

# 06 PJT

# 관계형 데이터베이스 설계

# INDEX

---

- 목표
- 준비사항
- 요구사항
- 제출

# 목표

## 프로젝트 목표

- 데이터를 생성, 조회, 수정, 삭제할 수 있는 Web application 제작
- Django web framework를 사용한 데이터 처리
- Django Model과 ORM에 대한 이해
- Django Authentication System에 대한 이해
- Database many to one relationship (1:N)  
및 many to many relationship (M:N) 에 대한 이해

# 준비사항

## | 개발도구

- Visual Studio Code
- Google Chrome
- Django 4.2.x

# 요구사항



## 공통 요구사항

- 프로젝트의 이름
  - mypjt
- 앱 이름
  - movies
    - 영화정보 생성, 조회, 수정, 삭제, 좋아요, 댓글 작성, 댓글 조회, 댓글 삭제
  - accounts
    - 로그인, 로그아웃, 회원가입, 회원탈퇴, 회원정보 수정, 비밀번호 수정, 팔로우
- .gitignore 파일을 추가하여 불필요한 파일 및 폴더는 제출하지 않음

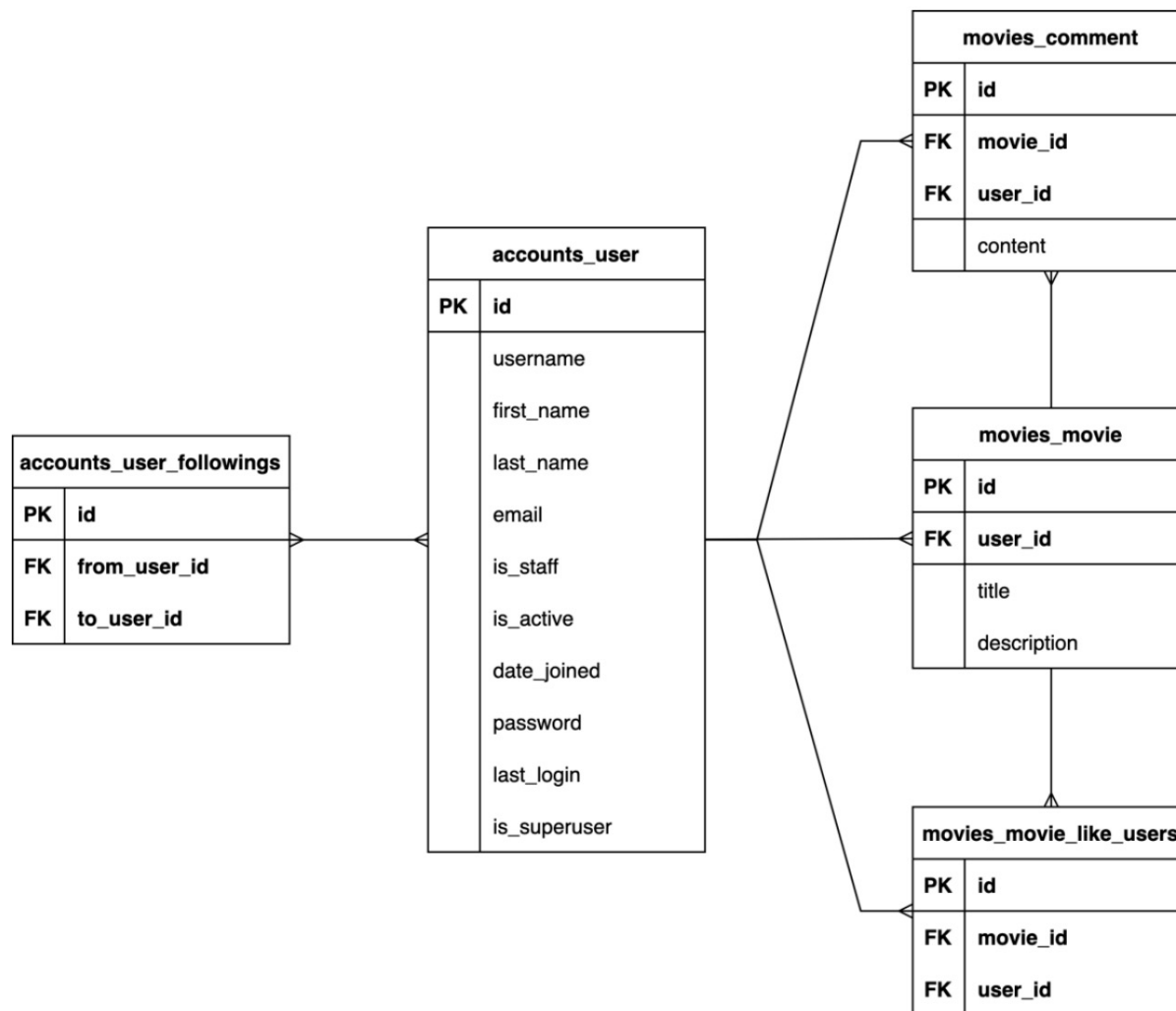
## 필수 요구사항 - 모델 클래스

- Movie 모델 클래스
  - 영화 제목과 줄거리를 입력할 필드 2개 지정
  - 작성자 정보를 저장할 필드 1개 지정
  - 좋아요 한 유저 정보를 저장하는 필드 1개 지정
- Comment 모델 클래스
  - 댓글 정보를 입력할 필드 1개 지정
  - 댓글이 작성 될 영화 정보와 댓글 작성자 정보를 저장할 필드 2개 지정

## 필수 요구사항 - 모델 클래스

- User 모델 클래스
  - AbstactUser 모델 클래스를 상속받는 커스텀 모델을 사용
  - 팔로우 유저 정보를 저장하는 필드 1개 지정
- 생성되는 중계 테이블 안내
  - A. `movies_movie_like_users`
