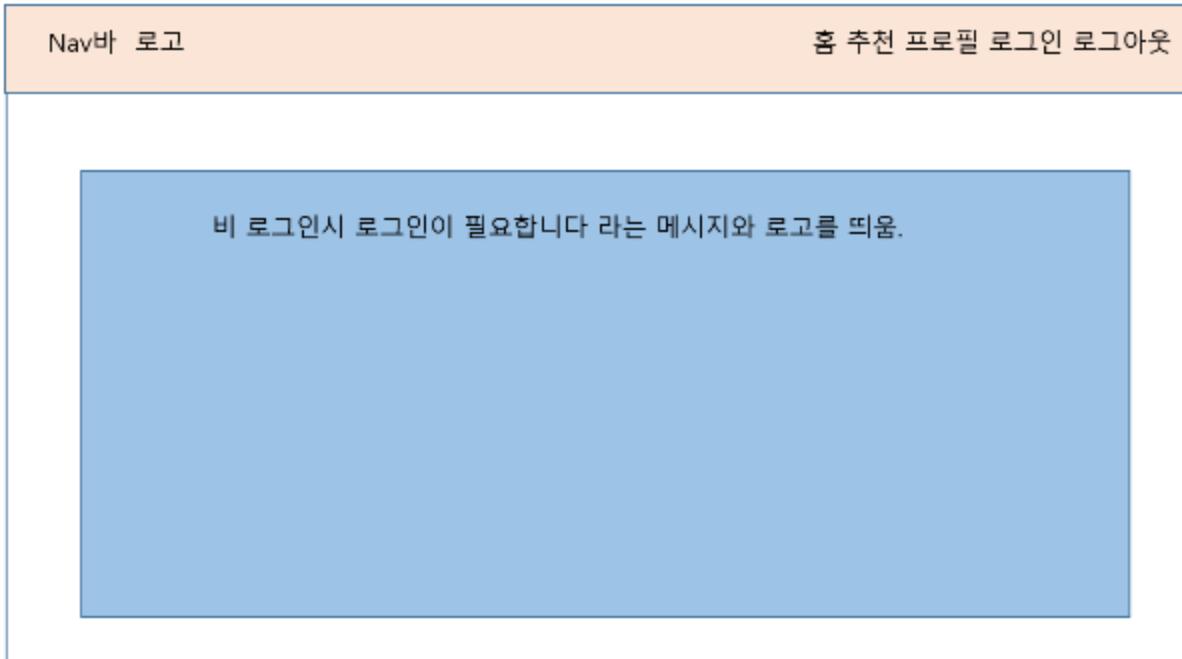
### 초기 기획 추천화면



### 초기 기획 프로필 화면



### 초기 기획 추천화면



### 초기 기획 회원가입 & 로그인



# 3

## Contents 3

개발작 소개 & 알고리즘 소개

S EUNGKYU  
S ONGSUB  
A ND  
M OVIE  
F OR  
Y OU

# 개발 진행과정

안전진압 ◁

전체 / 회의록

검색

업데이트

모든 팀스페이스

설정과 멤버

팀스페이스

전체

필수 구현 내용

팀 작업

- 캘린더
- 내 작업
- 모든 작업

그 외 할 일

회의록

프로젝트에 들어가며..

개인 페이지

시작하기

무제한으로 업그레이드

워크스페이스에서 1000개의 무료 불

륙 중 335(34%)개를 사용했습니다.

지금 업그레이드

새 페이지

회의록

커비 추가

설명 추가

회의록

모든 회의 +

발표 내용 스케치

회의록 221123

회의록 221122

회의록 221121

회의록 221118

회의록 221116 -2

회의록 221116

새로 만들기

8시간 전 편집 승 공유 ⚡ ⚡ ...

필터 정렬 Q ... 새로 만들기

2022년 11월 24일 오후 12:11

2022년 11월 23일 오전 8:02

2022년 11월 23일 오전 9:04

2022년 11월 21일 오전 10:05

2022년 11월 18일 오후 5:13

2022년 11월 16일 오후 11:20

2022년 11월 16일 오전 10:47

전체 / 회의록 / 회의록 221123

8분 전 편집 ⚡

## 과정 정리 및 마무리 계획

### 오늘 할 일

- 좋아요 기능 완성(디테일, 취소, 추천영화)
- 현재 상영중인 영화
- 댓글과 사용자 연동
- ERD 작성
- 좋아요 기반 알고리즘
- 실시간 API 적용

좋아요 누른것 해당 장르에 5+점  
 클릭시 해당 장르에 +1점  
 장르 점수를 백분율로 나눠서 장르마다 점수 퍼센트만큼 나오게 한다.  
 처음에는 균등점수 스타트 (한 장르만 나온는걸 방지)

추천목록에서  
 장르별로  
 data에 장르별로 점수를 5점씩 준다.  
 좋아요한 영화를 가져온다  
 좋아요한 해당영화의 장르를 가져와 해당 장르에 5점을 추가.

