

# 06 PJT

# 관계형 데이터베이스 설계

# INDEX

---

- 개요
- 준비사항
- 요구사항
- 도전 과제
- 제출

# 개요

## 프로젝트 개요

- 커뮤니티 웹 서비스의 모델 관계 및 인증 시스템 구성 단계
- 영화 데이터의 생성, 조회, 수정, 삭제가 가능한 애플리케이션
- 로그인, 로그아웃, 회원가입이 가능한 애플리케이션
- 영화, 댓글, 회원 간의 모델 관계가 형성된 애플리케이션

## 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

- 명시된 요구사항 이외에는 자유롭게 작성해도 무관
- Bootstrap을 이용하여 자유롭게 스타일링 가능
- `.gitignore` 파일을 추가하여 불필요한 파일 및 폴더는 제출하지 않음

## | 필수 요구사항 - 프로젝트 및 앱

- 프로젝트의 이름
  - mypjt
- 앱 이름
  - movies
  - accounts

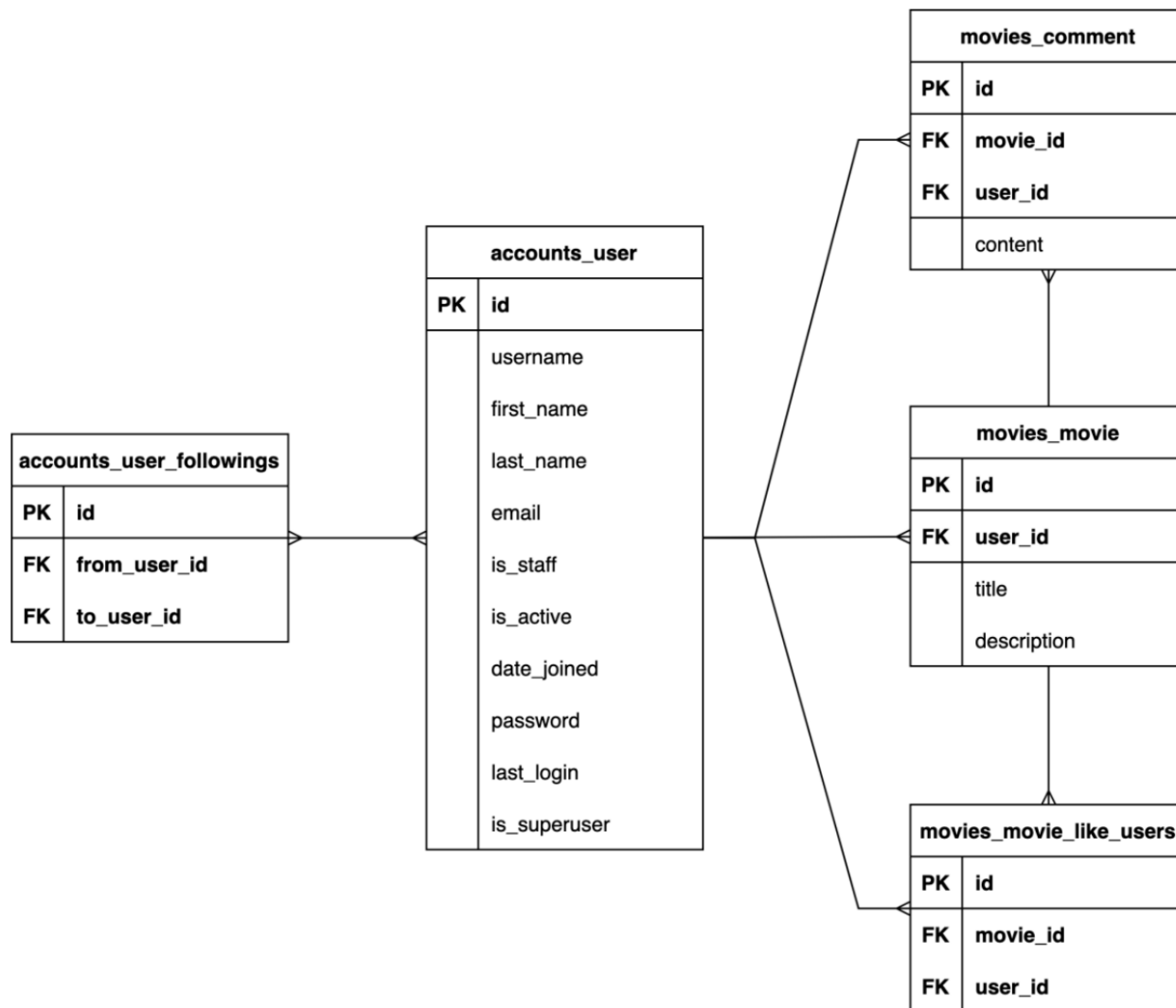
## 필수 요구사항 - 모델 클래스 (1/2)

