

```
1  
2  
3  
4 // Welcome to the Coding Plan  
5 console.log("Developer's Programming Plan");  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17 // Ready to Start?  
18 console.log("Next Page");  
19  
20  
21
```

# {리뷰파인더; JSP\_해보조



# 기획안 목차

```
1  
2  
3 // Table of Contents  
4 const contents = [
```

CONTENTS 01  
**프로젝트 설명**

CONTENTS 02  
**웹사이트 시연**

CONTENTS 03  
**코드 설명**

CONTENTS 04  
**팀원 소개**

CONTENTS 05  
**소감**

CONTENTS 06  
**질의 응답**

# 프로젝트 설명

1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21

<!-- 프로젝트명 -->

<!-- 개발일정 -->

<!-- 주제 설명 -->

<!-- 선정 이유 -->

- 리뷰 파인더

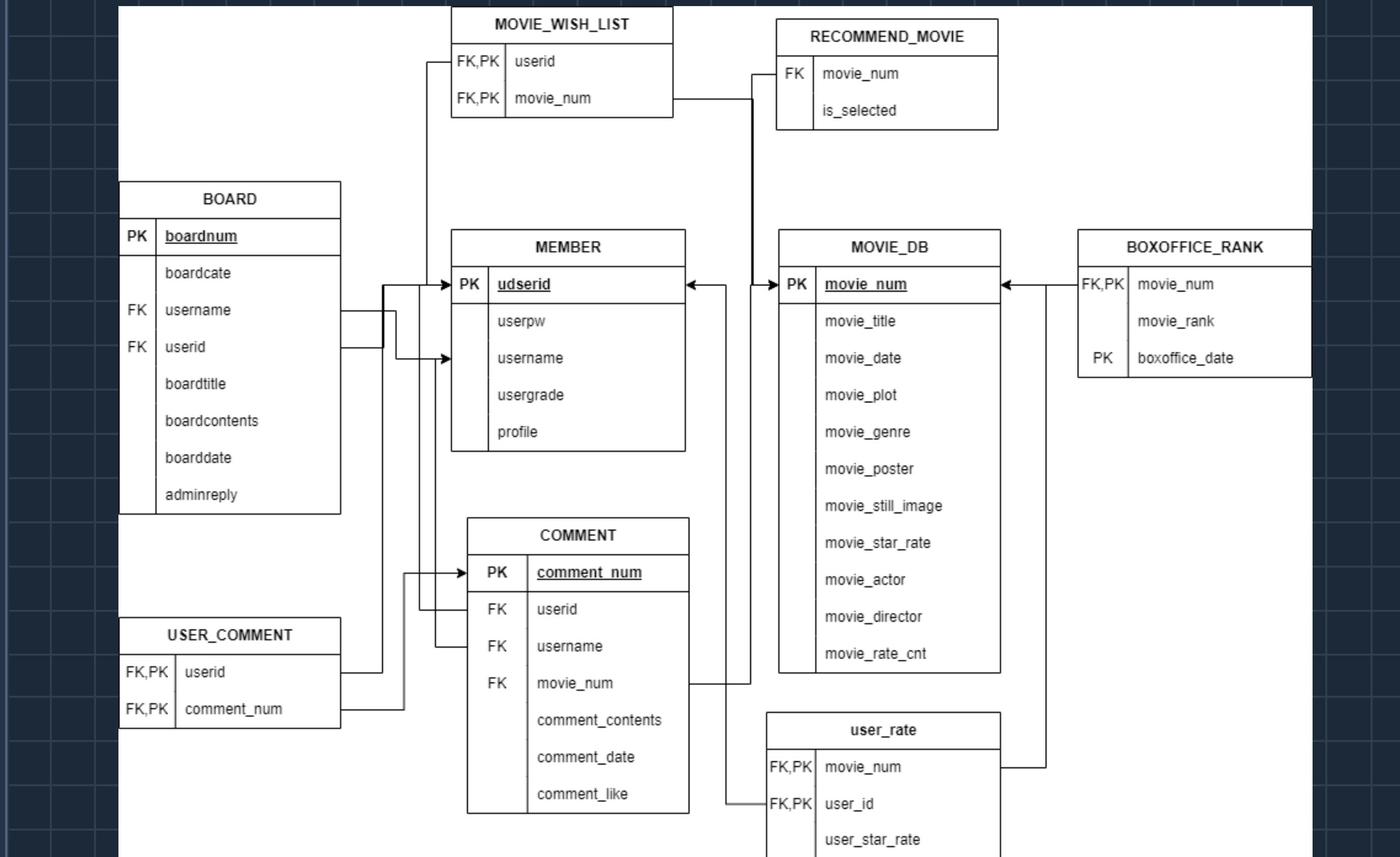
• 2024.09.09 ~ 2024.10.11

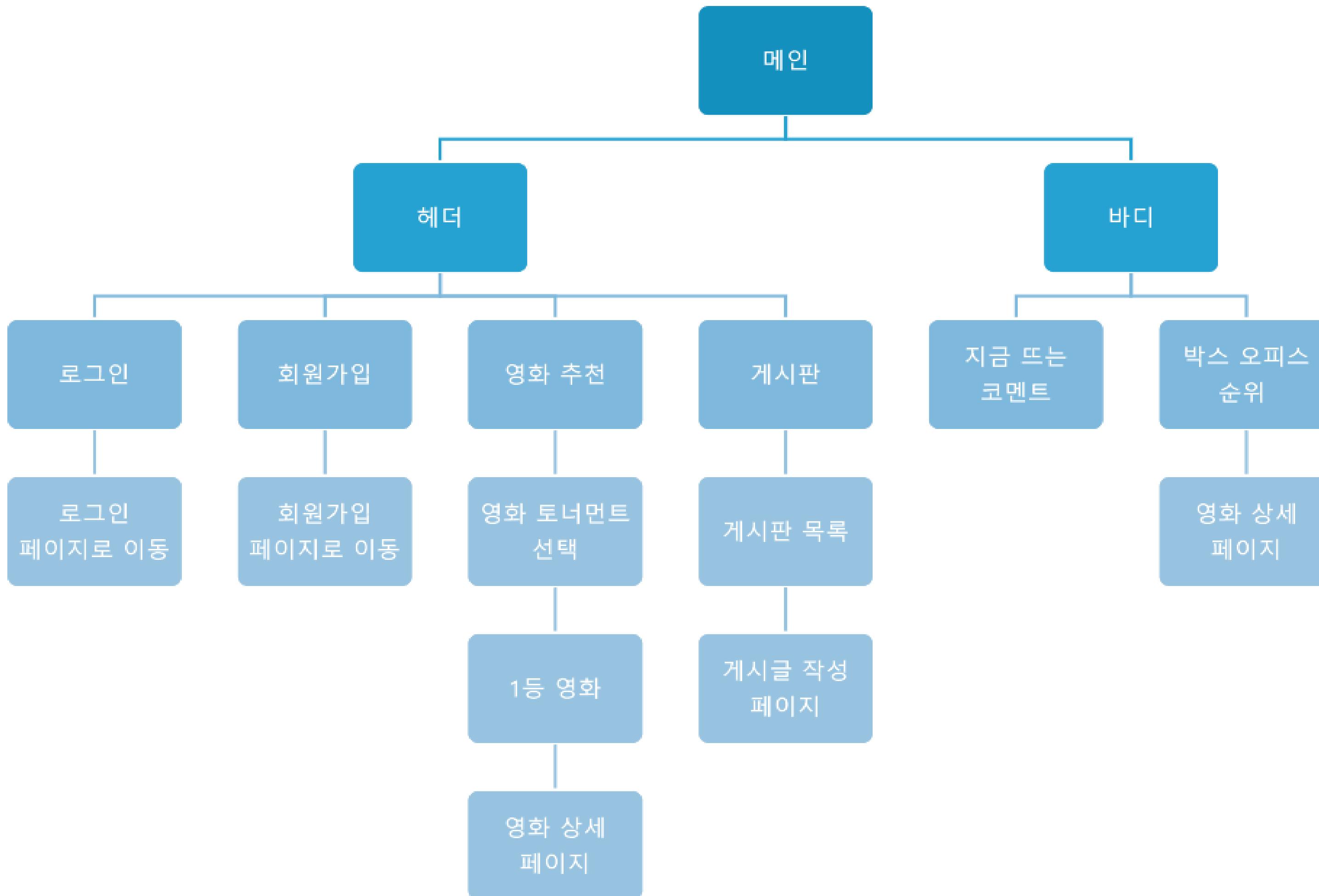
• 컨텐츠에 대한 리뷰 및 별점을 제공하는 플랫폼으로 사용자들이  