- 완료 -

영화 개당 장르마다 데이터에 가져온 점수를 더하고 장르의 수로 나눔  
 총합이 높은 순서대로 출력.

# 개발 진행과정

전체 / 회의록 / 회의록 221116-2

11월 18일 편집  공유

2일차(11/17) 기획

- 오전
  - 데이터 불러오기 및 정제
  - accounts 부분 구현
    - 1. sign up
    - 2. login
    - 3. logout
    - 4. 회원탈퇴
- 오후
  - 스크럼 회의를 통한 추천 알고리즘 작성
    - 1. 장르 추천
    - 2. 평점 추천
    - 3. 올 랜덤 추천
    - 4. 논문 기반 알고리즘 추천

그 후 개략적인 계획

토요일 중

- 기능부분 점검, 수정

일요일 중

- 컴포넌트 구조, 레이아웃 작성, 테마 결정

전체 / 회의록 / 회의록 221121

1일 전 편집  공유

+ 쟁점 수사

댓글 추가

- 오늘 할 일
  - 주말동안 작업 토론 및 병합
  - 예상차라
  - 프로필에 좋아요한 영화와 작성한 댓글 표시
  - 디자인 논의 및 결정
  - 알고리즘 고도화
- 앞으로 할 일
  - 11/22 무조건 좋아요
  - 좋아요 기반 알고리즘
  - css 디자인

# 개발 진행과정

전체 / 회의록 / 회의록 221116

3일차(11/18) 예상 계획

- 커뮤니티 기능 구현
  - 게시글 작성
  - 댓글 작성
  - 프로필 작성
    - 좋아요
    - 팔로우
  - 추천 알고리즘 기능 추가

실제 구현 내용

- API에서 많은 데이터 불러오기
- 영화 상세페이지 구현
- 커뮤니티 기능 구현
  - 영화 상세페이지 댓글 작성 & 불러오기
- 프로필 기능 구현
  - 작성자까지 진행

1. 로그인 시 로컬스토리지에 나의 이름 저장

2. 나중에 (좋아요 정보, 팔로우 정보...)이 필요할 때 로컬 스토리지에서 나의 이름(userName)을 Django에 보내서 요청

3. Django 입장에서는 이름(userName)을 통해서 요청한 Vue가 누구인지 알기 때문에

4. 그 사용자에 맞는 데이터(좋아요 정보, 팔로우 정보...)를 DB에서 탐색하여 Response()

영화 디테일에서

- 하단에 댓글쓰기 (input, submit)
- 댓글 표시(view) : {작성자, 내용, 작성 시간}
- 댓글 작성

※ 아래 대고 내용과 차이가 있다면 아래 대고 수정

11월 20일 면접

전체 / 회의록 / 발표 내용 스케치

4시간 전 관점

### 핵심 기능

- 알고리즘 기능 구현
  - 영화에 좋아요 버튼을 만들어 해당 영화를 유저의 좋아요 리스트에 담는다.
  - 영화를 좋아요 누르면 해당 영화의 장르에 가중치를 주어서 가중치 영화 리스트를 만든 후에 앞 부분, 즉 대체적으로 가중치가 높은 영화의 목록에서 영화를 골라 추천을 해준다.
  - 일반적인 장르추천과는 다른 점은 장르로 영화를 골라내리는 것은 알고리즘보다는 필터기능, 그것과는 다르게 가중치를 주어서 좋아요 누른 장르만 골라나오는게 아니라 해당장르의 해당하는 다른 장르와 겹쳐진 영화들을 보여준다. 즉 평소에 관심있던 영화장르가 아니더라도 본인이 좋아하는 영화장르와 결합된다면 추천화면에 보여질 수 있음.

