Movie API Dokumentatsiyasi

Tohtarov Hosilbek Fn 28 guruh

Ushbu Dokument API’ning to‘liq tavsifi bo‘lib, bu koproq frontend chilar uchun va ushbu api dan barcha foydalanuvchilar uchun yartdim . Unda modellar, serializerlar, ruxsatlar, admin paneli, JWT autentifikatsiyasi va Swagger integratsiyasi keltirilgan.

# 1. Modellar

Quyida loyihadagi barcha modellar va ularning maydonlari keltiriladi.

## 1.1 User

Foydalanuvchi modeli, Django AbstractUser’dan meros oladi.

* Maydonlar:
* username: Foydalanuvchi nomi (CharField).
* email: Elektron pochta (EmailField).
* password: Parol (hash qilingan yani shifirlangan).
* first\_name: Ism (CharField).
* last\_name: Familiya (CharField).
* role: Rol (CharField, choices: admin yoki exploiter (exploiter-bu foydalanuvchilar)).

## 1.2 Admin

Admin foydalanuvchisi uchun profil.

* Maydonlar:
* user: User modeli bilan OneToOneField aloqa (related\_name="admin\_profile").
* phone\_number: Telefon raqami (CharField, max\_length=13).
* addres: Manzil (CharField, max\_length=255, tavsiya: address sifatida tuzatish).

## 1.3 Exploiter

Exploiter yani barcha foydalanuvchisi uchun profil.

* Maydonlar:
* user: User modeli bilan OneToOneField aloqa (related\_name="exploiter\_profile").
* phone\_number: Telefon raqami (CharField, max\_length=13).

## 1.4 Genre

Kino janri modeli.

* Maydonlar:
* name: Janr nomi (CharField, max\_length=100).

## 1.5 Company

Kino ishlab chiqaruvchi kompaniya modeli.

* Maydonlar:
* name: Kompaniya nomi (CharField, max\_length=100).
* description: Tavsif (TextField, ixtiyoriy).

## 1.6 Actor

Aktyor modeli.

* Maydonlar:
* first\_name: Ism (CharField, max\_length=100).
* last\_name: Familiya (CharField, max\_length=100).
* birth\_date: Tug‘ilgan sana (DateField, ixtiyoriy).
* photo: Rasm (ImageField, ixtiyoriy).

## 1.7 Film

Kino modeli.

* Maydonlar:
* name: Kino nomi (CharField, max\_length=150).
* views: Ko‘rishlar soni (PositiveIntegerField, default=0).
* genre: Janr bilan ForeignKey aloqa (SET\_NULL).
* company: Kompaniya bilan ForeignKey aloqa (SET\_NULL).
* actors: Aktyorlar bilan ManyToManyField aloqa.

## 1.8 Comment

Foydalanuvchi izohlari modeli.

* Maydonlar:
* user: Foydalanuvchi bilan ForeignKey aloqa.
* film: Kino bilan ForeignKey aloqa.
* text: Izoh matni (TextField).
* created\_at: Yaratilgan vaqt (DateTimeField, auto\_now\_add=True).

## 1.9 Rating

Kino baholash modeli bu maxsilam 10 dan minemal 1 gacha baholay oladi

* Maydonlar:
* user: Foydalanuvchi bilan ForeignKey aloqa.
* film: Kino bilan ForeignKey aloqa.
* score: Baho (PositiveSmallIntegerField, 1-10).
* created\_at: Yaratilgan vaqt (DateTimeField, auto\_now\_add=True).
* Meta: user va film birikmasi unik (unique\_together).

# 2. Serializerlar

Serializerlar ma’lumotlarni JSON formatiga to’g’rirog’i esa dict malumot turiga o‘tkazish va validatsiya qilish uchun ishlatiladi. Quyida har bir serializerning maqsadi va logikasi, ayniqsa yangilangan funksionallik keltiriladi.

## 2.1 UserSerializer

Foydalanuvchi ma’lumotlarini serializatsiya qiladi.

