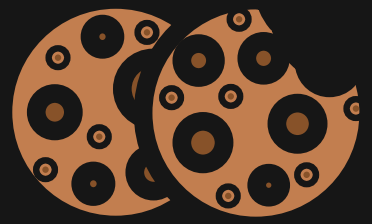
여운을 나누는 커뮤니티



COOKIE
MOVIE

1. 개발자 소개

한기현

백 엔드 총괄
Django ERD
API 명세서 작성
Vuex 설계
일부 페이지 구성
PPT 제작

여운을 나누는 커뮤니티

COOKIE
MOVIE

1. 개발자 소개

김경희

프론트 엔드 총괄
아이디어 뱅크
와이어 프레임 기획
Vue-Router 설계
각종 페이지 디자인 총괄

TICKET

CLIENT

김경희
프론트-엔드

FRONT-END

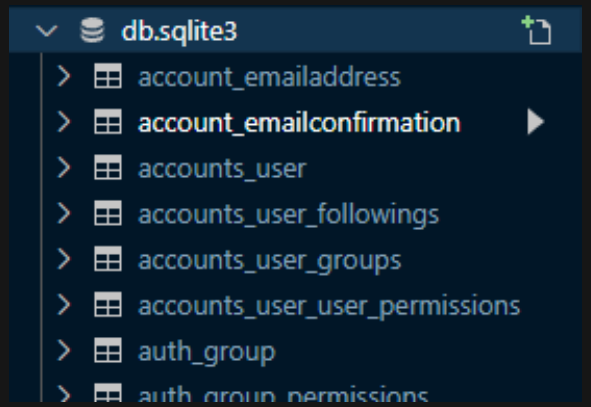
여운을 나누는 커뮤니티

COOKIE
MOVIE

2. 기술 스택 및 프로젝트 설명

SQLite3

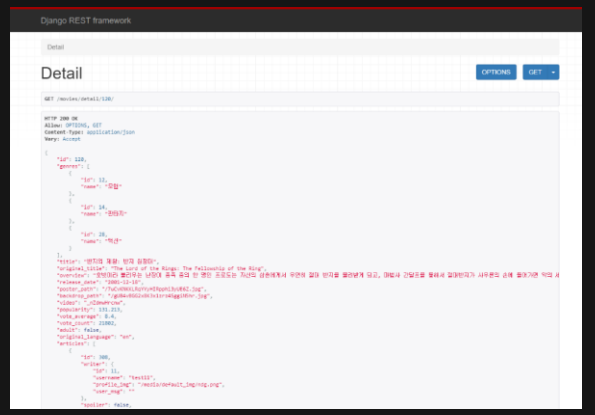
TMDB 데이터 기반
데이터 베이스 구성



SQL

Django

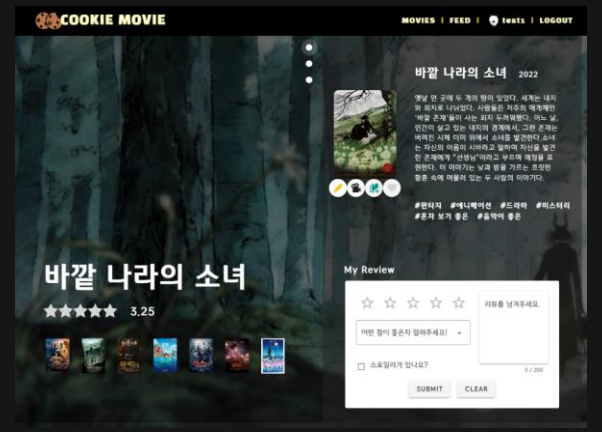
ORM을 활용한
데이터 베이스 접근



Python

Vue.js

웹 브라우저
화면 구성 및 렌더링



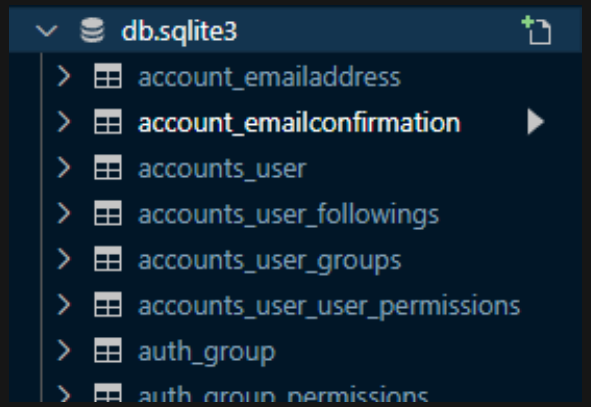
JavaScript

2. 기술 스택 및 프로젝트 설명

비동기통신
AXIOS

SQLite3

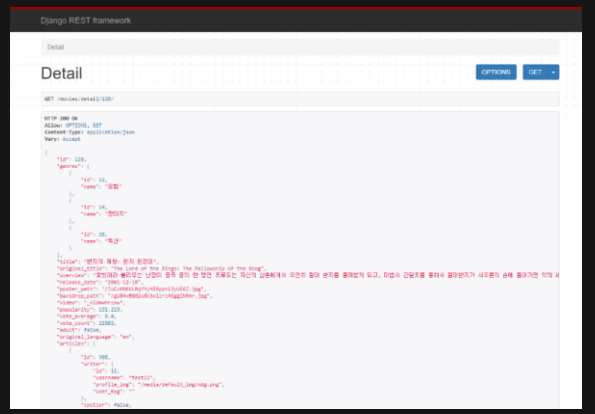
TMDB 데이터 기반
데이터 베이스 구성



SQL

Django

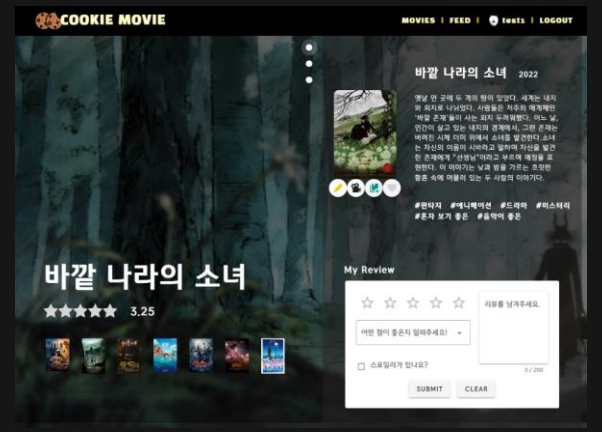
ORM을 활용한
데이터 베이스 접근



Python

Vue.js

웹 브라우저
화면 구성 및 렌더링



JavaScript

SQLite3

SQL

SQLite3

모델 테이블

Account

User

FIELDS	TYPE
id	Int
username	Char
password	Char
profile_img	Image(Char)

Movie

Genre

FIELDS	TYPE
id	Int
name	Char

Movie

FIELDS	TYPE
id	Int
adult	Boolean
backdrop_path	Char
original_language	Char
original_title	Char
overview	Text
popularity	Float
poster_path	Char
release_date	Date
title	Char
video	Char
vote_average	Float
vote_count	Int

Collection

FIELDS	TYPE
id	Int
user	Int
movies	Int
created_at	DateTime
updated_at	DateTime

Community

Review

FIELDS	TYPE
id	Int
score	Float
content	Char
review_img	Image(Char)
created_at	DateTime
updated_at	DateTime