- Movie 모델 클래스
  - 영화 제목, 줄거리, 작성일 및 수정일을 저장할 필드 4개 지정
  - 작성자 정보를 저장할 필드 1개 지정
  - 좋아요 누른 유저 정보를 저장하는 필드 1개 지정
- Comment 모델 클래스
  - 댓글 내용을 저장할 필드 1개 지정
  - 작성자 정보를 저장할 필드 1개 지정
  - 댓글이 작성된 영화 정보를 저장할 필드 1개 지정

## | 필수 요구사항 - 모델 클래스 (2/2)

- User 모델 클래스
  - AbstactUser 모델 클래스를 상속받는 커스텀 모델을 사용
  - 팔로우 유저 정보를 저장하는 필드 1개 지정

## 필수 요구사항 - 모델 관계



## 필수 요구사항 - view 함수

- movies 앱의 view 함수 (1/2)

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 detail로 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 detail로 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 index로 리다이렉트</li></ul>	POST

## 필수 요구사항 - view 함수

- movies 앱의 view 함수 (2/2)

함수명	역할	허용 HTTP Method
comments_create	<ul style="list-style-type: none"><li>유효성 검증 및 댓글 데이터 저장 후 detail로 리다이렉트</li></ul>	POST
comments_delete	<ul style="list-style-type: none"><li>단일 댓글 데이터 삭제 및 detail로 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 관계인 경우는 좋아요 취소)</li><li>좋아요 기능 동작 후 index로 리다이렉트</li></ul>	POST



## 필수 요구사항 - view 함수

- accounts 앱의 view 함수 (1/2)

함수명	역할	허용 HTTP Method
login	<ul style="list-style-type: none"><li>login.html 렌더링</li><li>로그인 절차 진행 후 index로 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 index로 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 index로 리다이렉트</li></ul>	GET & POST
delete	<ul style="list-style-type: none"><li>단일 회원 데이터 삭제 및 index로 리다이렉트</li></ul>	POST
change_password	<ul style="list-style-type: none"><li>change_password.html 렌더링</li><li>비밀번호 변경 후 index로 리다이렉트</li></ul>	GET & POST

## 필수 요구사항 - view 함수

- accounts 앱의 view 함수 (2/2)

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.html로 리다이렉트</li></ul>	POST

## | 필수 요구사항 - Form Class

- **Movie** 모델의 데이터 검증, 저장, 에러메시지 등을 모두 관리하기 위해 적절한 **ModelForm**을 사용
- **Comment** 모델의 데이터 검증, 저장, 에러메시지 등을 모두 관리하기 위해 적절한 **CommentForm**을 사용
- **User** 모델의 데이터 데이터 검증, 저장, 에러메시지 등을 모두 관리하기 위해 적절한 Built-in **Form**과 **ModelForm** 그리고 필요한 경우 커스텀 한 **ModelForm**을 사용