Maydonlar: id, username, email, password (write\_only), first\_name, last\_name, role

* Create logikasi:
* create\_user metodi orqali yangi foydalanuvchi yaratiladi.
* Parol hash qilinadi.
* role maydoni default sifatida "exploiter" qabul qilinadi.
* Update logikasi:
* Mavjud foydalanuvchi ma’lumotlari yangilanadi.
* Agar parol berilgan bo‘lsa, set\_password orqali hash qilinadi.

## 2.2 AdminSerializer

Admin profilini serializatsiya qiladi, user maydoni nested UserSerializer orqali.

Maydonlar: id, user, phone\_number, addres

* Create logikasi:
* user ma’lumotlari UserSerializer orqali yangi User yaratadi (role="admin").
* Admin obyekti yaratiladi va user bilan bog‘lanadi.
* Update logikasi:
* user ma’lumotlari UserSerializer orqali yangilanadi (role="admin" saqlanadi).
* phone\_number va addres yangilanadi.

## 2.3 ExploiterSerializer

Exploiter profilini serializatsiya qiladi, user maydoni nested UserSerializer orqali.

Maydonlar: id, user, phone\_number

* Create logikasi:
* user ma’lumotlari UserSerializer orqali yangi User yaratadi (role="exploiter").
* Exploiter obyekti yaratiladi va user bilan bog‘lanadi.
* Update logikasi:
* user ma’lumotlari UserSerializer orqali yangilanadi (role="exploiter" saqlanadi).
* phone\_number yangilanadi.

## 2.4 GenreSerializer

Kino janrini serializatsiya qiladi. Yangilanish: yangi janr yaratish yoki mavjudini ishlatish.

Maydonlar: id, name

* Create logikasi:
* Agar berilgan nomdagi janr mavjud bo‘lsa, uni qaytaradi (get\_or\_create).
* Aks holda, yangi janr yaratadi.

## 2.5 CompanySerializer

Kompaniya ma’lumotlarini serializatsiya qiladi. Yangilanish: yangi kompaniya yaratish yoki mavjudini ishlatish.

Maydonlar: id, name, description

* Create logikasi:
* Agar berilgan nomdagi kompaniya mavjud bo‘lsa, uni qaytaradi (get\_or\_create).
* Aks holda, yangi kompaniya yaratadi, description ixtiyoriy.

## 2.6 ActorSerializer

Aktyor ma’lumotlarini serializatsiya qiladi. Yangilanish: yangi aktyor yaratish yoki mavjudini ishlatish.

Maydonlar: id, first\_name, last\_name, birth\_date, photo

* Create logikasi:
* Agar first\_name va last\_name bo‘yicha aktyor mavjud bo‘lsa, uni qaytaradi (get\_or\_create).
* Aks holda, yangi aktyor yaratadi, birth\_date va photo ixtiyoriy.

## 2.7 FilmSerializer

Kino ma’lumotlarini serializatsiya qiladi. Yangilanish: genre, company, actors uchun nested serializerlar va yangi/mavjud obyektlar bilan ishlash.

Maydonlar: id, name, views, genre, company, actors

* Create logikasi:
* Agar genre uchun JSON ma’lumot kelsa (masalan, {"name": "Sci-Fi"}), yangi Genre yaratiladi.
* Agar genre\_id kelsa, mavjud janr ishlatiladi.
* company va actors uchun xuddi shunday: JSON ma’lumot yoki company\_id/actor\_ids ishlatiladi.
* Film obyekti yaratiladi, actors ManyToManyField orqali qo‘shiladi.
* Update logikasi:
* Mavjud film yangilanadi, genre, company, actors uchun yangi ma’lumot yoki ID’lar ishlatiladi.
* Agar ma’lumot kelmagan bo‘lsa, eski qiymatlar saqlanadi.

## 2.8 CommentSerializer

Izoh ma’lumotlarini serializatsiya qiladi.

Maydonlar: id, user (read\_only), film (read\_only), text, created\_at

* Create logikasi:
* user joriy foydalanuvchidan (request.user) olinadi.
* film\_id request.data dan olinadi.
* Comment obyekti yaratiladi.