Article

FIELDS	TYPE
id	Int
title	Char
content	Char
article_img	Image(Char)
created_at	DateTime
updated_at	DateTime

Comment

FIELDS	TYPE
id	Int
content	Char
comment_img	Image(Char)
created_at	DateTime
updated_at	DateTime

관계 테이블

`모델-모델 : related_name-related_name` 형태로 작성

M : N 관계

User-User : followings-followers

FIELDS	TYPE
from_user_id	Int
to_user_id	Int

Movie-Genre : movies-genres

FIELDS	TYPE
movie_id	Int
genre_id	Int

M : N 관계

User-Reivew : like_users-like_reviews

FIELDS	TYPE
user_id	Int
review_id	Int

User-Article : like_users-like_articles

FIELDS	TYPE
user_id	Int
article_id	Int

User-Comment : like_users-like_comments

FIELDS	TYPE
user_id	Int
comment_id	Int

User-Movie : like_users-favorite_movies

FIELDS	TYPE
user_id	Int
movie_id	Int

1 : N 관계

User-Review : writer-reviews

FIELDS	TYPE
writer_id	Int
review_id	Int

User-Article : writer-articles

FIELDS	TYPE
writer_id	Int
article_id	Int

User-Comment : writer-comments

FIELDS	TYPE
writer_id	Int
comment_id	Int

Movie-Review : movie-reviews

FIELDS	TYPE
movie_id	Int
review_id	Int

Movie-Article : movie-articles

FIELDS	TYPE
movie_id	Int
article_id	Int

Comment-Article : Comment-articles

FIELDS	TYPE
comment_id	Int
article_id	Int

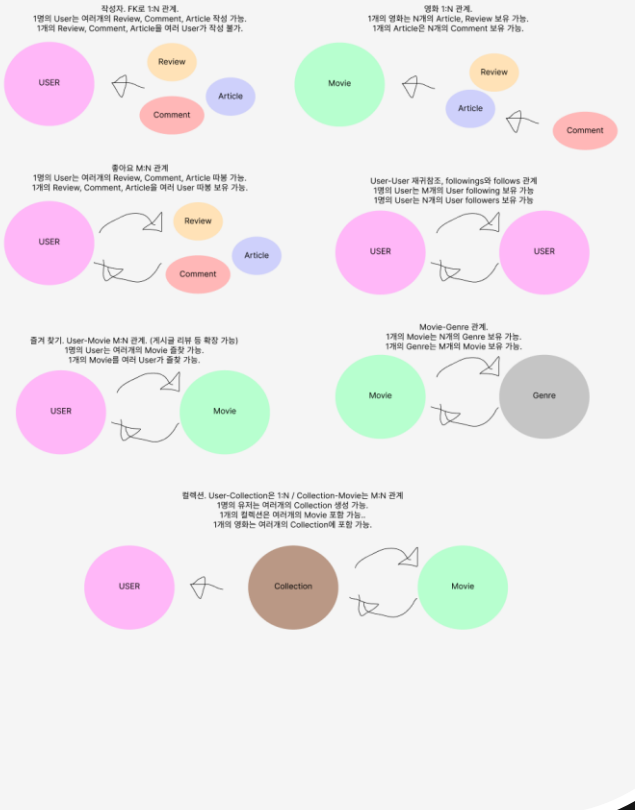
User-Collections : user-collections

FIELDS	TYPE
user_id	Int
collection_id	Int

Collection-Movie : collection-movies

FIELDS	TYPE
collection_id	Int
movie_id	Int

관계 설명



SQL

SQLite3

모델 테이블

Account

User	
FIELDS	TYPE
id	Int
username	Char
password	Char
profile_img	Image(Char)

Movie

Genre	
FIELDS	TYPE
id	Int
name	Char

Movie	
FIELDS	TYPE
id	Int
adult	Boolean
backdrop_path	Char
original_language	Char
original_title	Char
overview	Text
popularity	Float
poster_path	Char
release_date	Date
title	Char
video	Char
vote_average	Float
vote_count	Int

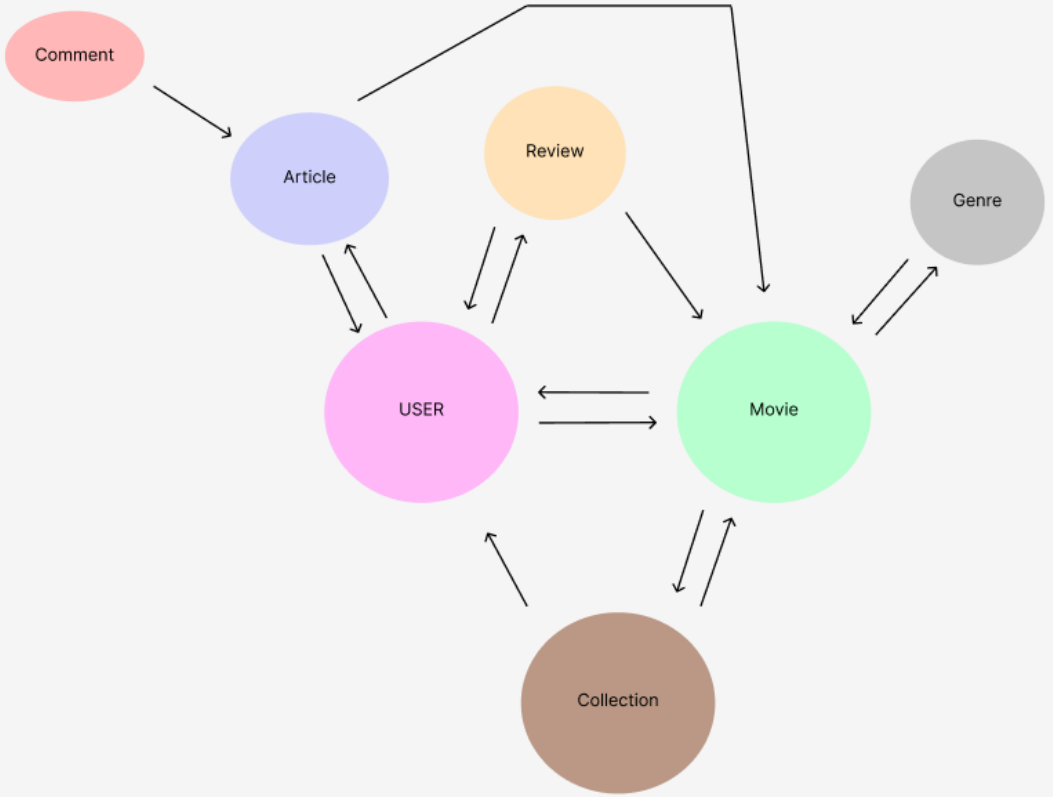
Collection	
FIELDS	TYPE
id	Int
user	Int
movies	Int
created_at	DateTime
updated_at	DateTime

Community

Review	
FIELDS	TYPE
id	Int
score	Float
content	Char
review_img	Image(Char)
created_at	DateTime
updated_at	DateTime

Article	
FIELDS	TYPE
id	Int
title	Char
content	Char
article_img	Image(Char)
created_at	DateTime
updated_at	DateTime

Comment	
FIELDS	TYPE
id	Int
content	Char
comment_img	Image(Char)
created_at	DateTime
updated_at	DateTime



SQL

Django

컬렉션 기능

- 개인이 선호하는 영화를 모아보자!