## | 필수 요구사항 - Admin

- Admin site에서 모델 Movie, Comment, User의 데이터 생성, 조회, 수정, 삭제가 가능해야 함

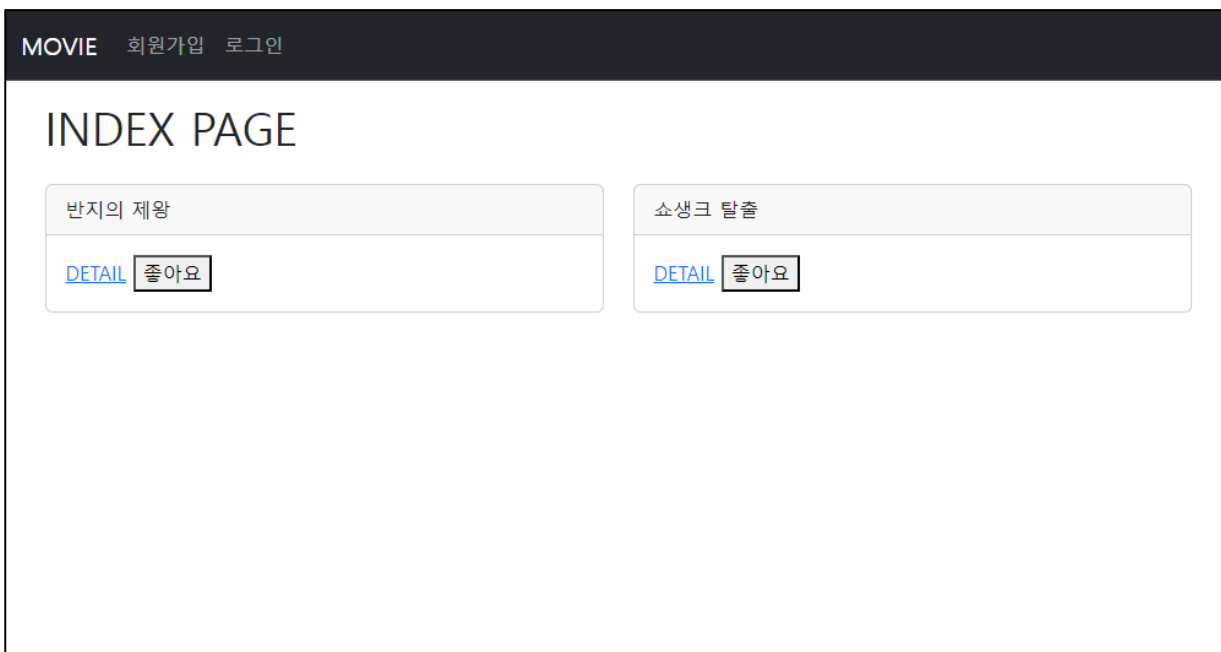
## | 필수 요구사항

- 본인이 작성한 데이터만 수정 및 삭제 가능
- 로그인 한 사용자만 데이터를 생성, 수정, 삭제 할 수 있음
- 로그인 한 사용자만 댓글을 생성, 삭제 할 수 있음
- 비로그인 사용자가 접속 할 수 있는 권한
  - 메인 페이지
  - 로그인 페이지
  - 영화 상세 정보 페이지
  - 회원가입 페이지