직접 작성한 리뷰를 읽고 자신의 의견을 남길 수 있으며 좋아하는  
작품을 리스트로 만들어 관리 할 수 있음

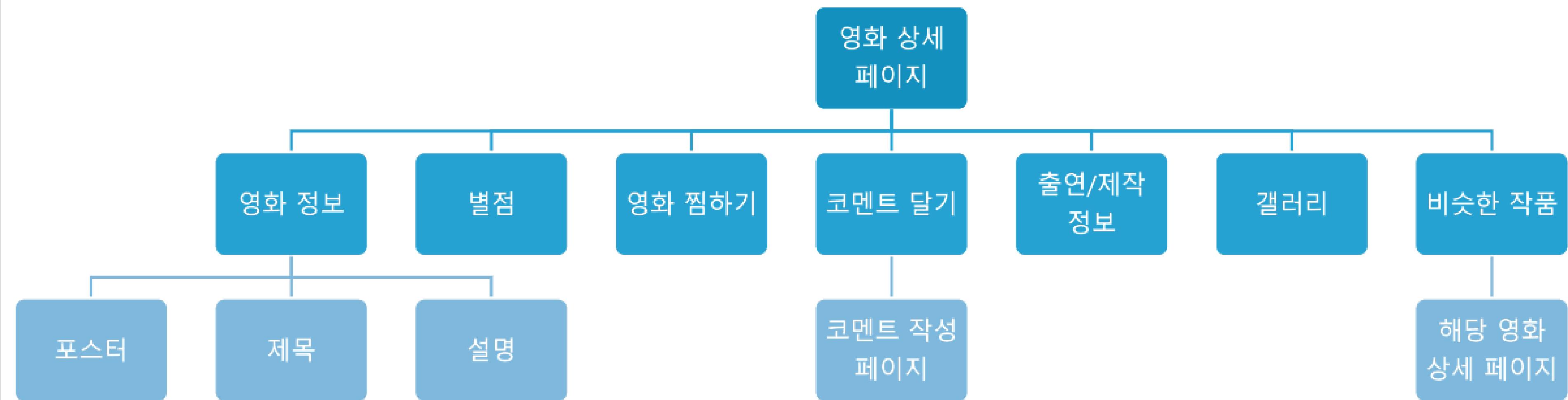
• 수업에서 배운 기능을 최대한 활용하여 흥미로운 작업물을 완성.  
• db를 채우고, 실제로 웹사이트에서 사용하는 기능들을 추가하기  
에 적합한 플랫폼이라고 생각.  
• 로드맵이 있어 프로젝트의 전반적인 방향성을 유지하면서 배운 내용을 실제로 적용해볼 수 있는 기회를 가질 수 있었음

프로젝트 이름	JSP 프로젝트	사이트 이름	리뷰 파인더 (Review Finder)	화면 경로	일정
SUN	MON	TUE	WED	THU	FRI
22	23	24	25	26	27
※ 한서진님 박성우님 신윤창님 김영범님 임지수님	※ 전체 F : Front B : Back-end			F : 메인 헤더 + 푸터 F : 마이페이지 F : 영화 페이지 F : 로그인+회원가입	B : 회원 정보 수정 + 탈퇴 F : 배우 페이지 F : 게시판 F : 영화 추천
29 보충	30	10/1	2	3	4 휴강
F : 메인 3단		B : 지금 또는 코멘트 B : 별점 평가한 영화 B : 별점 B : 로그인+회원가입	B : 찜하기	B : 박스 오피스 순위 B : 좋아요한 코멘트 B : 코멘트(+좋아요)	B : 추천 영화 B : 좋아요한 영화 B : 작품 활동 B : 게시판 B : 영화 추천
6 보충	7	8	9 휴강	10	11 발표일
			페이지 합치기	PPT 제작 발표 준비	