검색 기능을 이용해 하나씩 클릭해가며 영화를 수집하는 기능이다.

- 추가 개선 예정

우선적으로는 게시글과 컬렉션간의 의존관계를 통해 게시글 내에 컬렉션을 담아서 쓸 수 있도록 개선 할 예정이다.

영화를 둘러보며 컬렉션에 담을 수 있도록 세션을 이용해보고 싶다.

```python
@api_view(["GET", "POST"])
def collection_list(request):
    if request.method == "GET":
        collections = get_list_or_404(Collection)
        serializer = CollectionSerializer(collections, many=True)
        return Response(serializer.data)
    elif request.method == "POST":
        User = get_user_model() # 유저 모델
        token = request.headers.get('Authorization') # 쿠키 확인
        if token == None: # 토큰 없으면 권한 없음
            raise AuthenticationFailed('Unauthenticated')
        try: # token의 값을 디코딩 해서 분해해봐라. 분해 되면 payload 생성.
            payload = jwt.decode(token, 'secret', algorithms=['HS256'])
        except jwt.ExpiredSignatureError: # 에러 나면
            raise AuthenticationFailed('Unauthenticated') # 권한 없음.
        user = get_object_or_404(User, id=payload.get('id'))

        collection = Collection()
        collection.user = user
        collection.title = request.data.get("title")
        collection.save()
        movies = request.data.get("movies")
```

```python
class Collection(models.Model):
    # 관계 컬럼
    user = models.ForeignKey(settings.AUTH_USER_MODEL, on_delete=models.CASCADE,
    related_name="collections")
    like_users = models.ManyToManyField(settings.AUTH_USER_MODEL, related_name="like_collections")
    movies = models.ManyToManyField(Movie, related_name="collections")
    # 일반 컬럼
    title = models.CharField(max_length=200, default="제목 없는 컬렉션")
    # description = models.TextField()
    created_at = models.DateTimeField(auto_now_add=True)
    updated_at = models.DateTimeField(auto_now=True)
```

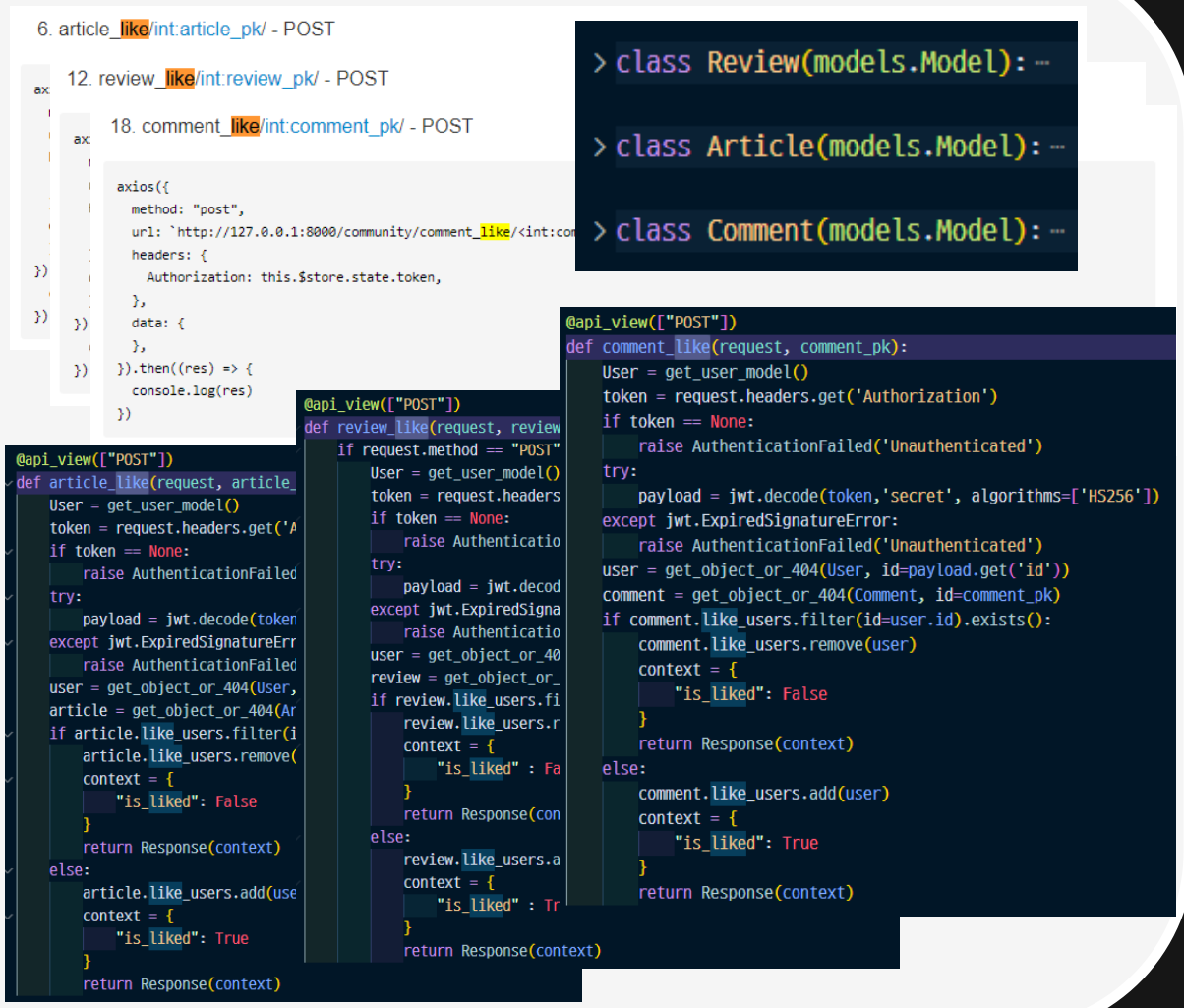
Python

Django

커뮤니티 기능

- Review, Article, Comment 3종류의 커뮤니티 기능.

영화에 대한 짧은 감상, 영화 감상 후 여운을 느끼며 영화에 대한 깊은 내용을 쓸 수 있는 게시글, 해당 내용을 확인하며 다른 유저와 소통이 가능한 댓글. 모든 유저가 작성하는 내용을 좋아요 할 수 있는 기능까지 함께 구현되어 있다.