    - movie와 user의 id 정보가 저장됨
  - B. `accounts_user_followings`
    - follow 관계를 가지는 user 정보가 저장됨

# ERD (Entity-Relationship Diagram)



## | 필수 요구사항

- 본인이 작성한 데이터만 수정 및 삭제 가능하도록 구성
- 로그인 한 사람들만 데이터 생성, 수정, 삭제 할 수 있도록 구성
- 로그인 한 회원만 댓글을 생성, 삭제할 수 있음
- 본인이 작성한 댓글에 대해서만 삭제할 수 있음
- 비밀번호 변경 직후 로그인 상태를 유지해야 함

## 필수 요구사항 – view 함수

- movies 앱의 view 함수

함수명	역할	허용 HTTP Method
index	<ul style="list-style-type: none"><li>• 전체 영화 데이터 조회 및 index.html 렌더링</li></ul>	GET
create	<ul style="list-style-type: none"><li>• create.html 렌더링</li><li>• 유효성 검증 및 영화 데이터 저장 후 detail 로 리다이렉트</li></ul>	GET & POST
detail	<ul style="list-style-type: none"><li>• detail.html 렌더링</li><li>• 단일 영화 데이터 조회</li><li>• 단일 영화에 달린 모든 댓글 데이터 조회</li></ul>	GET
update	<ul style="list-style-type: none"><li>• 수정 대상 영화 데이터 조회 및 update.html 렌더링</li><li>• 본인이 작성한 영화 데이터만 수정 가능</li><li>• 유효성 검증 및 영화 데이터 수정 후 detail 로 리다이렉트</li></ul>	GET & POST
delete	<ul style="list-style-type: none"><li>• 본인이 작성한 영화 데이터만 삭제 가능</li><li>• 단일 영화 데이터 삭제 및 index 로 리다이렉트</li></ul>	POST