### 대표 서비스

- 알고리즘기반 추천과 개인유저의 좋아요목록 표시
- 고전 명작과 독립영화

15분

1. 인트로

- 3초 쉬고, 인사 ⇒ 발표 방식&발표 순서
- 팀원 소개 ⇒ 역할 분담 등

2. 본문

초기 계획과 구성

- 초기 계획을 회의록과 초기 목표 화면 꼭쳐 보여줌

개발작 소개

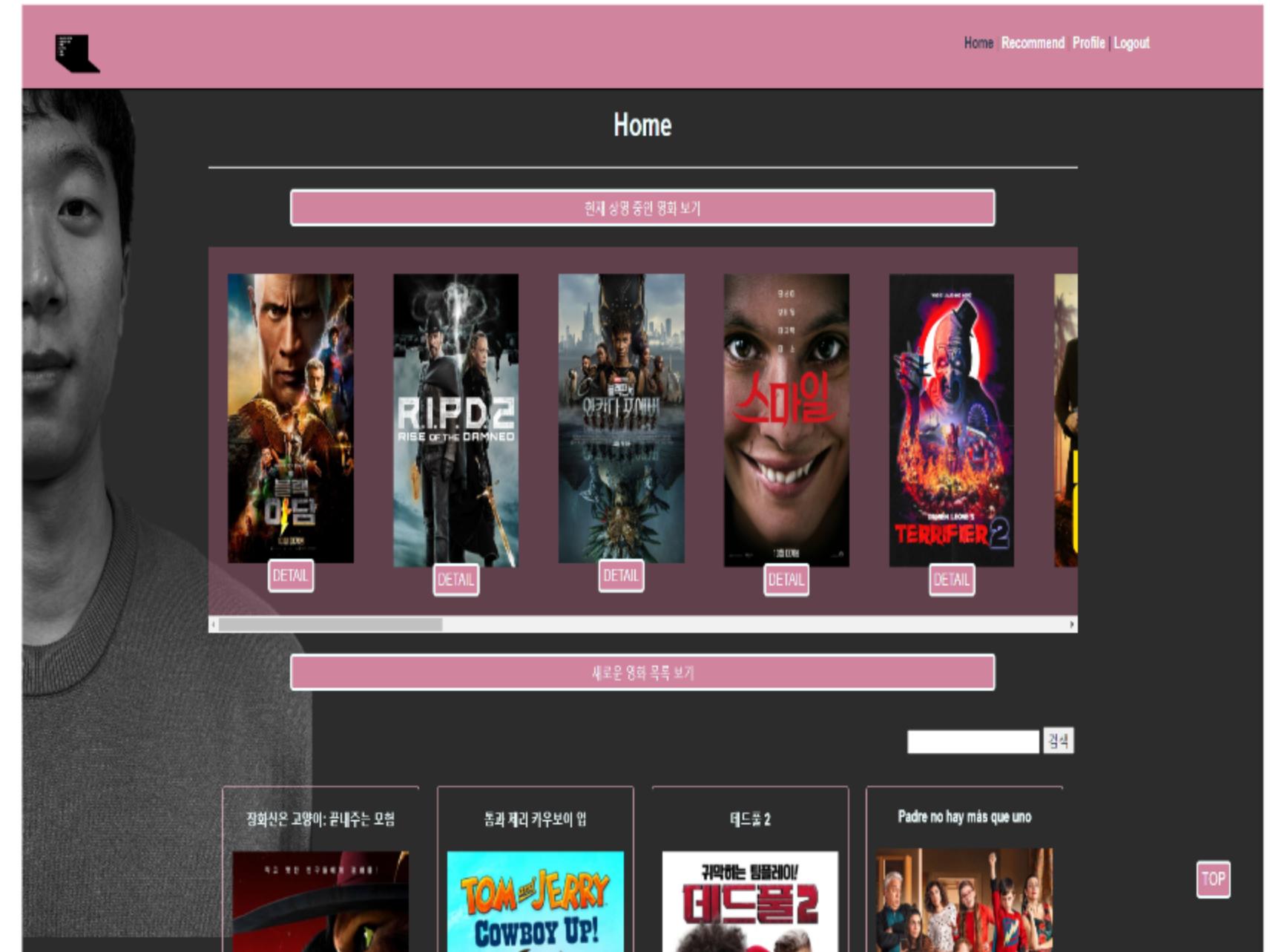
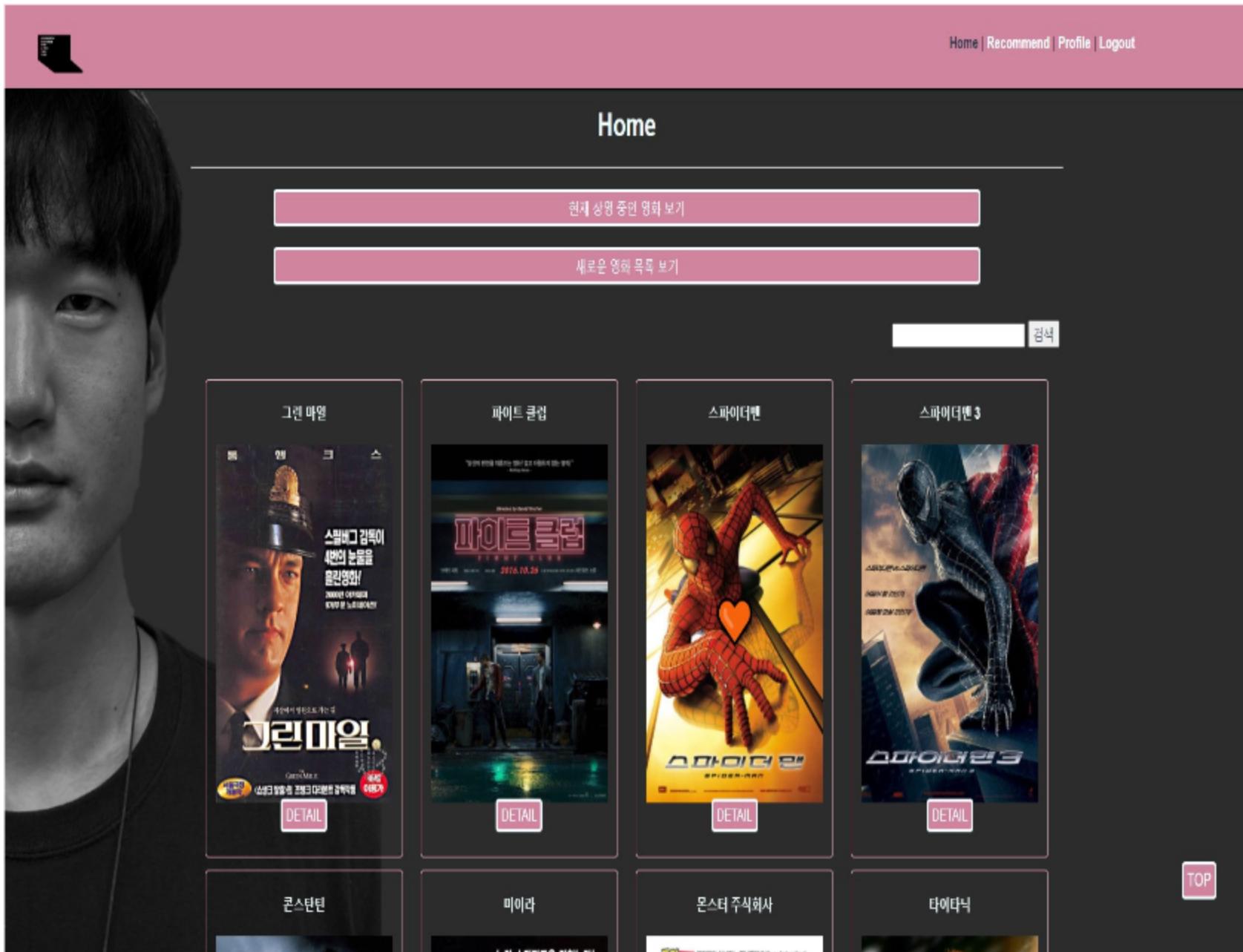
- 장고가 서로 어떻게 연결되어 있는지 ERD를 사용하여 소개
- 전체적인 페이지를 보면서 소개를 해줄 것임