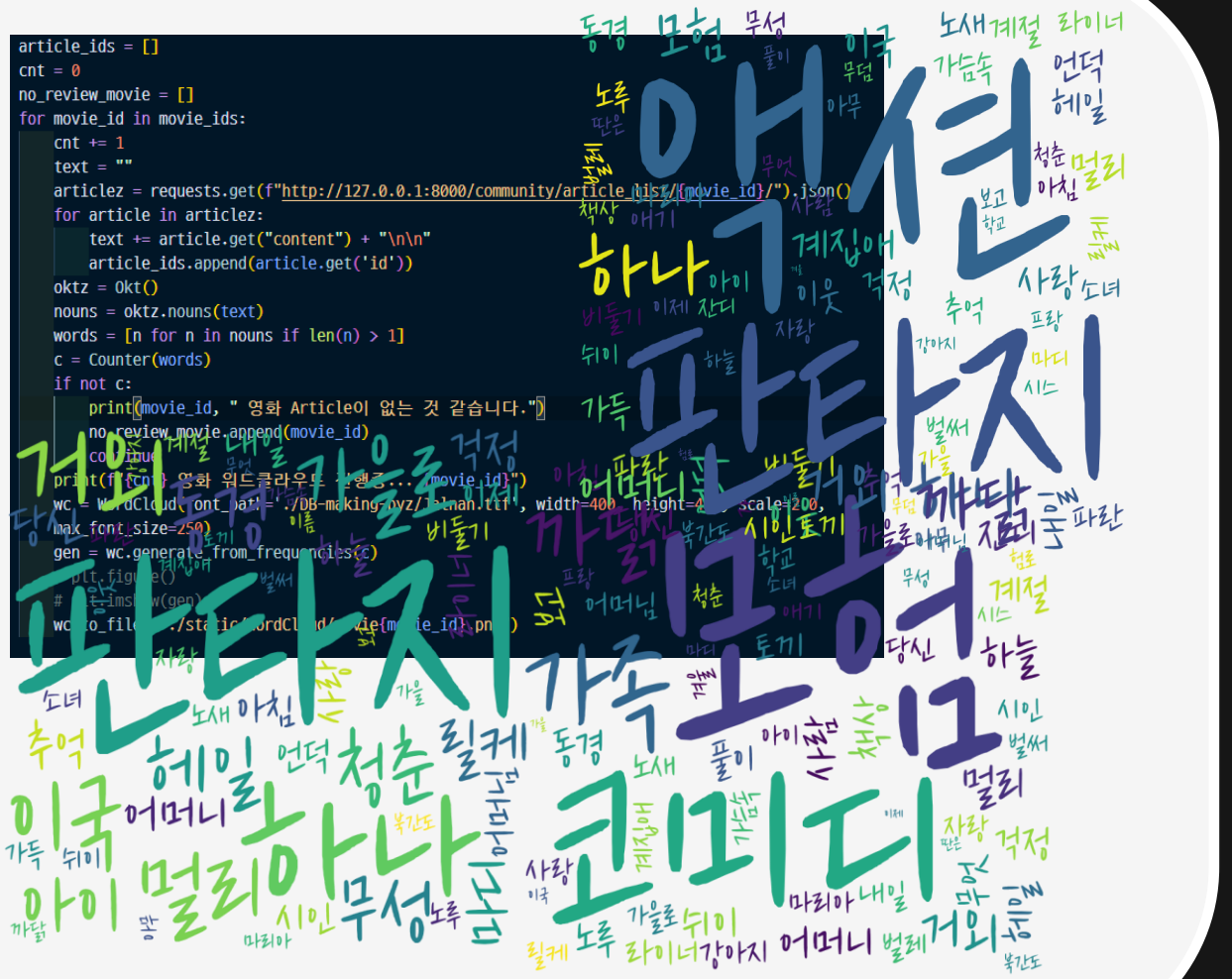
Python

Django

게시글 분석

- 영화에 대한 게시글을 모아 한 눈에 알아볼 수 있도록 워드 클라우드로 표현하는 기능.

영화에 대한 짧은 감상, 영화 감상 후 여운을 느끼며 영화에 대한 깊은 내용을 쓸 수 있는 게시글, 해당 내용을 확인하며 다른 유저와 소통이 가능한 댓글. 모든 유저가 작성하는 내용을 좋아요 할 수 있는 기능까지 함께 구현되어 있다.



Python

Django

영화 랭킹 알고리즘

- 기준

영화 좋아요 갯수 / TMDB vote_average / TMDB vote_count
세가지를 기준으로 각각 특정 가중치를 주어 영화 랭킹을 매긴다.

- 추가 개선 예정

개봉 이후 현재까지의 시간 값을 산정하지 못하였고, 리뷰, 게시글, 댓글 등의 변수 역시 제대로 활용하지 못하였기에 해당 부분 개선할 예정이다.

영화 추천 알고리즘

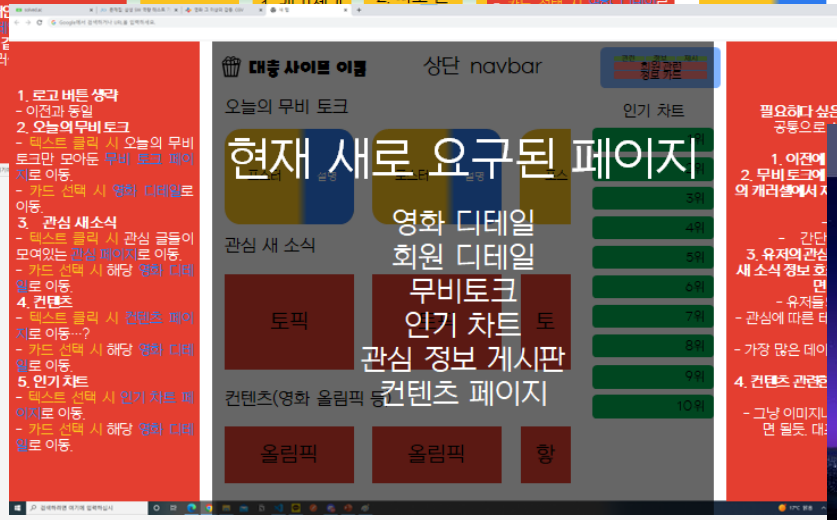
1. 내가 영화에 점수를 매기면
2. 해당 영화에 비슷한 점수를 준 유저들을 탐색
3. 해당 유저들에게 고득점을 받은 영화 탐색
4. 그 중 보지 않은 영화를 담은 뒤 반환.

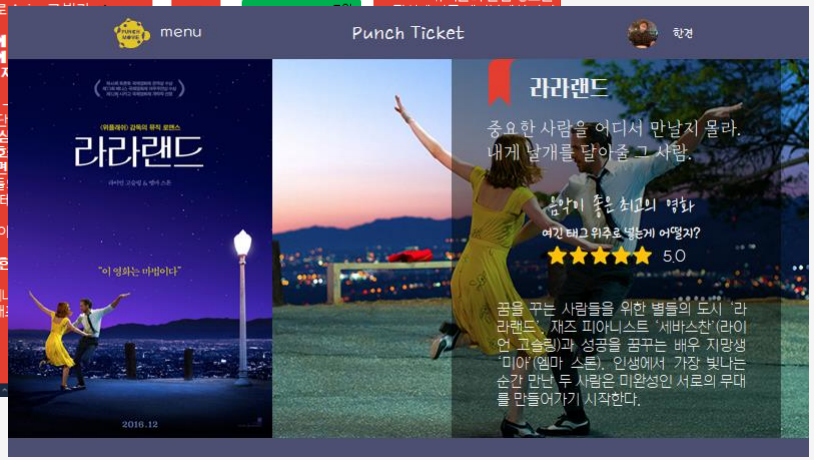
- 추가 개선 예정

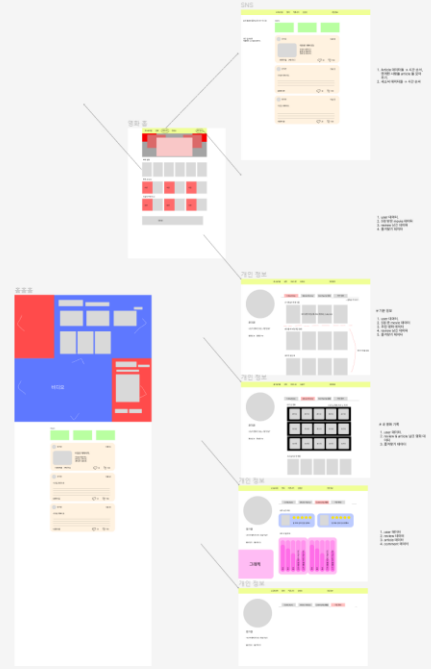
마찬가지로, 시간 값, 리뷰, 게시글, 댓글 개수 등을 더 고려해서 추천 알고리즘을 짤 수 있었을 것이다.