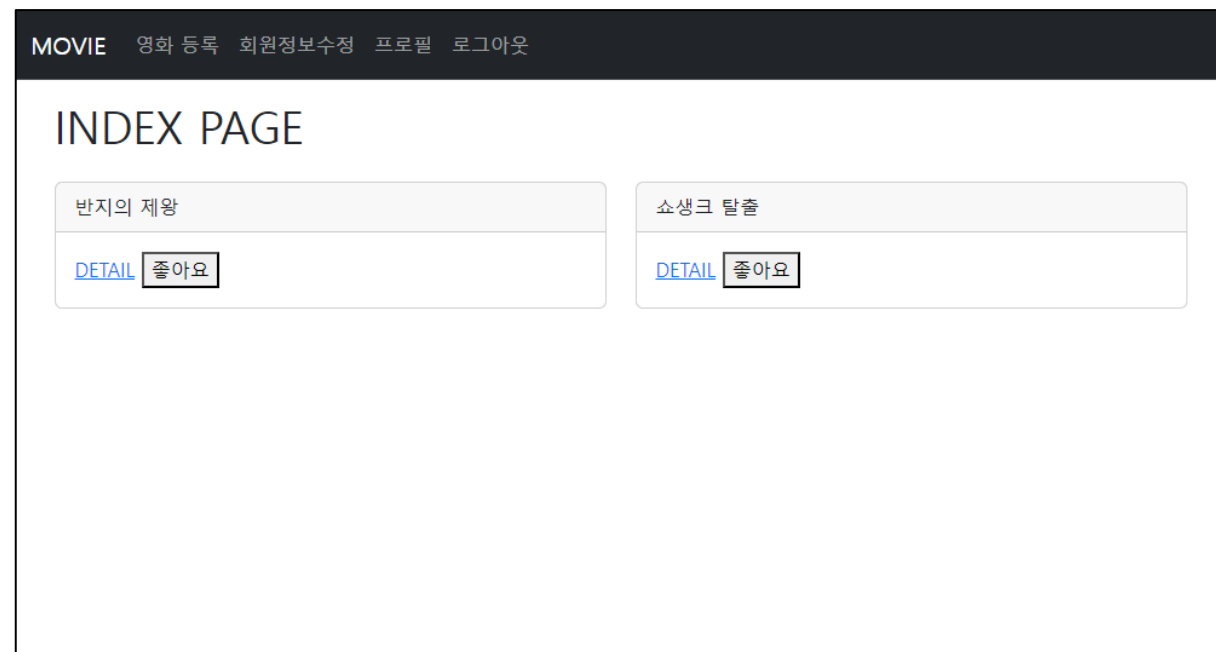
# 완성 화면 예시

## 완성 화면 예시 (1/9)

- 메인 페이지



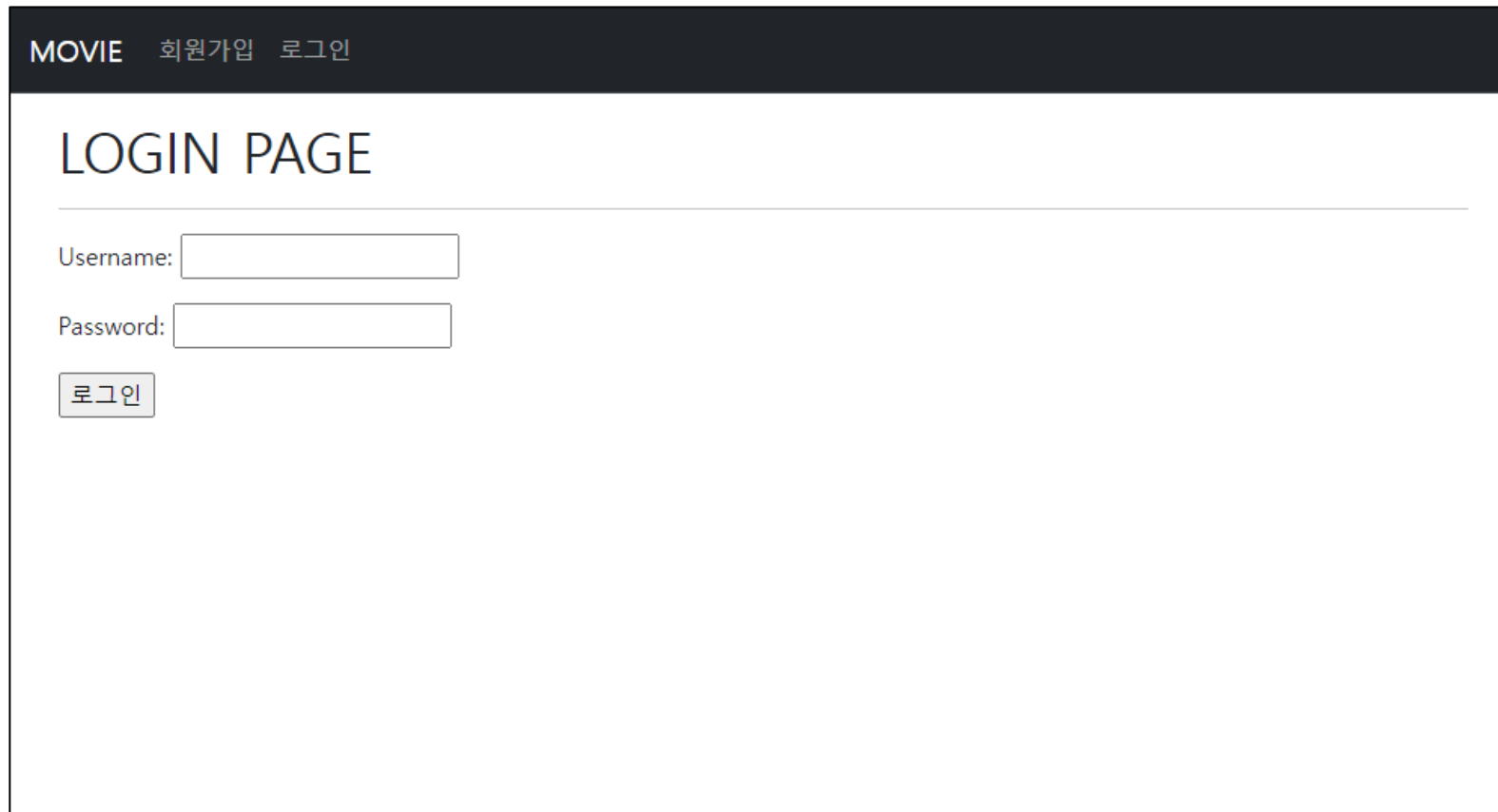
로그인 전



로그인 후

## 완성 화면 예시 (2/9)

- 로그인 페이지



The screenshot shows a web application interface for a movie site. At the top, a dark navigation bar contains the text "MOVIE" followed by links for "회원가입" (Sign Up) and "로그인" (Login). Below this, the main content area is titled "LOGIN PAGE". It features two input fields: "Username:" and "Password:". Below the password field is a button labeled "로그인" (Login).

MOVIE 회원가입 로그인

### LOGIN PAGE

Username:

Password:



## 완성 화면 예시 (3/9)

- 회원가입 페이지

[MOVIE](#) [회원가입](#) [로그인](#)

### SIGNUP PAGE

---

Username:  Required. 150 characters or fewer. Letters, digits and @/./+/-/\_ only.

Email address:

Password:

- Your password can't be too similar to your other personal information.
- Your password must contain at least 8 characters.
- Your password can't be a commonly used password.
- Your password can't be entirely numeric.

Password confirmation:  Enter the same password as before, for verification.

## 완성 화면 예시 (4/9)

- 영화 작성 페이지

The screenshot shows a web application interface for creating a movie. At the top, there is a dark navigation bar with the text "MOVIE" and several links: "영화 등록", "회원정보수정", "프로필", and "로그아웃". Below the navigation bar, the main content area has a heading "Create". Under "Create", there is a "Title:" label followed by a text input field. Below the title field is a large, empty rectangular area for the description. To the left of this area is a "Description:" label. At the bottom left of the form, there is a button labeled "제출" (Submit).

### 완성 화면 예시 (5/9)

- 개별 영화 상세 페이지
- 작성자 이름을 클릭하면 해당 작성자의 프로필 페이지로 이동

MOVIE
영화 등록
회원정보수정
프로필
로그아웃

DETAIL PAGE

반지의 제왕  
그의 《반지의 제왕》은 그 이후로 쓰여진 모든 다른 판타지를 뛰어넘어 현대 판타지를 형성한 산이다.

댓글 목록  
아직 댓글이 없네요...