※ 필터 기능은 가볍게 소개한 예전 ⇒ 별다른 알고리즘이 적용된 것은 아니기 때문에

## 대표서비스

- 사용자의 좋아요를 통한 영화 목록 관리
- 좋아요를 통한 알고리즘 기반 영화 추천

## 홈화면 구성



# 디테일화면 구성

Home | Recommend | Profile | Logout

Detail

팀버튼의 크리스마스 악몽

판타지 애니메이션 가족

1년에 한 번 홀로원 휴가를 위해 사는 할로윈 매출티켓이 깨운 해  
될 것입니다. 다른 티켓 대신 가격으로 홀로원을 즐기게 하는 깨운 할로  
윈 열정에 감동을 느낀다 어느 날 깨운 우연히 홀로원 티켓을 찾더니  
크리스마스 대출이 도착한다. 홀로원 대출자는 전자에게 다른 크리스  
마스 대출을 보고 깨운 할로윈 티켓에서 크리스마스 축제를 즐기고 한  
데 그의 일정은 전자 를 낭비하고 산타페인 헬렌(나지매, 별 등  
을 크리스마스 선물로 준비해 아이들에게 나누어주던, 크리스마스  
크 품은 기분은 사라지고 온갖 소동만 일어나는데..

▶상세보기 ▶등록하기

TOP

BOOK

Heart

Home | Recommend | Profile | Logout

자료로 영화 볼거리가

TOP

Home | Recommend | Profile | Logout

최근본 가봉은 사진과 같은 온갖 소동만 일어나는데..