## 2.9 RatingSerializer

Baho ma’lumotlarini serializatsiya qiladi.

Maydonlar: id, user (read\_only), film (read\_only), score, created\_at

* Create logikasi:
* user joriy foydalanuvchidan (request.user) olinadi.
* film\_id request.data dan olinadi.
* Rating obyekti yaratiladi.
* Update logikasi:
* Faqat score maydoni yangilanadi.

# 3. Maxsus Ruxsatlar

Standart ruxsatlar (IsAdminUser, IsAuthenticated, AllowAny) o‘rniga permissions.py faylida maxsus ruxsat sinflari yaratildi.

## 3.1 IsAdmin

Faqat role="admin" foydalanuvchilar uchun ruxsat beradi.

* Qo‘llanilishi: User va Admin ViewSet’larida.

## 3.2 IsAuthenticatedOrExploiter

Autentifikatsiya qilingan foydalanuvchilar uchun ochiq, lekin exploiter faqat o‘z profilini ko‘radi.

* Logika:
* has\_object\_permission orqali exploiter faqat o‘z user obyekti bilan bog‘langan profilni ko‘radi.
* Adminlar barcha exploter profillarini ko‘rishi mumkin.
* Qo‘llanilishi: ExploiterViewSet.

## 3.3 AllowReadOnlyForNonAdmins

GET so‘rovlari uchun hamma uchun ochiq, yozish (POST, PUT, DELETE) faqat adminlar uchun.

* Qo‘llanilishi: Genre, Company, Actor, Film ViewSet’larida.

## 3.4 IsAuthenticatedForComments

Autentifikatsiya qilingan foydalanuvchilar izoh yozishi mumkin, lekin faqat o‘z izohlarini ko‘radi.

* Logika:
* has\_object\_permission orqali foydalanuvchi faqat o‘z izohlarini ko‘radi.
* Adminlar barcha izohlarni ko‘rishi mumkin.
* Qo‘llanilishi: CommentViewSet.

## 3.5 IsAuthenticatedForRatings

Autentifikatsiya qilingan foydalanuvchilar baho qo‘yishi mumkin, lekin faqat o‘z baholarini ko‘radi.

* Logika:
* has\_object\_permission orqali foydalanuvchi faqat o‘z baholarini ko‘radi.
* Adminlar barcha baholarni ko‘rishi mumkin.
* Qo‘llanilishi: RatingViewSet.

# 4. ViewSet’lar va URL’lar

Har bir model uchun ViewSet va unga mos keluvchi API endpointlari.

## 4.1 UserViewSet

Foydalanuvchilarni boshqarish uchun.

Ruxsatlar: IsAdmin (faqat adminlar).

URL: http://127.0.0.1:8000/app/v1/users/

## 4.2 AdminViewSet

Admin profillarini boshqarish uchun.

Ruxsatlar: IsAdmin (faqat adminlar).

URL: <http://127.0.0.1:8000/app/v1/admins/>

## 4.3 ExploiterViewSet

Exploiter profillarini boshqarish uchun.

Ruxsatlar: IsAuthenticatedOrExploiter (autentifikatsiya qilingan foydalanuvchilar, faqat o‘z profilini ko‘radi).

Logika: Oddiy foydalanuvchilar faqat o‘z profilini ko‘radi, adminlar hammasini.

URL: http://127.0.0.1:8000/app/v1/exploiters/

## 4.4 GenreViewSet

Janrlarni boshqarish uchun.

Ruxsatlar: AllowReadOnlyForNonAdmins (hamma ko‘rishi mumkin, faqat adminlar yozishi mumkin).

URL: http://127.0.0.1:8000/app/v1/genres/

## 4.5 CompanyViewSet

Kompaniyalarni boshqarish uchun.

Ruxsatlar: AllowReadOnlyForNonAdmins.

URL: http://127.0.0.1:8000/app/v1/companies/

## 4.6 ActorViewSet

Aktyorlarni boshqarish uchun.