# DB 데이터그램







마이  
페이지

회원 정보  
설정

좋아요한  
영화

비밀번호  
변경

닉네임  
변경

문의하기

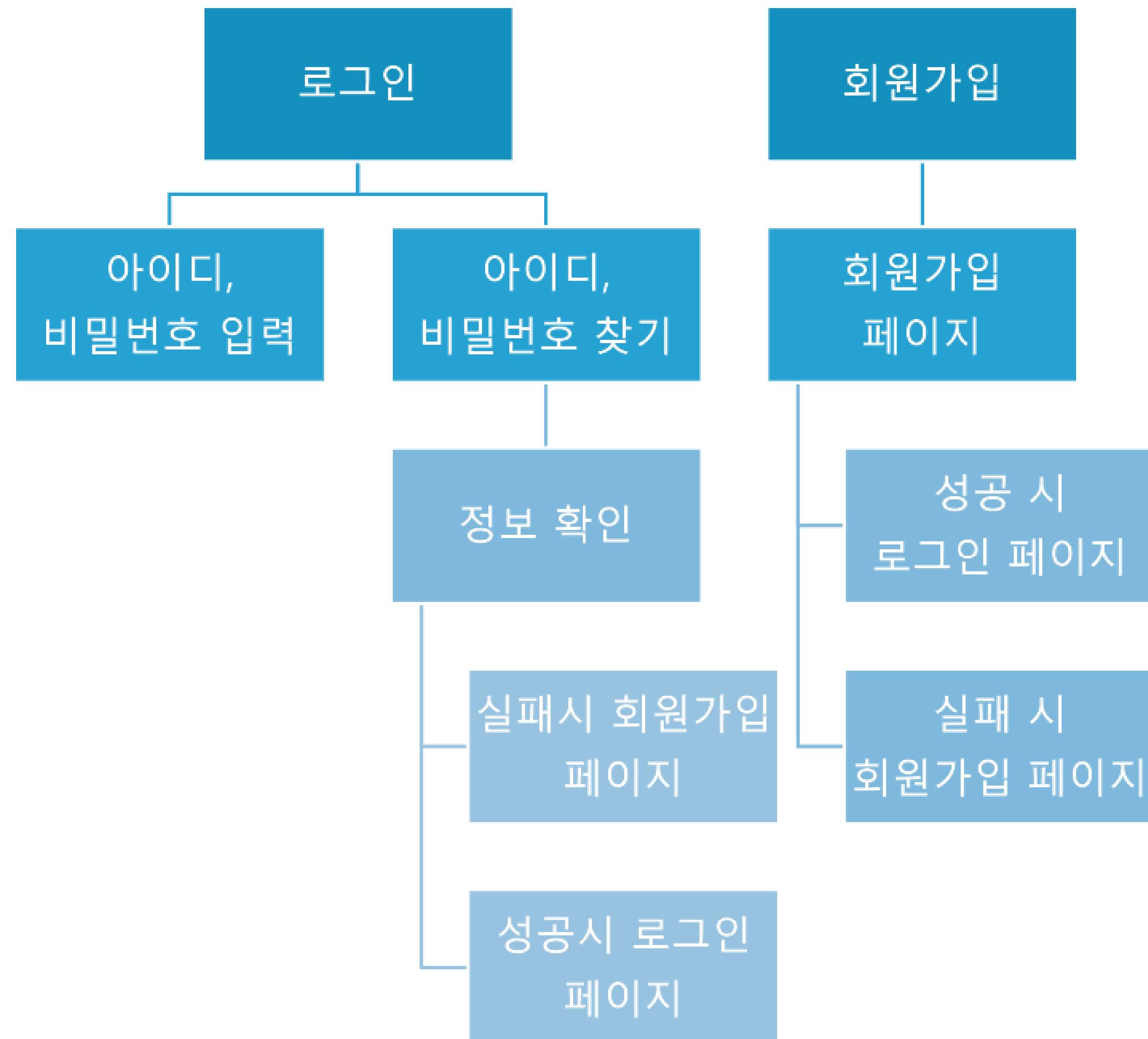
로그아웃

탈퇴

영화 상세  
페이지

게시판으로  
이동

메인  
페이지



**1-1** WATCHA PEDIA

**1-2** 영화 추천

**1-3** 콘텐츠, 인물, 컬렉션, 유저를 검색합니다.

**1-4** 로그인 회원가입

**2** 지금 뜨는 코멘트

**2-1** 도망쳐야되며너랑한약속도망쳐야돼

**2-2** 김현이 어글리 소재랑 세계관 CG는 나쁘지 않다.

**2-3** 더보기 >

**2-4** 박스오피스 순위

**2-5** 김해인 흥을 살피는 김해인

**2-6** 조커: 폴리 아 되 2024 · 미국 예매율 7.7%

**2-7** 트랜스포머 ONE 2024 · 미국 예매율 6.2% · 누적 관객 2.4만명

**2-8** 대도시의 사랑법 2024 · 한국 예매율 4.5%

**2-9** 비긴 어게인 2013 · 미국 예매율 3.6% · 누적 관객 351.1만명

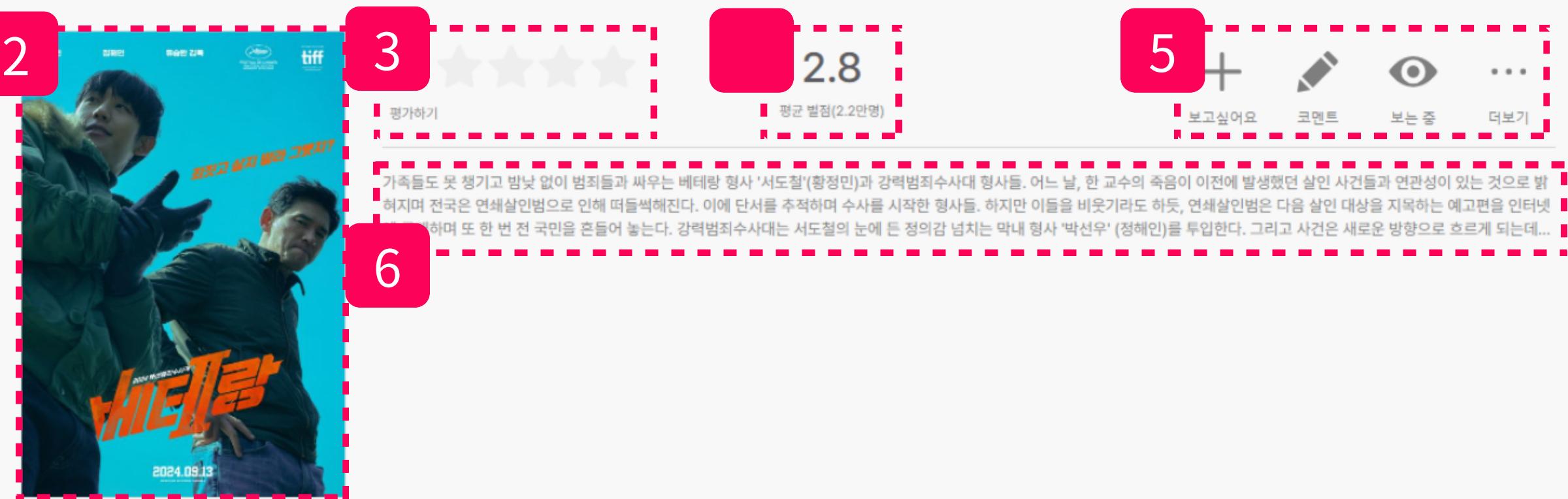
**3-1** 지금까지 ★724,618,159 개의 평가가 쌓였습니다.

