


Project Idea Selection: Movie Review API

Chosen Project: Movie Review API (Project Option 4)

Deployed Web APP Link: [Movie Review API](#)

<https://hashimaziz88.pythonanywhere.com/>

View the Slide Deck:  [Movie_Review_API Slides Hashim](#)

<https://docs.google.com/presentation/d/1bTdwqfRYv3OmdvDa6qfXhc6zR07JWXpJhn3UcswmrPA/edit?usp=sharing>

Reason for Choosing This Project

As an avid movie enthusiast and anime fan with a deep appreciation for cinema, particularly the works of directors like Christopher Nolan, I've chosen to develop a Movie Review API. This project aligns perfectly with my personal interests and provides an excellent opportunity to combine my passion for film with my growing skills in software development.

The Movie Review API will allow me to create a platform where users can search for movies, read and write reviews, and rate films. This project is not just a technical challenge but also a chance to build something I'm personally excited about and would use myself.

API to be Developed

The API I'll be developing will be a comprehensive Movie Review system with the following key features:

1. **Movie Search:** Utilize an external movie database API (such as OMDb) to allow users to search for movies.
2. **User Authentication:** Implement a secure user registration and login system.

3. **Review Management:** Allow authenticated users to create, read, update, and delete their movie reviews.
4. **Rating System:** Implement a rating system where users can rate movies on a scale (e.g., 1-5 stars).
5. **Movie Information Per Review:** Display detailed information about each Review and its associated movie and Review content.
6. **User Profiles:** Create user profiles to track review history and favorite movies.

Project Planning and Specifications

Movies

- **/search/** (GET):
List movies retrieved from OMDb.
 - **View:** `search_movie`
-

Reviews

- **/reviews/** (GET, POST):
 - **GET:** List all reviews.
 - **POST:** Create a new review.
 - **/reviews/<int:id>/** (GET, PUT, DELETE):
 - **GET:** Retrieve a specific review.
 - **PUT:** Update a specific review.
 - **DELETE:** Delete a specific review.
 - **/reviews/<int:id>/comment/** (POST):
Add a comment to a review.
 - **/reviews/<int:id>/like/** (POST):
Like a specific review.
-

Users

- **/api/user-profile/** (GET, POST):
 - **GET:** Retrieve user profile details.

- **POST:** Register a new user.
 - **/api/user-profile/edit/** (GET, PUT):
 - **GET:** Retrieve the profile details for editing.
 - **PUT:** Update the user profile.
-

Authentication

- **/register/** (POST):
User registration endpoint.
 - **View:** `register`
 - **/login/** (POST):
User login endpoint.
 - **View:** `LoginView`
 - **Options:**
 - `template_name='login.html'`
 - `redirect_authenticated_user=True`
 - `next_page='user-profile'`
 - **/logout/** (POST):
User logout endpoint.
 - **View:** `LogoutView`
 - **Options:**
 - `template_name='logout.html'`
 - `next_page='home'`
 - **/api/token/** (POST):
Obtain a JWT token for user authentication.
 - **/api/token/refresh/** (POST):
Refresh the JWT token.
-

Recommendations

- **/recommendations/** (GET):
Get personalized movie recommendations.

Development Phases:

1. Setup Django project and configure Django REST Framework
2. Design and implement database models
3. Implement user authentication system using `rest_framework_simplejwt`
4. Integrate external movie API (OMDb)
5. Develop core API functionality (CRUD operations for movies and reviews)
6. Implement additional features (rating system, user profiles)
7. Thorough testing and debugging
8. Deployment to PythonAnywhere
9. Documentation and final testing

This project will not only showcase my technical skills but also demonstrate my ability to create a fully functional, deployed application that serves a real purpose. The combination of working with external APIs, implementing authentication, and managing complex data relationships will provide valuable experience in building robust web applications.