비슷한 영화 보기하기 | 줄여진 보기

Heart

BOOK

TOP

THIS IS CONTENTLIST

작성일	작성자	내용	수정
2022-11-24	여진 무서운 느낌이었어요		
2022-11-24	미쳤다고 생각해요		
2022-11-24	정말 놀라웠어요		

TOP

# 추천화면 구성

Home | Recommend | Profile | Logout

독립 영화 off   고전 명작 on   좋아요 기반 다시 추천받기

Recommend

팀버튼의 크리스마스 악몽

캐시트릭

레전드 퀘스트: 오리진

Padre no hay más que uno

조제, 호랑이 그리고 물고기들

벼랑 위의 포뇨

일곱 개의 대죄: 빛에 저주받은 자들

천로역정: 천국을 찾아서

TOP

# 프로필 화면 구성

Home | Recommend | Profile | Logout



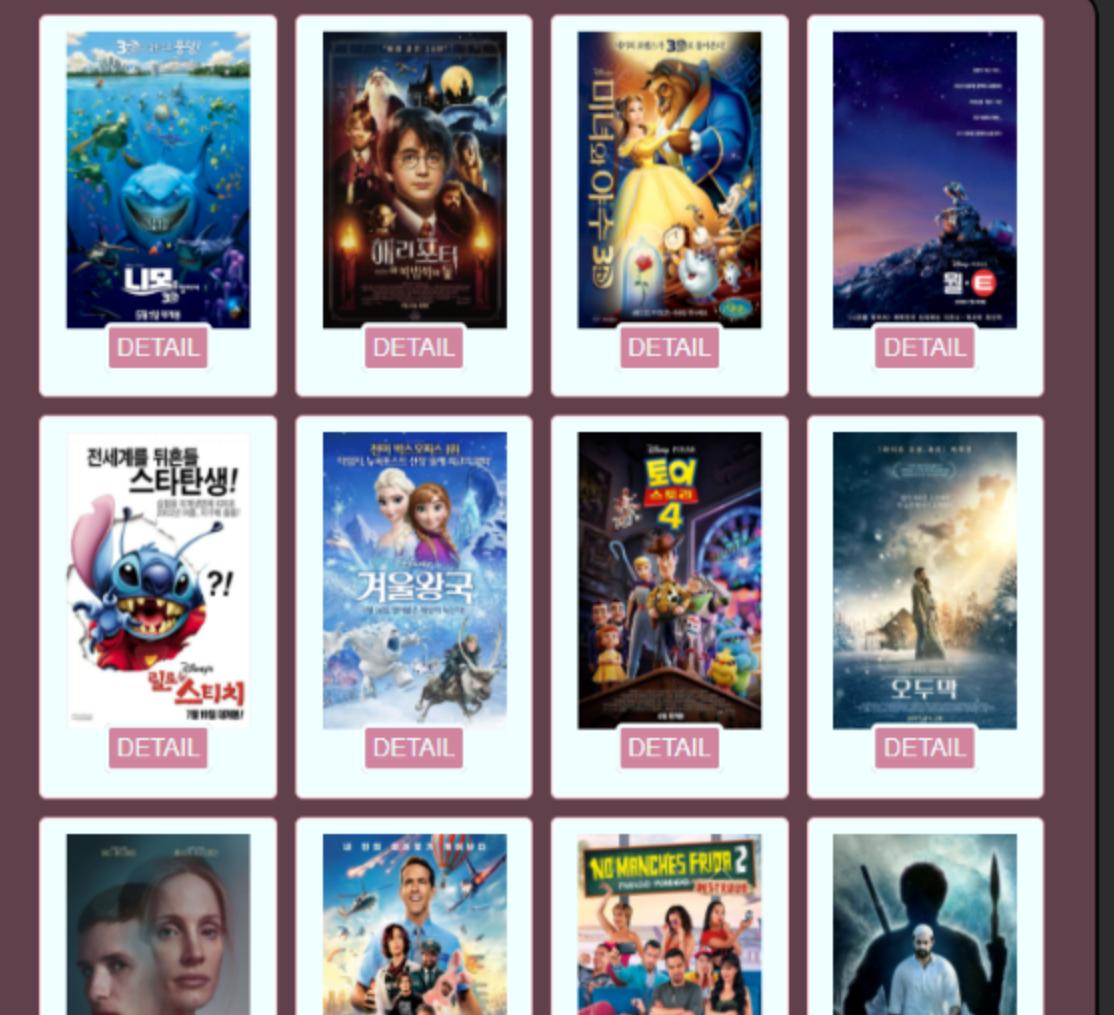
## final 님의 프로필

### 최근 본 영화 댓글

**팀버튼의 크리스마스 악몽**

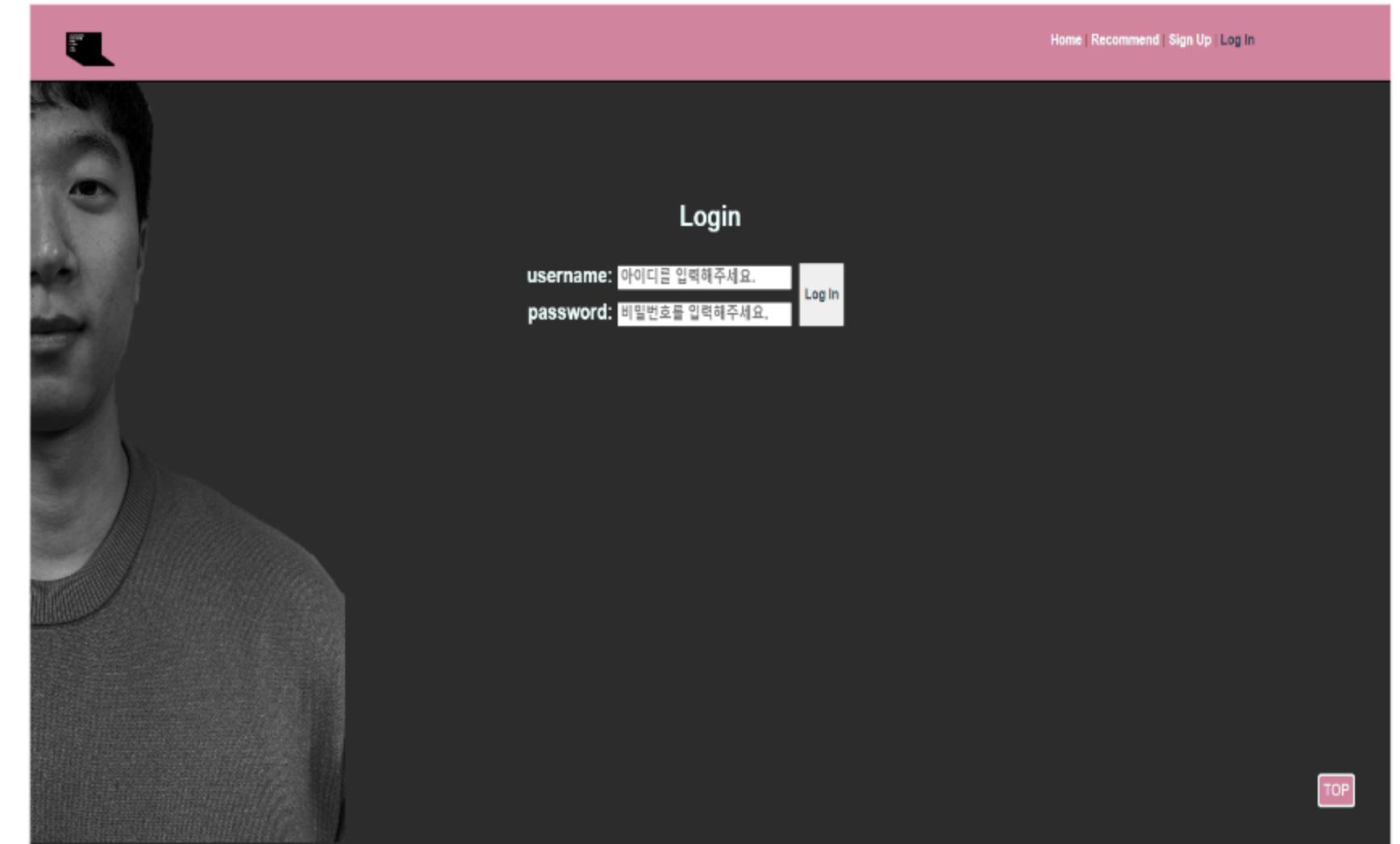
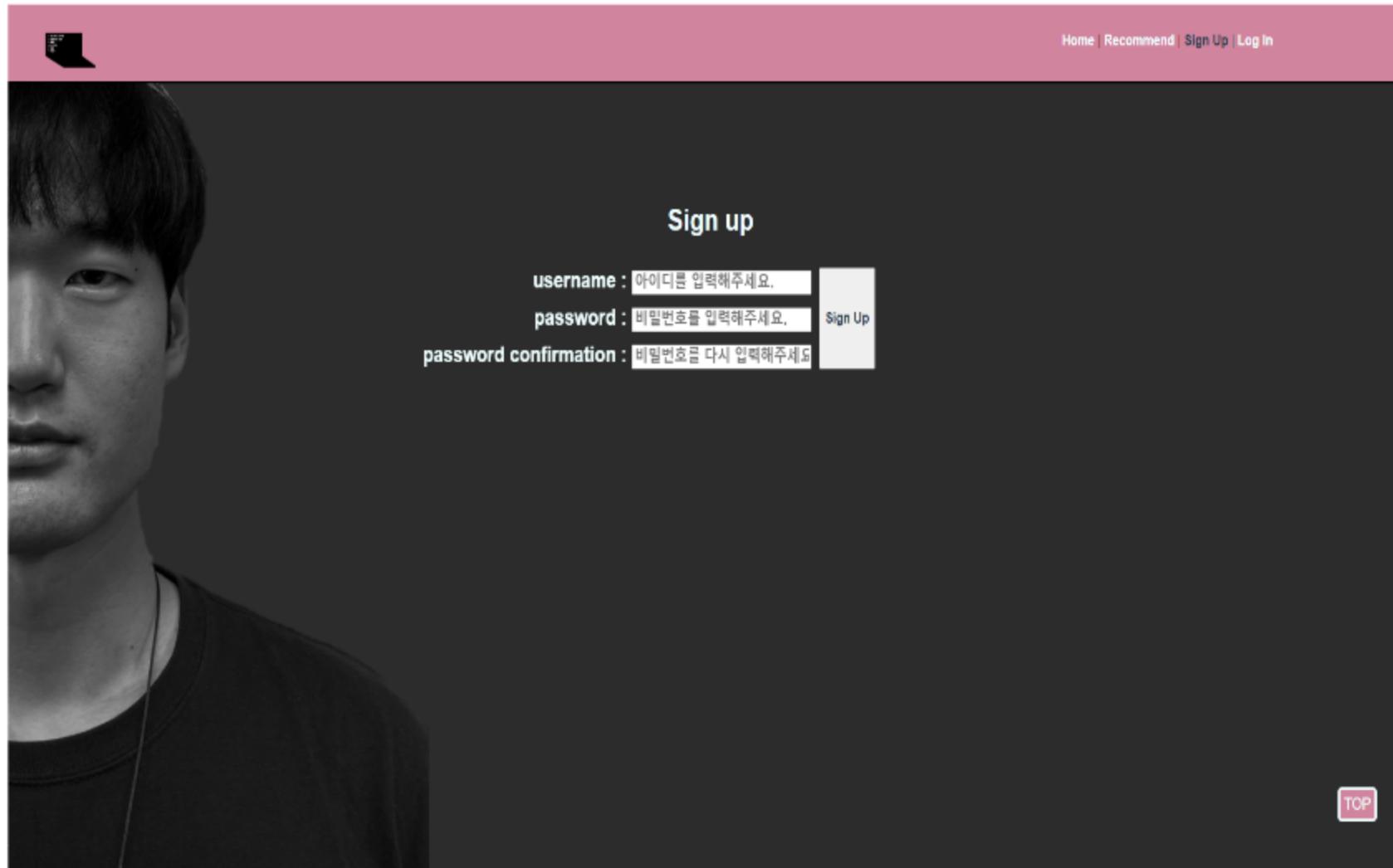
"약간 무서운 느낌이 있어요"	2022-11-24
"따뜻하고 재밌어요"	2022-11-24
"정말 놀라워요 😱"	2022-11-24