**3-2** 서비스 이용약관 | 개인정보 처리방침 | 회사 안내  
고객센터 | cs@watchapedia.co.kr, 02-515-9985  
제휴 및 대외 협력 | https://watcha.team/contact  
주식회사 왓챠 | 대표 박태훈 | 서울특별시 서초구 강남대로 343 신द빌딩 10층  
사업자 등록 번호 211-88-66013  
WATCHAPEDIA © 2024 by WATCHA, Inc. All rights reserved.

## 설명

## 메인 화면

1	Header
1-1	로고 (클릭 시 메인 화면으로 이동)
1-2	영화 추천
1-3	검색창 (영화, 배우 검색)
1-4	로그인, 회원가입
2	Container (지금 뜨는 코멘트, 박스오피스 순위, 추천 영화)
2-1	각 컨테이너 이미지 (좌우 롤링)
2-2	전체 코멘트 페이지로 이동
2-3	해당 영화 페이지로 이동
2-4	Footer
3	지금까지 리뷰 개수 표시
3-1	서비스 이용약관, 개인정보 처리방침, 회사 안내



별점 그래프

평균 ★ 2.8 (2.2만명)

## 설명

## 영화 상세 페이지\_1

- |   |            |
|---|------------|
| 1 | 영화 스타일 이미지 |
| 2 | 영화 포스터     |
| 3 | 별점 매기기     |
| 4 | 평균 평점      |
| 5 | 찜하기 / 코멘트  |
| 6 | 영화 설명      |

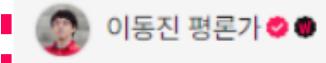
## 출연/제작

1

류승완  
감독오달수  
조연 | 오 팀장김시후  
조연 | 윤 형사정만식  
조연 | 전석우황정민  
주연 | 서도철오대환  
조연 | 황 형사진경  
조연 | 이주연권해효  
조연 | 강수대 총경정해인  
주연 | 박선우장윤주  
조연 | 봉 형사신승환  
조연 | 정의 부장변홍준  
조연 | 서우진

## 코멘트 3500+

2

이동진 평론가  
당혹스런 오프닝과 엔딩을 위한 엔딩 그리고 그 사이의 종종 가웃거려지는 장면들.손정빈 기자  
류승완의 용서받지 못한 자 영화 '베테랑2'는 반성문 같다. '베테랑'(2015)은 개봉 당시 한 단아로 요약됐다. 사이다. 평범한 형사 서도철이 재벌 3세 조태오를 말 그대로 때려 잡는 모습을 본 관객은 그 통쾌함에 열광했다(1341만명). 그런데 '베테랑'...

641 좋아요

526 좋아요

좋아요

좋아요



베테랑말고 베를린2 좀 내놔봐요 제발...나 아직 베를린에 갈혀 살아...



STONE 정의를 쾌락으로 왜곡하는 소비 체계에 대한 반성. 정의구현의 윤리를 다루는 (무식할 정도로 강력한) 담론

3



함성규 황정민은 내가 꽤 좋아하는 배우이며, 정해인은 내가 대놓고 좋아하는 배우라서 객관적인 후기를 남기는 것은 좀 어려운 일이지만.. 감독 이름값이 있기에 작품에 대한 기대치는 원래 없었지만, 초반 오프닝부터 일말의 기대치도 버리고 시작했다. 수요일...

390 좋아요

365 좋아요

3 좋아요



새침

- 서도철이 자신의 신념을 선택으로 보여줘야하는 순간(후반부 딜리마파트) 선택을 보여주지 않고 데우스 엑스 마키나적 상황으로 유야무야 넘어간 것.
- 박선우가 왜 싸이코패스인지 캐릭터를 구성한 본인들도 몰라서 아무런 이유 없이 그냥 눈...

4

더보기

## 설명

## 영화 상세 페이지\_2

1

배우 / 감독 상세 페이지로 이동

2

클릭한 댓글 전체 보기

3

댓글 좋아요

4

댓글 전체 더보기

## 갤러리

1



## 비슷한 작품

2



베테랑

평균 ★ 4.0

영화



노웨이아웃: 더 룰렛

평균 ★ 3.0

시리즈



비질란테

평균 ★ 3.2

시리즈



범죄도시4

평균 ★ 2.9

시리즈



범죄도시14

평균 ★ 2.9

영화



형사록 시즌 1

평균 ★ 3.2

시리즈



검사외전

2016.02.03



무도실무관

ONLY ON NETFLIX | 9월 13일 공개



나쁜녀석들: 더 무비

## 설명

## 영화 상세 페이지\_3

1

갤러리 (스틸 이미지)

2

클릭 시 영화 페이지로 이동



## 강아지 월드컵

귀요미들

총 라운드를 선택하세요.

32강

총 32명의 후보 중 무작위 32명이 대결합니다.

1

2

시작하기

설명

영화 추천\_1

1 32강, 16강, 8강, 4강 중 선택 가능

2 선택한 라운드로 페이지 이동

## 강아지 월드컵 8강 1/4



설명

영화 추천\_2

1, 2

다음 페이지로 넘어감



1

2

C 다시시작     ≡ 랭킹보기

https://www.\*\*\*.com/ (주소복사)

사용자 의견 (49323)

닉네임  
익명  
한마디 남기기  
메세지를 입력하세요.

유리 - (시츄) 2024-09-19 16:24:05 신고  
넘 귀엽지 않아요

익명 - (요크셔테리어) 2024-09-19 09:57:49 신고  
강아지좋아라고숯냥이가말한다

익명 - (진돗개) 2024-09-18 22:52:20 신고  
아니 국뽕이고 뭐고 (국뽕심은 있음)저는 그냥

허슈킹 - (시베리안 허스키) 2024-09-18 11:12:00 신고  
역쉬 시베리안 허스키가 1등이라구!!!!

익명 - (시츄) 2024-09-16 18:38:23 신고  
시츄 너무 귀여운데??