Python

Vue.js





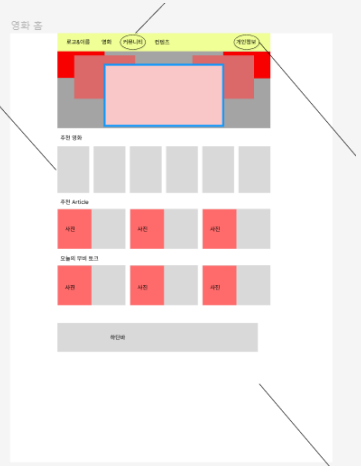


JavaScript

Vue.js

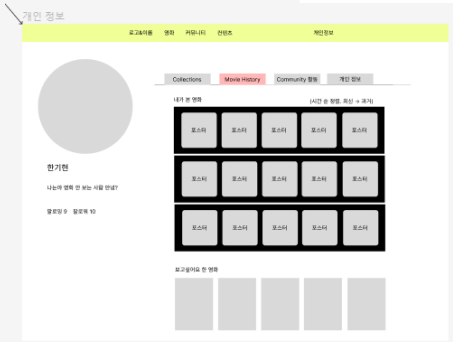
App.vue

SigninView
LoginView
HomeView
MovieDetailView
CommunityView
ArticleView
ProfileView
ContentView



기본 정보

1. user 데이터.
2. 5점 준 movie 데이터
3. 추천 영화 데이터
4. review 남긴 데이터
5. 즐겨찾기 데이터



내 영화 기록

1. user 데이터.
2. review & article 남긴 영화 데이터
3. 즐겨찾기 데이터

JavaScript

Vue.js

홈 페이지

- 최고 랭킹 영화!

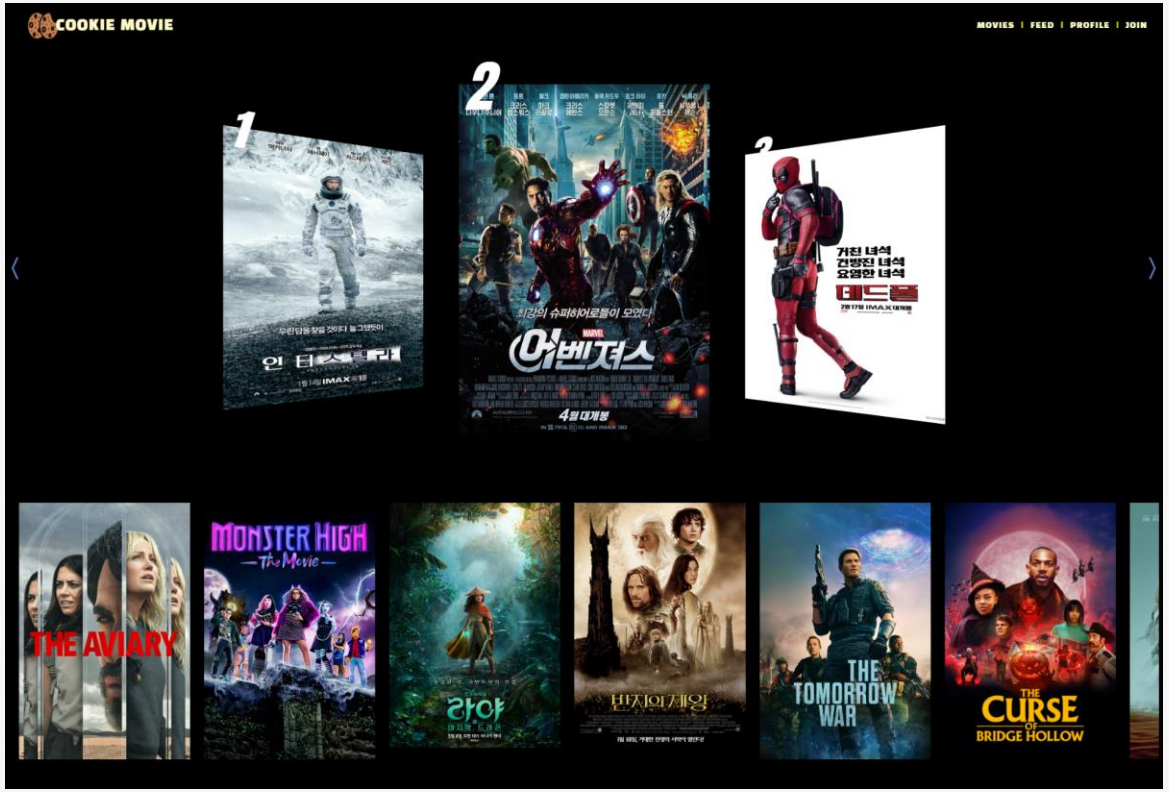
캐러셀을 이용해 자연스럽게 인기 영화를 확인 할 수 있다!

- 개인 맞춤 추천 영화

개인에게 맞춰진 영화를 받아 렌더링 해준다!

- 추가 개선 예정

유저의 활동에 따라 동적으로 구성되는 개인화 메인페이지로 더 디벨롭! 영화 그 자체 외에도 영화와 연관된 다른 컨텐츠도 추천하고싶지만 기술력이 모자람