### 좋아요한 영화



TOP

# 회원가입 & 로그인 구성



# 알고리즘 구현 과정

④ 아이콘 추가 ⑤ 커버 추가

## 승규's 알고리즘

■ 날짜 1      비어 있음  
⑦ 상태      비어 있음  
+ 속성 추가

⑥ 댓글 추가

---

1. 고전명작 추천 알고리즘

- 개인적으로 고전작품을 좋아하는데 우리에게 있는 데이터에 개봉일이 있으므로 일정기간내, 혹은 일정 시간대 전의 작품중 평점이 높은 작품들을 추려 보여준다. ⇒ 최신영화 추천은 흔히 예전 작품을 추천하고자 하는 의도

2. 키워드 추천 알고리즘

- 우리가 가진 데이터에 오버뷰가 있으므로 추천화면에서 키워드 검색을 통해 오버뷰에 해당 키워드가 있는 작품들을 추려 추천해준다. ⇒ 영화제목만으로 검색하는거에는 한계가 있다고 본다. 오버뷰 설명이 전부 한글로 있는건 아니지만 데이터가 방대하면 충분히 추천 가능할 듯.

3. 숨겨진 땅작 추천 알고리즘

- 우리가 가진 데이터에 popularity 가 인기에 해당되어서 영화에 대한 관심을 나타낸다. 여기에 일정수준 이하의 인기를 가진 영화중 평점이 8점대 이상인 것을 보여준다. ⇒ 즉 잘 안알려졌지만, 본 사람들은 재밌게 본 영화 popularity 는 기준이 원지 잘 몰라서 votecount(투표 수)는 낮지만 voteaverage(평점) 이 높은 작품으로 대체 해도 될듯 물론 투표수도 최소 50표 정도는 넘어야 할듯.