## 필수 요구사항 – view 함수

- movies 앱의 view 함수

함수명	역할	허용 HTTP Method
comments_create	<ul style="list-style-type: none"><li>• 유효성 검증 및 댓글 데이터 저장 후 detail 로 리다이렉트</li></ul>	POST
comments_delete	<ul style="list-style-type: none"><li>• 단일 댓글 데이터 삭제 및 detail 로 리다이렉트</li><li>• 본인이 작성한 댓글만 삭제 가능</li></ul>	POST
likes	<ul style="list-style-type: none"><li>• 단일 영화 좋아요 기능 (이미 좋아요 눌린 경우는 좋아요 취소)</li><li>• 좋아요 기능 동작 후 index 로 리다이렉트</li></ul>	POST

## 필수 요구사항 – view 함수

- accounts 앱의 view 함수

함수명	역할	허용 HTTP Method
login	<ul style="list-style-type: none"><li>login.html 렌더링</li><li>로그인 절차 진행 후 index 로 리다이렉트</li></ul>	GET & POST
logout	<ul style="list-style-type: none"><li>DB와 클라이언트의 쿠키에서 인증된 사용자의 세션 데이터 삭제</li></ul>	POST
signup	<ul style="list-style-type: none"><li>signup.html 렌더링</li><li>유효성 검증 및 회원 데이터 저장 후 index 로 리다이렉트</li></ul>	GET & POST
update	<ul style="list-style-type: none"><li>수정 대상 회원 데이터 조회 및 update.html 렌더링</li><li>유효성 검증 및 회원 데이터 수정 후 index 로 리다이렉트</li></ul>	GET & POST
delete	<ul style="list-style-type: none"><li>단일 회원 데이터 삭제 및 index 로 리다이렉트</li></ul>	POST
change_password	<ul style="list-style-type: none"><li>change_password.html 렌더링</li><li>비밀번호 변경 후 index 로 리다이렉트</li></ul>	GET & POST



## 필수 요구사항 – view 함수

- accounts 앱의 view 함수

함수명	역할	허용 HTTP Method
profile	<ul style="list-style-type: none"><li>profile.html 렌더링</li><li>단일 회원 데이터 및 작성한 영화, 댓글, 팔로우 수, 팔로잉 수 조회</li></ul>	POST
follow	<ul style="list-style-type: none"><li>단일 회원 팔로우 기능 (이미 팔로우 한 경우 팔로우 취소)</li><li>팔로우 기능 동작 후 profile.html 리다이렉트</li></ul>	POST

## | 필수 요구사항 - Admin

- 모델 Movie, Comment, User 를 Admin site에 등록
- Admin site에서 각 모델들의 데이터 생성, 조회, 수정, 삭제가 가능해야 함

## | 필수 요구사항 – Form Class

- Movie 모델과 Comment 모델의 데이터 검증, 저장, 에러메시지, HTML을 모두 관리하기 위해 적절한 ModelForm을 사용
- User 모델의 데이터 검증, 저장, 에러메시지, HTML을 모두 관리하기 위해 적절한 Built-in Form과 Built-in ModelForm 그리고 필요한 경우 커스텀 한 ModelForm을 사용

## | 필수 요구사항 - 템플릿

- 모든 페이지 상단에는 네비게이션 바가 나올 수 있도록 작성
  - 로그인 유무에 따라 보여지는 메뉴가 다름
- Index 페이지는 주어진 Image가 나올 수 있도록 작성

## 템플릿 예시 - Index

MOVIE
회원가입 로그인

INDEX PAGE

[첫 번째 영화](#)

Sept. 22, 2023, 1:55 p.m.

좋아요 1

좋아요


[두 번째 영화](#)

Sept. 22, 2023, 1:56 p.m.

좋아요 1

좋아요

### 템플릿 예시 - 좋아요



INDEX PAGE

첫 번째 영화

Sept. 22, 2023, 1:55 p.m.

좋아요 1


좋아요

두 번째 영화

Sept. 22, 2023, 1:56 p.m.

좋아요 1

좋아요



첫 번째 영화

Sept. 22, 2023, 1:55 p.m.