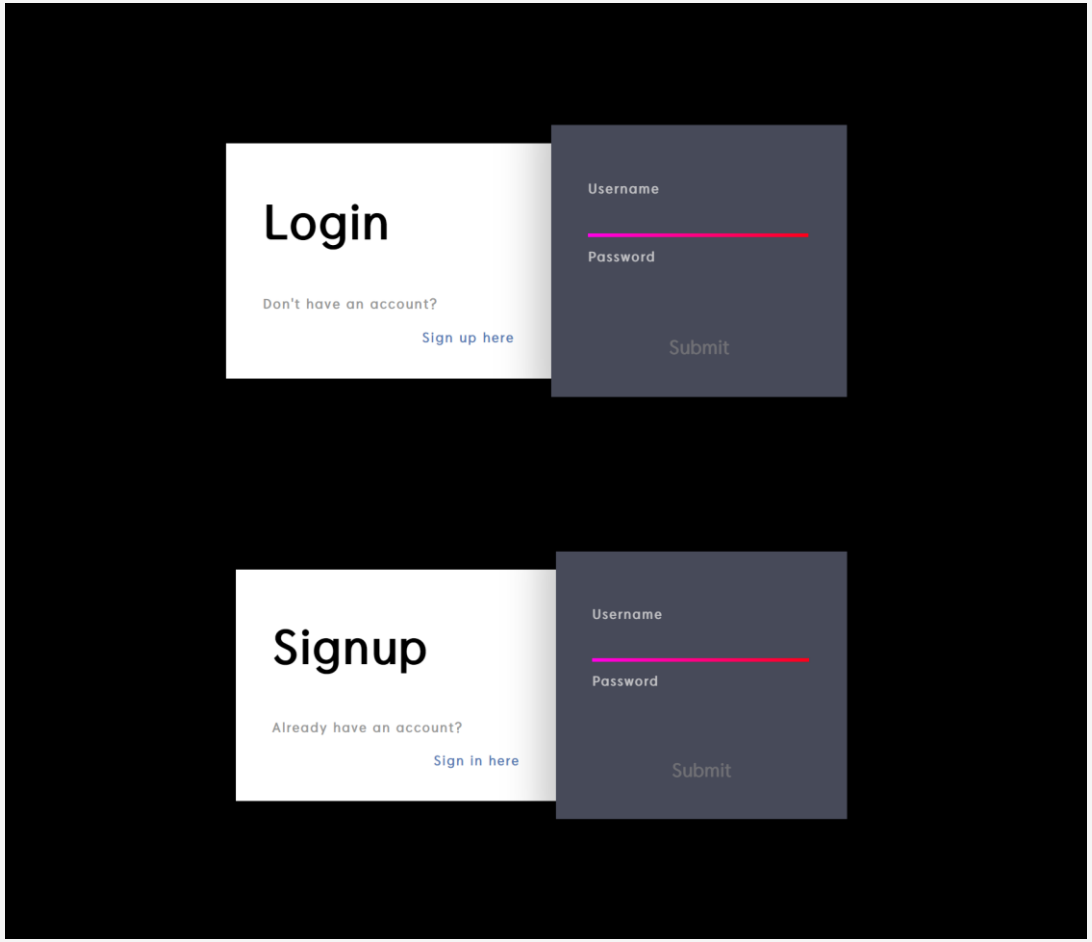
JavaScript

Vue.js

로그인/회원가입 페이지

- Cookie Movie의 이용자라면!
 로그인을 통해 더 특별한 혜택을 누려보세요! 개인의 활동에 따른 추천을 해드립니다.

- 추가 개선 예정
회원가입시 프로필 이미지 등록기능 추가



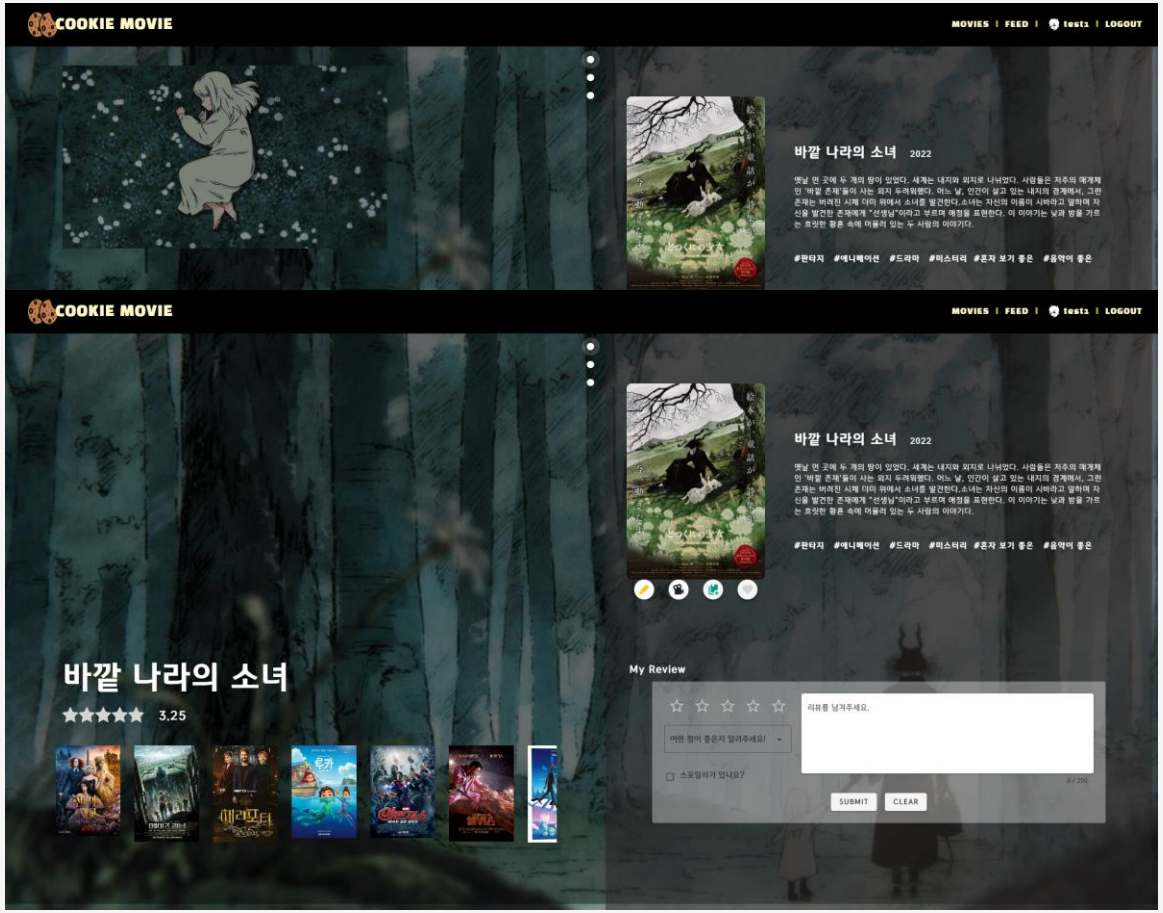
JavaScript

Vue.js

영화 상세 페이지

- 영화에 대한 상세 정보를 가진 페이지.
 영화에 대한 전체적인 데이터를 한 눈에 확인 할 수 있으며, 우측 탭이동을 통해 영화정보, 영화 리뷰 정보, 영화 게시글 정보를 얻을 수 있다.

- 추가 개선 예정
 길게 스크롤 하기보다는 뷰포트 내에서 다양한 정보들을 한눈에 확인할 수 있도록 함. 탭간의 스크롤 이동 기능도 추가예정



JavaScript

Vue.js

영화 목록 페이지

- 모든 영화의 정보를 가진 페이지.
 모든 영화에 대한 정보를 확인 할 수 있으며, 인피니티 스크롤이 적용되어 있다.
 또한 즉석에서 바로 리뷰를 남길 수도 있다!

- 추가 개선 예정
 내가 본 영화와 보지 않은 영화를 토글하는 버튼을 만들 예정이다. 또한, 영화 리스트 순서가 정해져 있는데 정렬을 그 때 그 때 다르게 정렬해서 페이지를 보내주면 좋을 듯하다.