Ruxsatlar: AllowReadOnlyForNonAdmins.

URL: http://127.0.0.1:8000/app/v1/actors/

## 4.7 FilmViewSet

Kinolar boshqarish uchun.

Ruxsatlar: AllowReadOnlyForNonAdmins.

URL: http://127.0.0.1:8000/app/v1/films/

## 4.8 CommentViewSet

Izohlarni boshqarish uchun.

Ruxsatlar: IsAuthenticatedForComments (autentifikatsiya qilingan foydalanuvchilar, faqat o‘z izohlarini ko‘radi).

Logika: Foydalanuvchilar faqat o‘z izohlarini ko‘radi, adminlar hammasini.

URL: http://127.0.0.1:8000/app/v1/comments/

## 4.9 RatingViewSet

Baholarni boshqarish uchun.

Ruxsatlar: IsAuthenticatedForRatings (autentifikatsiya qilingan foydalanuvchilar, faqat o‘z baholarini ko‘radi).

Logika: Foydalanuvchilar faqat o‘z baholarini ko‘radi, adminlar hammasini.

URL: http://127.0.0.1:8000/app/v1/ratings/

# 5. Admin Paneli

Admin paneli har bir model uchun chiroyli va qulay interfeys bilan sozlandi. Har bir model uchun ModelAdmin sinflari yaratildi.

## 5.1 UserAdmin

Foydalanuvchilarni boshqarish uchun.

* Sozlamalar:
* list\_display: username, email, role, first\_name, last\_name, is\_staff.
* list\_filter: role, is\_staff, is\_active.
* search\_fields: username, email, first\_name, last\_name.
* fields: username, email, password, first\_name, last\_name, role, is\_staff, is\_active.

## 5.2 AdminAdmin

Admin profillarini boshqarish uchun.

* Sozlamalar:
* list\_display: user, phone\_number, addres.
* list\_filter: phone\_number.
* search\_fields: user\_\_username, phone\_number, addres.
* raw\_id\_fields: user.

## 5.3 ExploiterAdmin

Exploiter profillarini boshqarish uchun.

* Sozlamalar:
* list\_display: user, phone\_number.
* list\_filter: phone\_number.
* search\_fields: user\_\_username, phone\_number.
* raw\_id\_fields: user.

## 5.4 GenreAdmin

Janrlarni boshqarish uchun.

* Sozlamalar:
* list\_display: name.
* search\_fields: name.
* fields: name.

## 5.5 CompanyAdmin

Kompaniyalarni boshqarish uchun.

* Sozlamalar:
* list\_display: name, description.
* search\_fields: name, description.
* fields: name, description.

## 5.6 ActorAdmin

Aktyorlarni boshqarish uchun.

* Sozlamalar:
* list\_display: first\_name, last\_name, birth\_date.
* list\_filter: birth\_date.
* search\_fields: first\_name, last\_name.
* fields: first\_name, last\_name, birth\_date, photo.

## 5.7 FilmAdmin

Kinolar boshqarish uchun.

* Sozlamalar:
* list\_display: name, genre, company, views.
* list\_filter: genre, company.
* search\_fields: name.
* fields: name, genre, company, actors, views.
* filter\_horizontal: actors (ManyToManyField uchun qulay interfeys).
* raw\_id\_fields: genre, company.

## 5.8 CommentAdmin

Izohlarni boshqarish uchun.

* Sozlamalar:
* list\_display: user, film, text\_preview, created\_at.
* list\_filter: created\_at, user, film.
* search\_fields: user\_\_username, film\_\_name, text.
* fields: user, film, text, created\_at.
* raw\_id\_fields: user, film.
* readonly\_fields: created\_at.
* text\_preview: Izoh matnining qisqa ko‘rinishi.

## 5.9 RatingAdmin

Baholarni boshqarish uchun.

* Sozlamalar:
* list\_display: user, film, score, created\_at.
* list\_filter: score, created\_at, user, film.
* search\_fields: user\_\_username, film\_\_name.
* fields: user, film, score, created\_at.
* raw\_id\_fields: user, film.
* readonly\_fields: created\_at.