익명 - (포메라니안) 2024-09-15 18:10:56 신고  
포메라니아

정시아 - (시츄) 2024-09-15 16:45:16 신고  
말티즌 너무 기엽다

김연아 스케이트선수 - (시베리안 허스키) 2024-09-15 16:45:16 신고  
시베리안 허스키 정말 귀엽지 않나요 근데 시츄

설명

영화 추천 결과 화면

1등으로 뽑은 영화 보여주기

맨 처음 추천 화면으로 이동

**Q&A**

**1** 구매하시려는 상품에 대해 궁금한 점이 있으신 경우 문의해주세요. 판매자 톡톡문의를 통해 판매자와 간편하게 1:1 상담도 가능합니다.

**2** 상품 Q&A 작성하기      나의 Q&A 조회 >

**3** 비밀글 제외      내 Q&A 보기      답변상태 OFF

**4** 답변상태      제목      작성자      작성일

답변완료	비밀글입니다. 🔒	ga***	2024.07.01.
답변완료	[Redacted]	jins*****	2024.05.27.
답변완료	[Redacted]	clfn****	2024.05.16.
답변완료	비밀글입니다. 🔒	shn5***	2024.05.08.
답변완료	비밀글입니다. 🔒	kang*****	2024.04.24.
답변완료	비밀글입니다. 🔒	sill*****	2024.04.23.
답변완료	비밀글입니다. 🔒	wh***	2024.04.22.

설명	
게시판	
1	카테고리 구분(이용 문의 / 영화 추가 요청)
2	글 작성 / 수정 / 댓글
3	내가 쓴 글 보기
4	답변 상태, 제목, 작성자, 작성일

**1** 상담분류

**2** 제목   
0/100

**3** 내용   
0/5,000

문의 시 주민번호, 계좌번호와 같은 개인정보 입력은 자양하여 주시기 바랍니다.  
 쇼핑&페이 고객센터는 산업안전보건법을 준수하여 고객 응대근로자를 보호하고 있습니다.  
 성희롱, 욕설 등의 폭언을 하지 말아주세요, 폭언 시 상담이 제한되고 법령에 따라 조치될 수 있습니다.

## 이메일

## 답변알림

[답변이 완료되면 네이버 앱 알림으로 알려드립니다.](#)

## 파일첨부

## 파일첨부

10MB 이하의 파일 7개까지 첨부하실 수 있으며(최대 50MB) exe, zip 등 일부 파일형식은 첨부하실 수 없습니다.

## 개인정보 수집 및 동의 안내

수집하는 개인정보 항목 : 이메일 주소

작성해주시는 개인 정보는 문의 접수 및 고객 불만 해결을 위해 **3년간 보관됩니다.**

이용자는 본 동의를 거부할 수 있으나, 미동의 시 문의 접수가 불가능합니다.

 동의합니다.**4**

문의하기

**5**

취소



## 설명

## 게시글 작성 페이지

- |   |                            |
|---|----------------------------|
| 1 | 카테고리 구분 (이용 문의 / 영화 추가 요청) |
| 2 | 제목 작성                      |
| 3 | 내용 작성                      |
| 4 | 게시판 페이지로 이동                |
| 5 | 새로운 게시판 작성 페이지로 이동         |

## 설명

## 로그인 페이지

1	아이디 / 비밀번호 입력
2	성공 시 메인 화면으로 이동 실패 시 로그인 화면으로 이동
3	비밀번호 찾기 페이지로 이동
4	회원가입 페이지로 동

1 이메일

비밀번호

2 로그인

3 비밀번호를 잊어버리셨나요?

4 계정이 없으신가요? 회원가입

# WATCHA PEDIA

## 회원가입

1

| 이름

이메일

비밀번호

한국어(대한민국) ▼

2

회원가입

3

이미 가입하셨나요? [로그인](#)

**WATCHA PEDIA**

영화

🔍 콘텐츠, 인물, 컬렉션, 유저를 검색해보...

1

**바그다드**

tim0708@naver.com

팔로워 0 | 팔로잉 1

3

0  
평가

4

0  
코멘트

5

0  
컬렉션**보관함**

6



영화

2

**설명**

마이페이지\_1

- |   |                     |
|---|---------------------|
| 1 | 마이페이지로 이동           |
| 2 | 회원 정보 수정 폐동         |
| 3 | 별점으로 평가한 영화 목록으로 이동 |
| 4 | 코멘트 페이지로 이동         |
| 5 | 좋아요 영화 페이지로 이동      |
| 6 | 시청 기록               |

좋아요

좋아한 컬렉션 0

좋아한 코멘트 0

컬렉션



영화

평가 0



아직 평가하신 작품이 없어요.

1

2

3

4

5

더보기

6

7

## 설명

## 마이페이지\_2

좋아요 누른 컬렉션으로 이동

좋아요 누른 코멘트 목록으로 이동

영화 포스터 클릭 후 영화 정보 페이지로 이동

시청 목록이 없을 경우

평가한 작품목록 페이지로 이동

좋아요한 영화 목록

시청 기록

시연

// Project Features

FEATURE 01

{ 시연 }

# 코드 설명

```
1 // apicontroller
2
3 public class APIController {
4     // 기본URL + 요청 인자로 api 호출
5     public String callAPI(HashMap<String, String> url_key ,HashMap<String, String> var_data) throws IOException {
6         StringBuilder urlBuilder = new StringBuilder(url_key.get("url"));
7         urlBuilder.append("?" + URLEncoder.encode(url_key.get("keytype"), "UTF-8") + "=" + url_key.get("key"));
8
9         for(int i=1;i<=var_data.size()/2;i++) {
10             urlBuilder.append("&" + URLEncoder.encode(var_data.get("var"+i), "UTF-8") + "=" + URLEncoder.encode(var_data.get("data"+i),"UTF-8"));
11         }
12         URL url = new URL(urlBuilder.toString());
13         if(url_key.get("url").equals("https://kobis.or.kr/kobisopenapi/webservice/rest/boxoffice/searchDailyBoxOfficeList.json")) {
14             System.out.println(urlBuilder);
15         }
16
17         HttpURLConnection conn = (HttpURLConnection) url.openConnection();
18         conn.setRequestMethod("GET");
19         conn.setRequestProperty("Content-type", "application/json");
20         // 상태 코드 출력
21         //System.out.println("Response code: " + conn.getResponseCode());
22         if(conn.getResponseCode()!=200) {
23             return callAPI(url_key, var_data);
24         }
25         BufferedReader rd;
26         if (conn.getResponseCode() >= 200 && conn.getResponseCode() <= 300) {
27             rd = new BufferedReader(new InputStreamReader(conn.getInputStream()));
28         } else {
29             rd = new BufferedReader(new InputStreamReader(conn.getErrorStream()));
30         }
31         StringBuilder sb = new StringBuilder();
32         String line;
33         while ((line = rd.readLine()) != null) {
34             sb.append(line);
35         }
36         rd.close();
37         conn.disconnect();
38         return sb.toString();
39     }
40 }
```