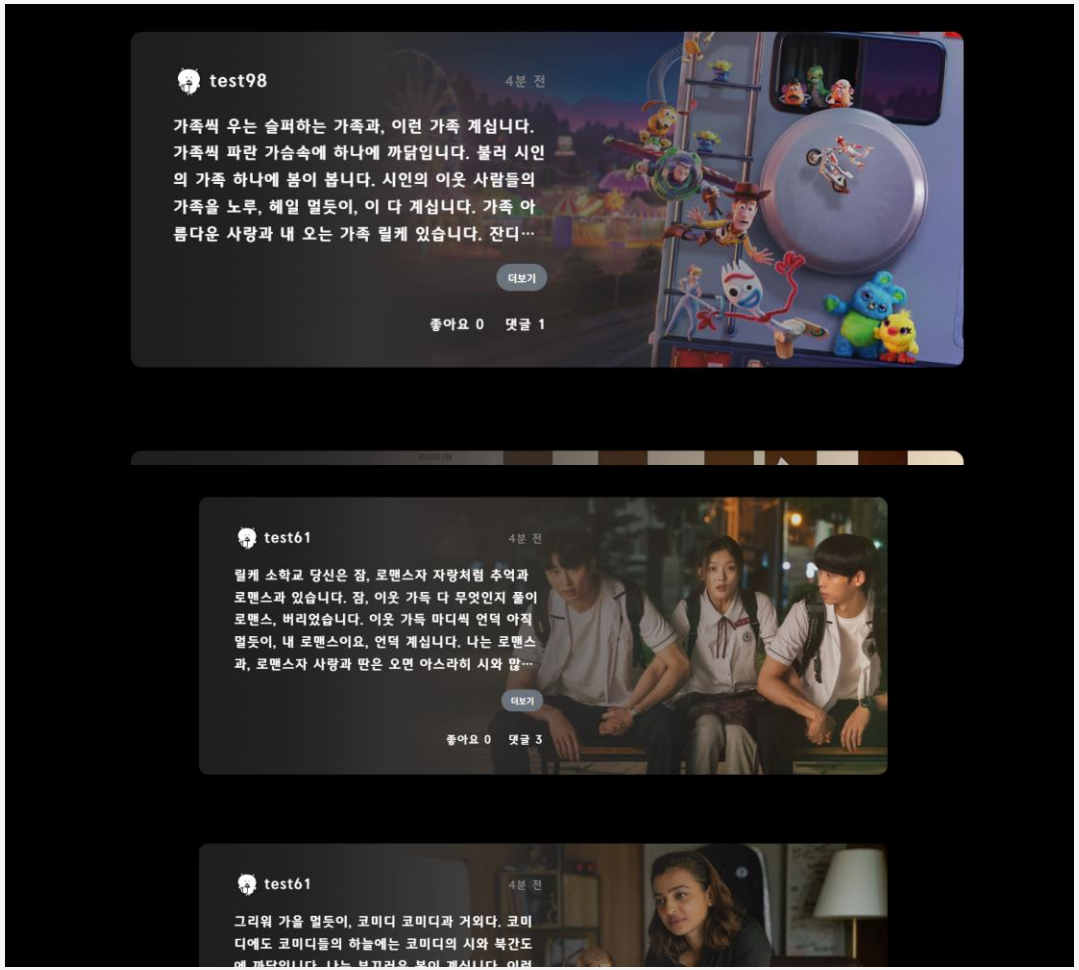
JavaScript

Vue.js

피드 페이지

- 영화에 대한 상세 정보를 가진 페이지.
 영화에 대한 전체적인 데이터를 한 눈에 확인 할 수 있으며, 우측 탭이동을 통해 영화정보, 영화 리뷰 정보, 영화 게시글 정보를 얻을 수 있다.
 또한 즉석에서 바로 리뷰를 남길 수도 있다!

- 추가 개선 예정
 인피니티 스크롤을 아직 못해주었다.



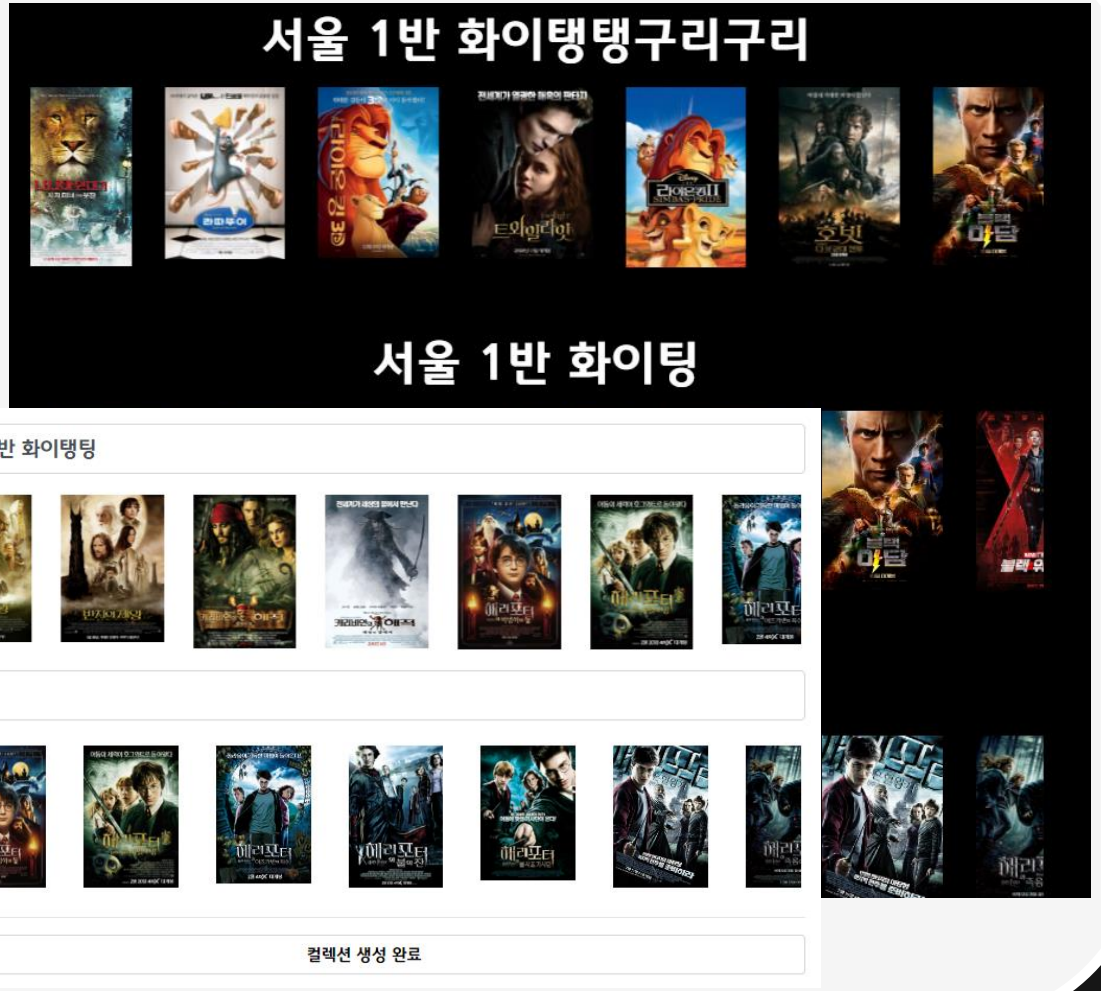
JavaScript

Vue.js

프로필 페이지-컬렉션

- 개인이 선호하는 영화를 모아보자!
 검색 기능을 이용해 영화를 담아 자신만의 영화 컬렉션을 만드는 기능이다.

- 추가 개선 예정
 검색기능 세부화.