# 6. JWT Autentifikatsiyasi

API’da autentifikatsiya uchun djangorestframework-simplejwt kutubxonasi ishlatildi. Bu foydalanuvchilarga access va refresh tokenlari orqali kirish imkonini beradi.

* Sozlamalar:
* settings.py’da REST\_FRAMEWORK va SIMPLE\_JWT sozlamalari qo‘shildi.
* Access token muddati: 60 daqiqa.
* Refresh token muddati: 1 kun.
* Token olish: POST /api/token/ (username va password bilan).
* Token yangilash: POST /api/token/refresh/ (refresh token bilan).
* Swagger’da token kiritish: Bearer <access\_token> formatida.

# 7. Swagger Integratsiyasi

API dokumentatsiyasi uchun drf-yasg kutubxonasi ishlatildi, bu Swagger UI va ReDoc interfeyslarini taqdim etadi.

* Sozlamalar:
* settings.py’da drf\_yasg INSTALLED\_APPS’ga qo‘shildi.
* urls.py’da /swagger/ va /redoc/ endpoint’lari qo‘shildi.
* Swagger UI: http://127.0.0.1:8000/swagger/
* ReDoc: http://127.0.0.1:8000/redoc/
* JWT integratsiyasi: SWAGGER\_SETTINGS’da Bearer token sozlamasi qo‘shildi.
* Foydalanish: “Authorize” tugmasi orqali Bearer <access\_token> kiritiladi.

# 8. Foydalanuvchilar (User, Admin, Exploiter)

Foydalanuvchilar bilan ishlashda quyidagi ishlar amalga oshirildi:

* User modeli:
* AbstractUser’dan meros oladi, role maydoni qo‘shildi (admin yoki exploiter).
* JWT autentifikatsiyasi orqali foydalanuvchilar login qiladi.
* Admin modeli:
* User bilan OneToOneField orqali bog‘langan.
* AdminSerializer orqali yangi User (role="admin") va Admin obyekti yaratiladi.
* Update’da user va admin ma’lumotlari alohida yangilanadi.
* Exploiter modeli:
* User bilan OneToOneField orqali bog‘langan.
* ExploiterSerializer orqali yangi User (role="exploiter") va Exploiter obyekti yaratiladi.
* Update’da user va exploiter ma’lumotlari alohida yangilanadi.
* Ruxsatlar:
* User va Admin faqat adminlar tomonidan boshqariladi (IsAdmin).
* Exploiter autentifikatsiya qilingan foydalanuvchilar tomonidan ko‘riladi (IsAuthenticatedOrExploiter).

# 9. Frontendchilar uchun Ko‘rsatmalar

Ushbu API frontend ishlab chiquvchilar uchun mo‘ljallangan bo‘lib, quyidagi ko‘rsatmalar bilan ishlaydi:

* API Endpoint’lari:
* Barcha endpoint’lar http://127.0.0.1:8000/app/v1/ prefiksi bilan ishlaydi.
* Masalan: /app/v1/films/, /app/v1/comments/, /app/v1/ratings/.
* JWT Autentifikatsiyasi:
* POST /api/token/ orqali access va refresh tokenlari olinadi.
* So‘rovlarda Authorization header’ida Bearer <access\_token> ishlatiladi.
* Access token muddati tugasa, POST /api/token/refresh/ orqali yangilanadi.
* Swagger UI:
* http://127.0.0.1:8000/swagger/ orqali barcha endpoint’larni sinab ko‘rish mumkin.
* “Authorize” tugmasi orqali Bearer token kiritiladi.
* Maslahatlar:
* Tokenlarni xavfsiz saqlang (masalan, HttpOnly cookie’larda).
* So‘rovlar uchun Content-Type: application/json ishlatiladi.
* Xatolarni tekshirish uchun Swagger’dagi response kodlarini ko‘ring (masalan, 401 Unauthorized).