# 코드 설명

```
1 // KobisJson  
2  
15 // 이름과 상영날짜로 영화 검색을 최초로 실행되는 클래스 최종적으로 movie를 return 해줄 클래스 [ex) 박스오피스 순위 별, {이름, 상영날짜} 건네 받고]  
16 public class KobisJson extends APIController{  
17     String keytype = "key";  
18     String key = "b841d040f088509344cac5574340c4cd";  
19     HashMap<String, String> url_key = new HashMap<String, String>();  
20     HashMap<String, String> var_data = new HashMap<String, String>();  
21  
22     public KobisJson() {  
23         url_key.put("keytype", keytype);  
24         url_key.put("key", key);  
25     }  
26  
27     // 제목으로 검색  
28     public List<MovieDTO> searchToMovieName(String movieNm) throws IOException, ParseException {  
29         List<MovieDTO> movieList = new ArrayList<MovieDTO>();  
30         MovieDTO moviedto = null;  
31         String url = "http://www.kobis.or.kr/kobisopenapi/webservice/rest/movie/searchMovieList.json";  
32         url_key.put("url", url);  
33  
34         String var = "movieNm";  
35         String data = movieNm;  
36         var_data.put("var1", var);  
37         var_data.put("data1", data);  
38  
39         String movieListData = callAPI(url_key, var_data);  
40  
41         JSONObject objData = (JSONObject) new JSONParser().parse(movieListData);  
42  
43         JSONObject movieListResult = (JSONObject) objData.get("movieListResult");  
44         JSONArray movieListArr = (JSONArray) movieListResult.get("movieList");  
45         if(movieListArr.size()==0) { // 영화 리스트가 비어있는지 체크  
46             return null;  
47         }
```

# 코드 설명

```
1 // KobisJson
2
3     for(Object obj : movieListArr) {
4         JSONObject movie = (JSONObject)obj;
5         moviedto = new MovieDTO();
6
7         String movieName = (String)movie.get("movieNm");
8         String movieDate = (String)movie.get("openDt");
9
10        if(movieDate.equals("")) {
11            continue;
12        }
13
14        System.out.println(moviedto.getMovie_title());
15        System.out.println(moviedto.getMovie_date());
16
17        moviedto = new KoreafilmJson().getMovieInfo(moviedto);
18
19        movieList.add(moviedto);
20
21    }
22
23    return movieList;
24 }
```

# 코드 설명

```
1 // KobisJson
2
3     // 영화 추천에 쓸 영화 박스오피스 기간별로 총 50개 받아오기
4     public List<MovieDTO> getRecommendList() throws IOException, ParseException{
5         List<MovieDTO> movieList = new ArrayList<MovieDTO>();
6
7         for(int i=0;i<10;i++) {
8             HashMap<String, MovieDTO> movieMap = null;
9             String date = null;
10            switch(i) {
11                case 0:
12                    date = "20230101";
13                    url_key.put("key", "6974fb00d854190ff2f7728d4a150966");
14                    break;
15                case 1:
16                    date = "20230501";
17                    url_key.put("key", "b841d040f088509344cac5574340c4cd");
18                    break;
19                case 2:
20                    date = "20230701";
21                    url_key.put("key", "6974fb00d854190ff2f7728d4a150966");
22                    break;
23                case 3:
24                    date = "20240201";
25                    url_key.put("key", "b841d040f088509344cac5574340c4cd");
26                    break;
27                case 4:
28                    date = "20240501";
29                    url_key.put("key", "6974fb00d854190ff2f7728d4a150966");
30                    break;
31            }
32            movieMap = getBoxOffice(date);
33            for(int j=1;j<=10;j++) {
34                movieList.add(movieMap.get(""+j+""));
35            }
36        }
37
38        return movieList;
39    }
```

# 코드 설명

```
// KobisJson

139     // 박스오피스 1~10위 가져오기
140     public HashMap<String, MovieDTO> getBoxOffice(String date) throws IOException, ParseException {
141         HashMap<String, MovieDTO> boxOfficeList = new HashMap<>();
142         String url = "https://kobis.or.kr/kobisopenapi/webservice/rest/boxoffice/searchDailyBoxOfficeList.json";
143         url_key.put("url", url);
144
145         String var = "targetDt";
146         String data = "20240923";
147         if(date != null) {
148             data = date;
149         }
150         var_data.put("var1", var);
151         var_data.put("data1", data);
152
153         String boxOfficeData = callAPI(url_key, var_data);
154
155         MovieDTO movie = null;
156
157         JSONObject objData = (JSONObject)new JSONParser().parse(boxOfficeData);
158         JSONObject boxOfficeResult = (JSONObject)objData.get("boxOfficeResult");
159
160         JSONArray dailyBoxOfficeList = null;
161
162         try{
163             dailyBoxOfficeList = (JSONArray)boxOfficeResult.get("dailyBoxOfficeList");
164         }catch (NullPointerException e) {
165             getBoxOffice(date);
166         }
167
168         for(Object obj : dailyBoxOfficeList) {
169             JSONObject movieData = (JSONObject)obj;
170             movie = new MovieDTO();
```

# 코드 설명

```
1 // KobisJson
2
3     for(Object obj : dailyBoxOfficeList) {
4         JSONObject movieData = (JSONObject)obj;
5         movie = new MovieDTO();
6
7         movie.setMovie_title((String)movieData.get("movieNm"));
8         movie.setMovie_date((String)movieData.get("openDt"));
9
10        // 영화 상세정보 전부 받아오기
11        movie = new KoreafilmJson().getMovieInfo(movie);
12
13        boxOfficeList.put((String)movieData.get("rank"), movie);
14    }
15    return boxOfficeList;
16 }
17 // 오버로딩
18 public HashMap<String, MovieDTO> getBoxOffice() throws IOException, ParseException{
19     return getBoxOffice(null);
20 }
21
```