Content:

[UPDATE](#)

[BACK](#)

댓글이 없을 때

MOVIE
영화 등록
회원정보수정
프로필
로그아웃

DETAIL PAGE

반지의 제왕  
그의 《반지의 제왕》은 그 이후로 쓰여진 모든 다른 판타지를 뛰어넘어 현대 판타지를 형성한 산이다.

댓글 목록

댓글 1
작성자 : [admin](#)

댓글 2
작성자 : [admin](#)

Content:

[UPDATE](#)

[BACK](#)

댓글이 있을 때

## 완성 화면 예시 (6/9)

- 영화 수정 페이지

MOVIE

영화 등록

회원정보수정

프로필

로그아웃

### Update

Title: 반지의 제왕

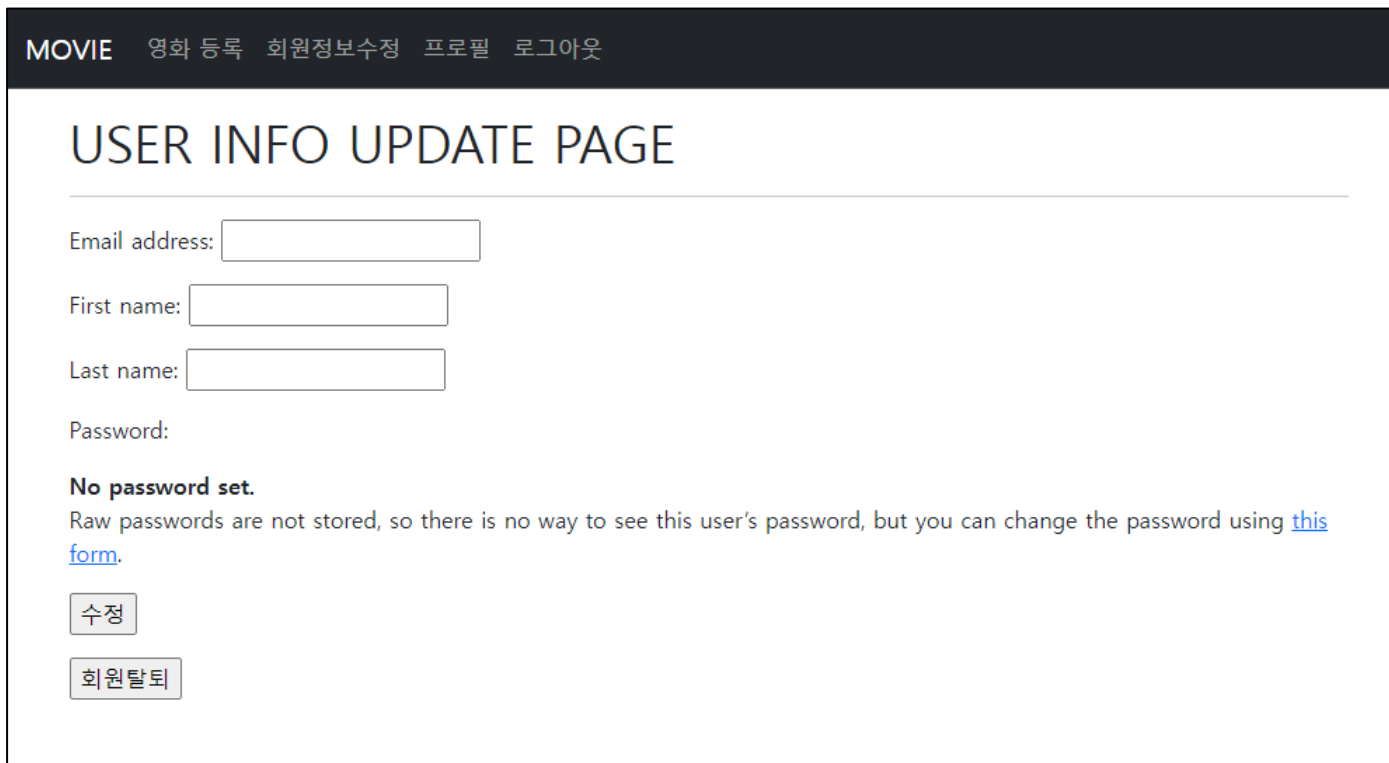
그의 《반지의 제왕》은 그 이후로 쓰여진 모든 다른 판타지를 뛰어넘어 현대 판타지를 형성한 산이다.