좋아요 2

좋아요 취소

두 번째 영화

Sept. 22, 2023, 1:56 p.m.

좋아요 1

좋아요

### 템플릿 예시 - 프로필 (본인)

- 본인은 팔로우 할 수 없음

Lorem님의 프로필

영화 목록

[네 번째 영화](#)

팔로워: 0

팔로잉: 1

### 템플릿 예시 - 프로필 (타 유저)

b님의 프로필

영화 목록

[첫 번째 영화](#)

[두 번째 영화](#)

[세 번째 영화](#)

팔로워: 0

팔로우

팔로잉: 1

b님의 프로필

영화 목록

[첫 번째 영화](#)

[두 번째 영화](#)

[세 번째 영화](#)

팔로워: 1

팔로우 취소

팔로잉: 1



## [도전 과제 1] - 좋아요 누른 영화 목록 확인

1. 프로필 페이지에 해당 프로필의 사용자가 누른 좋아요 개수를 출력
2. 좋아요 한 영화 리스트 목록 출력
  - 새로운 페이지로 이동할 수 있는 링크 추가
  - 좋아요를 누른 영화를 리스트로 출력하고 해당 영화의 세부 페이지로 이동할 수 있는 링크 작성

## [도전 과제 1] - 좋아요 누른 영화 목록 확인

Lorem님의 프로필

영화 목록

[네 번째 영화](#)

**좋아요 누른 개수 : 1**

[좋아요 영화 확인](#)

[팔로워](#): 0

[팔로잉](#): 1

Liked Movie List

[Back](#)

- [첫 번째 영화](#)

## [도전 과제 2] - 팔로우 / 팔로잉 목록 확인

- 팔로워/팔로잉 목록을 확인하는 페이지 추가
- 팔로워/팔로잉 목록의 유저는 링크로 작성
  - 해당 유저의 프로필 페이지로 이동할 수 있는 링크 추가

### [도전 과제 2] - 팔로우 / 팔로잉 목록 확인



### [도전 과제 3] - 대댓글 구현하기

- 대댓글 기능 구현
- 대댓글은 댓글이며, 다른 댓글을 참조
- 즉, 댓글과 대댓글의 차이점은 참조하는 댓글의 유무
  - 참조하는 댓글이 있다면 대댓글
  - 참조하는 댓글이 없다면 그냥 댓글
    - 참고) [null=True](#)
- 참고) [retrieving-specific-objects](#)

## [도전 과제 3] - 대댓글 구현하기

댓글 목록

진심 리얼로 재미 있습니다. [b](#)

◦ 이거 맞다

X

Content:  대댓글 달기

재미있어요 [Lorem](#) DELETE

◦ 찐 입니다.

X

Content:  대댓글 달기

강추!! [Lorem](#) DELETE

◦ 완전 강추!!

X

Content:  대댓글 달기

Content:  Submit

UPDATE DELETE

## | 선택 요구사항

- 명시된 요구사항 이외에는 자유롭게 작성해도 무관
- Bootstrap을 이용하여 자유롭게 스타일링 가능

# 제출



## 제출 시 주의사항

- 제출기한은 금일 18시까지 입니다. 제출기한을 지켜 주시기 바랍니다.
- 반드시 README.md 파일에 단계별로 구현 과정 중 학습한 내용, 어려웠던 부분, 새로 배운 것들 및 느낀 점을 등을 상세히 기록하여 제출합니다.
  - 단순히 완성된 코드만을 나열하지 않습니다.
- 위에 명시된 요구사항은 최소 조건이며, 추가 개발을 자유롭게 진행할 수 있습니다.
- <https://lab.ssafy.com/>에 프로젝트를 생성하고 제출합니다.
  - 프로젝트 이름은 '프로젝트 번호 + pjt'로 지정합니다. (ex. 01\_pjt)
- 반드시 각 반 담당 강사님을 Maintainer로 설정해야 합니다.