# 코드 설명

```
1 // KoreafilmJson
2
3 // 영화 이름, 상영날짜를 넘겨받아서 장르, 줄거리, 포스터, 스틸이미지 가져올 클래스
4 public class KoreafilmJson extends APIController{
5     String url = "http://api.koreafilm.or.kr/openapi-data2/wisenut/search_api/search_json2.jsp";
6     String keytype = "ServiceKey";
7     String key = "066G060N254R5Q89BD66";
8     HashMap<String, String> url_key = new HashMap<String, String>();
9     HashMap<String, String> var_data = new HashMap<String, String>();
10
11     public KoreafilmJson(){
12         url_key.put("url", url);
13         url_key.put("keytype", keytype);
14         url_key.put("key", key);
15         var_data.put("var1", "collection");
16         var_data.put("data1", "kmdb_new2");
17     }
18
19     // 장르별 영화 검색
20     public List<MovieDTO> searchToGenre(String genre) throws IOException, ParseException{
21         List<MovieDTO> movieList = new ArrayList<MovieDTO>();
22         MovieDTO moviedto = null;
23
24         var_data.put("var2", "genre");
25         var_data.put("data2", genre);
26         var_data.put("var3", "releaseDts");
27         var_data.put("data3", "20230101");
28
29         String movie = callAPI(url_key, var_data);
30
31         JSONObject objData = (JSONObject)new JSONParser().parse(movie);
32         JSONArray data = (JSONArray)objData.get("Data");
```

# 코드 설명

```
1 // KoreafilmJson
2
3     // data 둘면서 result배열에서 title 빼오기,
4     // result배열의 ratings객체의 rating배열 0번째 releaseDate (substring(0,8) 할 것) 빼오기
5     // title이랑 date 구했으면 getMovieInfo 메서드에 dto 넣고 상세정보까지 가져오기
6     for(Object obj : data) {
7         JSONObject movieData = (JSONObject)obj;
8
9         JSONArray result = (JSONArray)movieData.get("Result");
10
11        for(Object obj2 : result) {
12            JSONObject movieData2 = (JSONObject)obj2;
13            moviedto = new MovieDTO();
14
15            String title = (String)movieData2.get("title");
16            // 개봉일자 빼오기
17            JSONObject ratings = (JSONObject)movieData2.get("ratings");
18            JSONObject rating = (JSONObject)((JSONArray)ratings.get("rating")).get(0);
19            String date = (String)rating.get("releaseDate");
20            date = date.substring(0,8);
21
22            moviedto.setMovie_title(title);
23            moviedto.setMovie_date(date);
24
25            moviedto = getMovieInfo(moviedto);
26            movieList.add(moviedto);
27        }
28    }
29
30    return movieList;
31}
```

# 코드 설명

```
1 // KoreafilmJson
2
3     // 영화 상세정보 가져와서 MovieDTO로 반환
4     public MovieDTO getMovieInfo(MovieDTO movie) throws IOException, ParseException {
5         String title = movie.getMovie_title();
6         String date = movie.getMovie_date();
7         date = date.replace("-", "");
8
9         var_data.put("var2", "title");
10        var_data.put("data2", title);
11
12        var_data.put("var3", "releaseDts");
13        var_data.put("data3", date);
14
15        String movieinfo = callAPI(url_key, var_data);
16
17        JSONObject objData = (JSONObject)new JSONParser().parse(movieinfo);
18        JSONArray data = (JSONArray)objData.get("Data");
19        JSONObject data1 = (JSONObject)data.get(0);
20        JSONArray result = (JSONArray)data1.get("Result");
21        if(result != null) {
22            for(Object obj : result) {
23                JSONObject movieresult = (JSONObject)obj;
24
25                // 줄거리
26                JSONObject plots = (JSONObject)movieresult.get("plots");
27                JSONArray plot = (JSONArray)plots.get("plot");
28                JSONObject plot1 = (JSONObject)plot.get(0);
29                String plotText = (String)plot1.get("plotText");
30
31                movie.setMovie_plot(plotText);
32            }
33        }
34    }
35}
```

# 코드 설명

```
// KoreafilmJson

107         // 감독
108         JSONObject directors = (JSONObject)movieresult.get("directors");
109         JSONArray director = (JSONArray)directors.get("director");
110         String directorNm = "";
111         if(director != null) {
112             for(Object obj2 : director) {
113                 JSONObject director_ = (JSONObject)obj2;
114                 directorNm = directorNm + (String)director_.get("directorNm") + "|";
115             }
116         }
117         movie.setMovie_director(directorNm);
118
119         // 배우
120         JSONObject actors = (JSONObject)movieresult.get("actors");
121         JSONArray actor = (JSONArray)actors.get("actor");
122         String actorNm = "";
123         if(actor != null) {
124             for(Object obj2 : actor) {
125                 JSONObject actor_ = (JSONObject)obj2;
126                 actorNm = actorNm + (String)actor_.get("actorNm") + "|";
127             }
128         }
129         movie.setMovie_actor(actorNm);

130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
```

# 코드 설명

```
1 // KoreafilmJson
2
3     131         // 장르
4     132         String genre = (String)movieresult.get("genre");
5     133         movie.setMovie_genre(genre);
6
7     135         // 포스터
8     136         String posters = (String)movieresult.get("posters");
9     137         movie.setMovie_poster(posters);
10
11    139         // 고화질 스틸 이미지를 위한 movieId, movieSeq 가져오기
12    140         String movieId = (String)movieresult.get("movieId");
13    141         String movieSeq = (String)movieresult.get("movieSeq");
14
15    143         String stlls = new StillImageCrawling().stillImageCrawling(movieId, movieSeq);
16    144         movie.setMovie_still_image(stlls);
17
18    145     }
19
20    146 }
21
22    147     return movie;
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148 }
```

# 코드 설명

```
1 // StillImageCrawling
2
3     public String stillImageCrawling(String movieId, String movieSeq) {
4         String url = "https://www.kmdb.or.kr/db/kor/detail/movie/" + movieId + "/" + movieSeq + "/own/image#dataHashStillDetail0";
5         String still_image = "";
6
7         try {
8             // 1. Jsoup을 이용해 타겟 HTML 페이지 불러오기
9             Document document = Jsoup.connect(url).get();
10
11             // 2. li 태그 내에 있는 <span class="mImg1"> 안에 있는 style 속성의 이미지 URL 추출
12             Elements listItems = document.select("div.mList8.type3 ul li");
13
14             // 3. 이미지 URL 필터링 및 출력
15             for (Element listItem : listItems) {
16                 // 각 li 태그 안의 a 태그 안에 <span class="mImg1"> 태그의 style 속성 추출
17                 Element spanMImg1 = listItem.selectFirst("a span.mImg1 span[style]");
18
19                 if (spanMImg1 != null) {
20                     // style 속성에서 이미지 URL 추출
21                     String style = spanMImg1.attr("style");
22                     String imageUrl = style.substring(style.indexOf("url(") + 4, style.indexOf(")").replace("\\"", ""));
23
24                     // "poster"가 포함되지 않은 URL만 출력 ("still" URL만)
25                     if (!imageUrl.contains("still")) {
26                         // 앞뒤 '' 제거 후 출력
27                         imageUrl = imageUrl.replace("'", "");
28                         still_image += still_image + imageUrl + "|";
29
30                     }
31                 }
32             }
33         } catch (Exception e) {
34             e.printStackTrace();
35         }
36     }
37
38     return still_image;
39
40
41 }
```