Description:

제출

## | 완성 화면 예시 (7/9)

- 회원 정보 수정 페이지
  - Django가 기본적으로 제공하는 비밀번호 변경 링크를 클릭하면 비밀번호 페이지로 이동



The screenshot shows a web browser window with a dark header bar containing the text "MOVIE" and navigation links: "영화 등록", "회원정보수정", "프로필", and "로그아웃". The main content area is titled "USER INFO UPDATE PAGE". It contains several input fields: "Email address:", "First name:", and "Last name:". Below these is a "Password:" label. A message states "No password set." followed by a note: "Raw passwords are not stored, so there is no way to see this user's password, but you can change the password using [this form](#)." At the bottom, there are two buttons: "수정" (Update) and "회원탈퇴" (Logout).

## 완성 화면 예시 (8/9)

- 비밀번호 변경 페이지

MOVIE

영화 등록

회원정보수정

프로필

로그아웃

CHANGE PASSWORD

Old password:

New password:

- Your password can't be too similar to your other personal information.
- Your password must contain at least 8 characters.
- Your password can't be a commonly used password.
- Your password can't be entirely numeric.

New password confirmation:

비밀번호 변경

## 완성 화면 예시 (9/9)

- 프로필 페이지
  - 프로필 유저의 팔로워/팔로잉 수 출력
  - 프로필 유저가 작성한 영화 목록과 좋아요를 누른 영화 목록 출력 (클릭 시 해당 영화 상세 페이지로 이동)



나의 프로필



남의 프로필

# 도전 과제



## 대댓글 기능 구현 - 요구사항

- 대댓글 생성과 삭제 기능 구현
- 로그인 한 사용자만 작성 할 수 있음
- 본인이 작성한 대댓글만 삭제 할 수 있음
- ❖ 댓글과 대댓글의 차이점은 참조하는 댓글의 유무
  - 참조하는 댓글이 없다면 댓글
  - 참조하는 댓글이 있다면 대댓글
    - 참고) [null=True](#)
- 참고) [retrieving-specific-objects](#)

## 대댓글 기능 구현 - 결과

[MOVIE](#) [영화 등록](#) [회원정보수정](#) [프로필](#) [로그아웃](#)

DETAIL PAGE

쇼생크 탈출  
희망은 좋은 거예요. 아마 가장 좋은 것일 거예요. 그리고 좋은 건 절대 사라지지 않아요.

댓글 목록

댓글 1 작성자 : [user1](#) DELETE

Content:  대댓글 달기

Content:  Submit

[UPDATE](#) DELETE

[BACK](#)

대댓글 작성 전

[MOVIE](#) [영화 등록](#) [회원정보수정](#) [프로필](#) [로그아웃](#)

DETAIL PAGE

쇼생크 탈출  
희망은 좋은 거예요. 아마 가장 좋은 것일 거예요. 그리고 좋은 건 절대 사라지지 않아요.

댓글 목록

댓글 1 작성자 : [user1](#) DELETE

- 대댓글 1 DELETE
- 대댓글 2 DELETE

Content:  대댓글 달기

Content:  Submit

[UPDATE](#) DELETE

[BACK](#)

대댓글 작성 후

## 대댓글 기능 구현 - 테이블 예시

id	content	main_comment_id	movie_id	user_id
1	댓글 1	NULL	1	1
2	댓글 2	NULL	1	1
3	댓글 1	NULL	2	2
4	대댓글 1	3	2	2
5	대댓글 2	3	2	2

# 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.