④ 아이콘 추가 ⑤ 커버 추가

## 송섭's 알고리즘

■ 날짜 1      비어 있음  
⑦ 상태      비어 있음  
+ 속성 추가

⑥ 댓글 추가

---

1. 콘텐츠 기반 좋아요와 클릭의 비중

처음에는 인기순이나 투표가 많은 순으로 정렬  
(좋아요를 누르려면 일단 아는 영화여야 하기 때문)

그 다음에 좋아요(5)나 클릭시(1) 비중을 두어 그 영화에 (장르, 감독)을 기반으로 정렬

2. User-based Collaborative filtering

A의 좋아요 리스트가 B의 좋아요 리스트와 어느정도 일치하면 서로 포함되지 않는 좋아요 리스트를 추천해주기

<https://www.analyticsvidhya.com/blog/2020/11/create-your-own-movie-movie-recommendation-system/>

3. 리스트

좋아요와 싫어요를 도입해서 싫어요를 누르면 추천 리스트에서 감독과 장르를 일정부분 제외

스크럼회의를 참고한 방식

# 알고리즘 구현

```
bringLike() {
  this.algoList = [];
  axios({
    method: "get",
    url: `http://127.0.0.1:8000/accounts/liked/${this.username}/`,
  })
  .then((res) => {
    this.likeList = res.data.liked_movie;
    // likeList에 해당하는 각각의 장르들을 저장(like_genres)
    for (const movie of this.movies) {
      if (this.likeList.includes(movie.id)) {
        this.like_genres.push(...movie.genre_ids);
      }
    }
    // 여기서 포문을 돌려 전체 장르(movie_genres)에서 좋아하는 장르(likeList.movie.genre_ids)에 점수를 주고싶음.
    for (const like_genre of this.like_genres) {
      this.movie_genres[like_genre] += 5;
    }
  })
  // 그 점수를 기반으로 새로운 리스트 정렬한 기준 완성.
  let movieScore = -1;
  for (const movie of this.movies) {
    movieScore = 0;
    for (const genreId of movie.genre_ids) {
      movieScore += this.movie_genres[genreId];
    }
    movieScore = parseInt(movieScore / movie.genre_ids.length);
    this.algoList.push({
      movieScore: movieScore,
      id: movie.id,
      title: movie.title,
      poster_path: movie.poster_path,
      release_date: movie.release_date,
      vote_average: movie.vote_average,
      vote_count: movie.vote_count,
    });
  }
  this.algoList.sort((a, b) => b["movieScore"] - a["movieScore"]);
  this.algoList = _.sampleSize(this.algoList.slice(0, 300), 100);
})
.catch((err) => {
  console.log(err);
});
```

```
data() {
  return {
    username: localStorage.getItem("userName"),
    isClassic: false,
    isIndependent: false,
    like_genres: [],
    movie_genres: {
      28: 5, // 액션
      12: 5, // 모험
      16: 5, // 애니메이션
      35: 5, // 코미디
      80: 5, // 범죄
      99: 5, // 다큐멘터리
      18: 5, // 드라마
      10751: 5, // 가족
      14: 5, // 판타지
      36: 5, // 역사
      27: 5, // 공포
      10402: 5, // 음악
      9648: 5, // 미스터리
      10749: 5, // 로맨스
      878: 5, // SF
      10770: 5, // TV 영화
      53: 5, // 스릴러
      10752: 5, // 전쟁
      37: 5, // 서부
    },
    likeList: [],
    algoList: [],
  };
}
```

```
@api_view(["POST"])
def liked(request, movie_id):
  username = request.data["username"]
  movie = Movie.objects.get(pk=movie_id)
  user = User.objects.get(username=username)
  # 만약 username에서 해당 DB에 일치하는 movie_id가 없으면 추가, 있으면 제거
  if request.method == "POST":
    if user.liked_movie.filter(pk=movie_id).exists():
      user.liked_movie.remove(movie)
    else:
      user.liked_movie.add(movie)
  serializer = LikedSerializer(user)
  return Response(serializer.data)

@api_view(["GET"])
def liked_set(request, username):
  user = User.objects.get(username=username)
  serializer = LikedSerializer(user)
  return Response(serializer.data)
```

# 4

## Contents 4

피드백 & 마무리

S EUNGKYU  
S ONGSUB  
A ND  
M OVIE  
F OR  
Y OU

아쉬운 점

알고리즘

컴포넌트

만족한 점

계획

분위기

목표달성



**Q n A**

We are **SSAMFY**

**Thank you all :)**

감사합니다.