# 팀원 소개

## // Team Responsibilities

&lt;!-- FRONT END --&gt;

team Leader

### 한서진

- ✓ 영화 추천/영화 상세 페이지 Front
- ✓ 전체적인 Front 수정
- ✓ PPT 제작

front-end Team

### 신윤창

- ✓ 마이 페이지
- ✓ 사이트 테스트
- ✓ 플로우 차트 작성

&lt;!-- FULL STACK --&gt;

Back-end Team

### 김영범

- ✓ API 가져오기
- ✓ 영화 추천/영화 상세페이지 Back
- ✓ DB 설계/통합

Back-end Team

### 박성우

- ✓ 메인 페이지
- ✓ 화면 통합
- ✓ GITHUB 관리

Back-end Team &amp; 발표자

### 임지수

- ✓ 게시판
- ✓ 로그인/회원가입
- ✓ 마이페이지

# 소감

## // Table of Contents

### 한재진

팀장으로써 부족한 점이 많았는데, 팀원들이 잘 따라주고 도와주셔서 감사했습니다. 부족한 부분이 있으면 서로 서로 도와줘서 프로젝트가 수월하게 끝날 수 있었습니다. 프로젝트를 하면서 제 자신의 부족한 부분을 알게 되는 계기가 되어 더 열심히 공부할 수 있는 동기가 되었습니다.

### 신윤창

첫 팀 프로젝트를 진행하면서 웹페이지 하나 만드는데 정말 많은 노력이 필요하다는 것을 알았고, 팀원들과의 소통과 협업이 얼마나 중요한지를 깨닫게 되었습니다!

### 박영우

웹페이지 제작은 처음이기도 해서 프로젝트를 진행하려니 막막했었는데, 걱정과 달리 좋은 팀원분들을 만나서 끝까지 완성할 수 있었던 것 같습니다. 소통과 협업이 원활하게 진행되었고, 팀원 모두 실력이 향상된 것 같아 매우 만족한 프로젝트입니다.

### 임지수

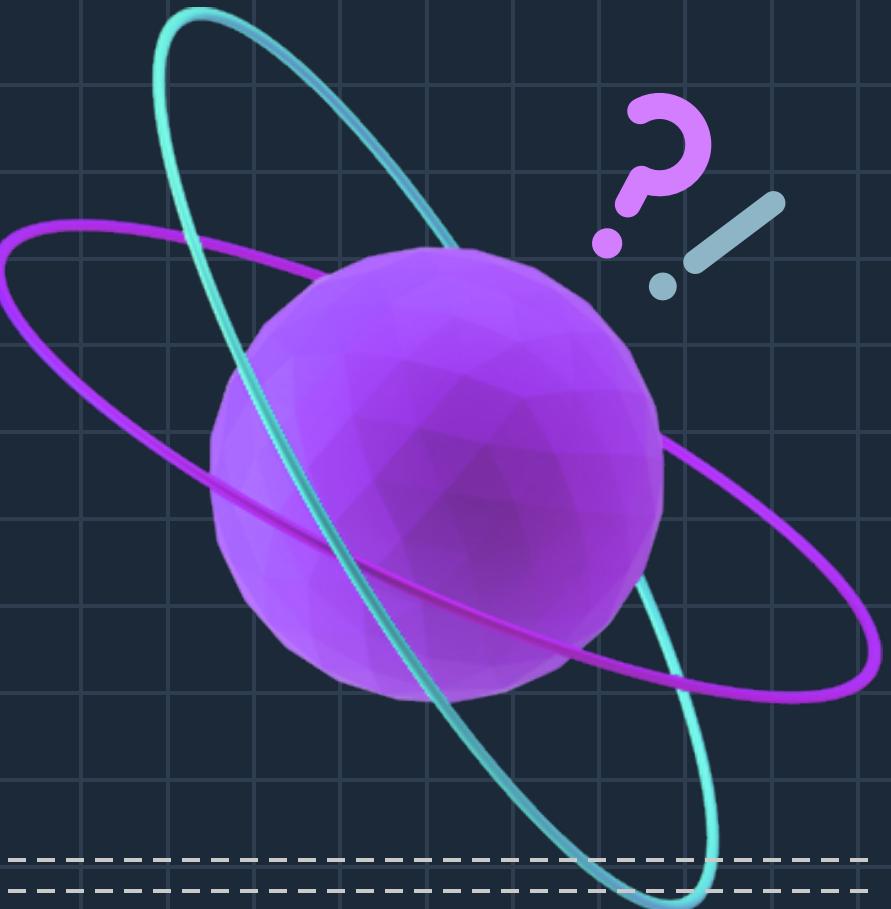
도움이 되지 않을 것 같아서 너무 걱정된 팀프로젝트였는데 팀원들이 많이 배려해주시고 쉽게 물어볼 수 있는 분위기를 만들어주셔서 마지막까지 즐겁게 임할 수 있었습니다. 코드를 작성하고 실제로 실행되는 과정이 아직도 신기하고 재미있어서 좀 더 해보고 싶어요.

### 김영범

팀원 분들과 협업이 너무 재미있었고, 기능 하나 하나 디테일을 추가할 때마다 너무 즐거웠습니다. 특히 이번 프로젝트로 Git을 처음 써 봤는데, 처음 Git을 사용할 땐 히스토리가 꼬이거나 conflict 나는 등 복잡하고 어려웠지만, 점점 Git의 사용법을 알아가며 사용해보니 유용하게 협업이 진행돼서 재미있었습니다. 또한 API를 사용하여 백엔드에서 API를 통해 데이터를 가져오는 과정이 신기하고 공부가 많이 됐습니다.

# 질문과 답변

자유롭게  
질문해주세요!



<!-->  
<!-- "Q & A" -->

13  
14  
15  
16  
17  
18  
19  
20  
21

---

<!-- 정보 -->  
팀원 : 김영범, 박성우, 신윤창, 임지수, 한서진  
강사님 : 곽지현 강사님  
학원 : kh 정보 교육원 (U강의실)

{thank you;