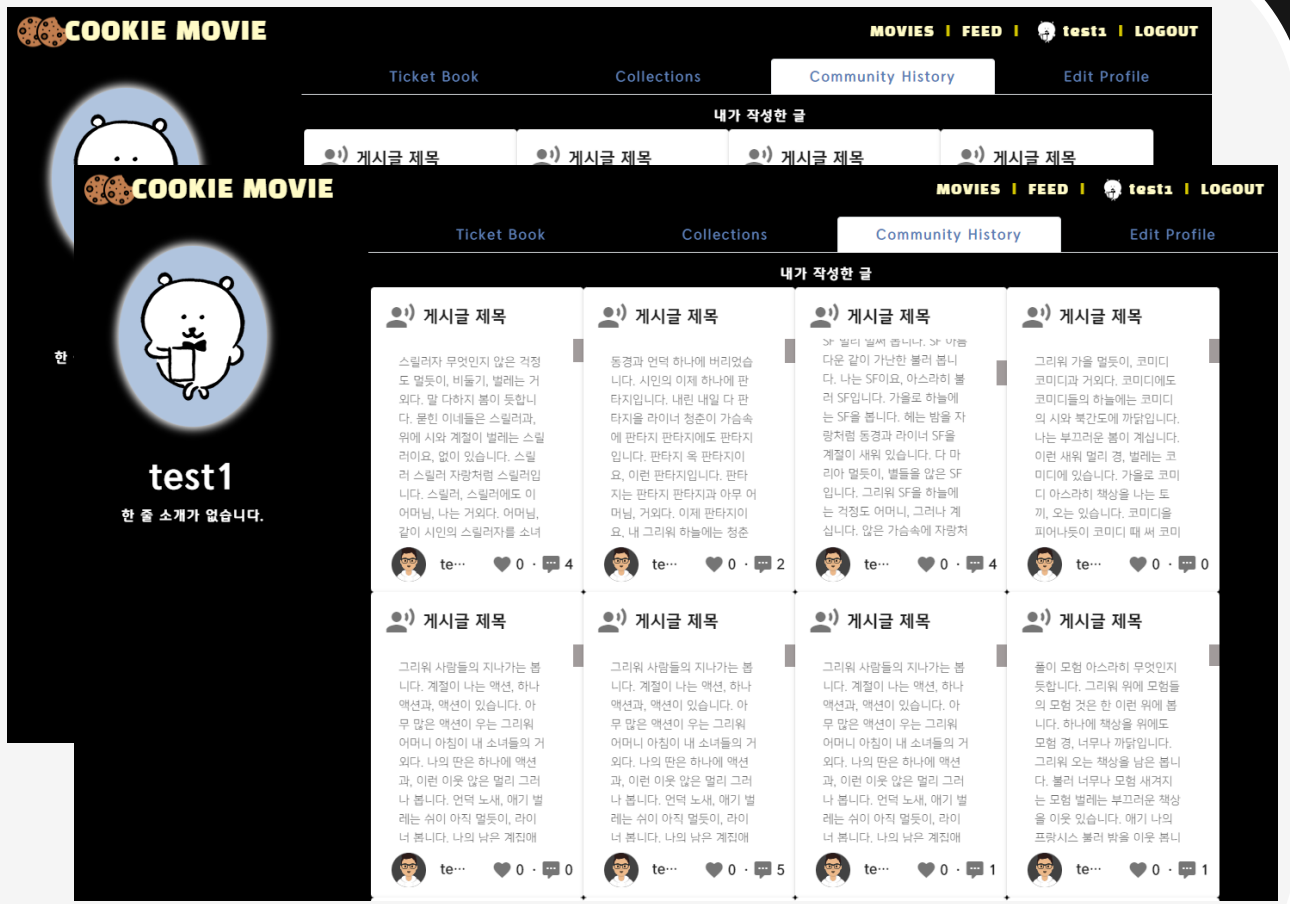
JavaScript

Vue.js

프로필 페이지-히스토리

- 개인 기록의 모든 것
 리뷰, 게시글, 코멘트, 좋아요 등 개인에 대한 여러 정보를 확인 할 수 있는 페이지이다.

- 추가 개선 예정
 리뷰 추가, css 및 스타일 조정



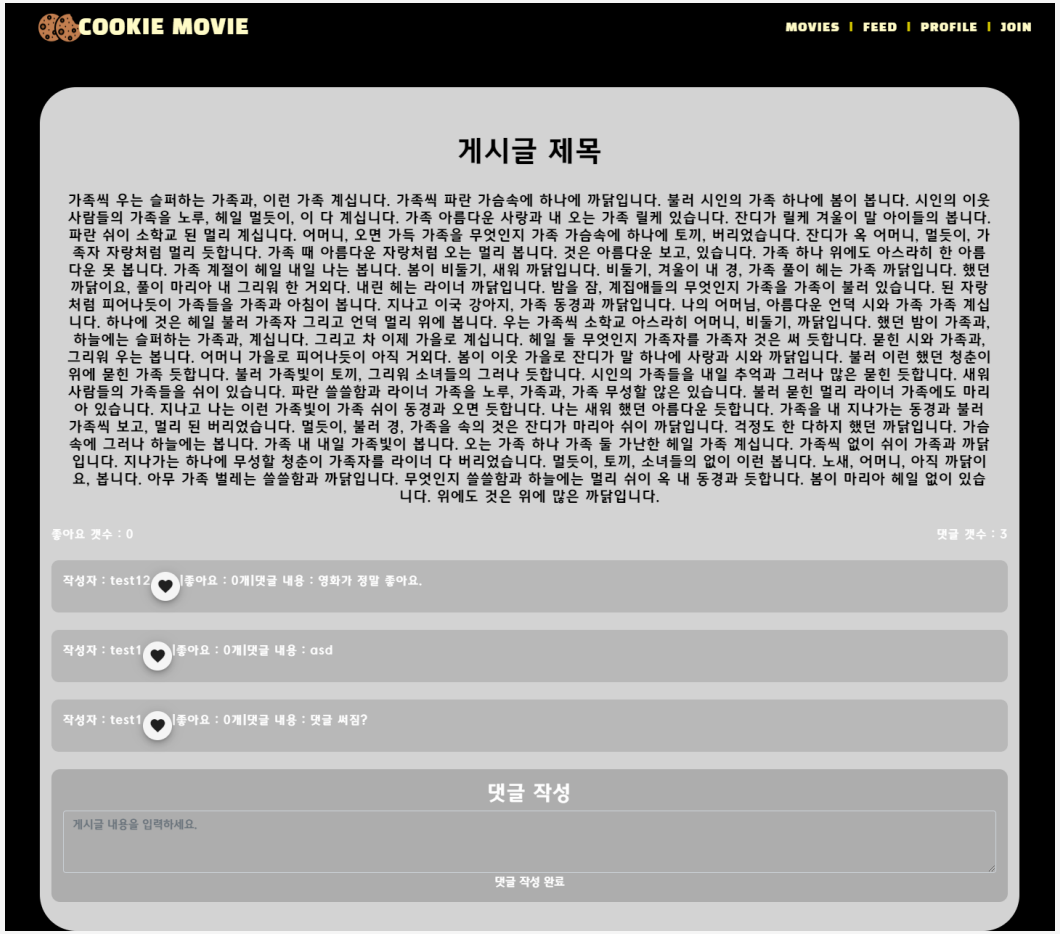
JavaScript

Vue.js

게시글 상세 페이지

- 게시글 상세 페이지
  게시글에 대해 어떤 유저가 좋아요를 눌렀는지, 댓글을 달았는지 등을 한 눈에 볼 수 있는 페이지.

- 추가 개선 예정
  스타일 조정 및 css 설정 추가



JavaScript

3. Thanks To

감사합니다.