**Министерство транспорта Российской Федерации**

**Федеральное государственное автономное образовательное  
учреждение высшего образования  
«РУТ (МИИТ)»**

**Институт транспортной техники и систем управления**

**Кафедра «Управление и защита информации»**

КУРСОВОЙ ПРОЕКТ

по дисциплине

**«Основы построения защищенных баз данных»**

**на тему  
«Фильмотека»**

Выполнил: ст. гр. ТКИ-541  
Соснин М.А.  
Проверил: Васильева М.А.

Москва 2022

**Цель работы:**

Научится использовать структуру ОРМ на любом языке программирования.

**Постановка задачи:**

Написать на языке Python с использованием фреймворка Django фильмотеку. Должна быть возможность хранить фильмы, серии фильмов, субтитры и дорожки озвучек.

**Выполнение работы:**

**models.py**

from django.db import models  
*# from django.contrib.auth.models import User*from django.contrib.auth.models import AbstractUser  
  
  
class User(AbstractUser):  
 gender = models.BooleanField(verbose\_name="пол", default=None, blank=True, null=True)  
 avatar = models.ImageField(upload\_to='avatars/', default="avatars/default\_user.png")  
  
 def \_\_str\_\_(self):  
 return self.username  
  
  
class Genre(models.Model):  
 name = models.CharField('Жанр', max\_length=150)  
  
 def \_\_str\_\_(self):  
 return self.name  
  
 class Meta:  
 verbose\_name = 'Жанр'  
 verbose\_name\_plural = 'Жанры'  
  
  
class Film(models.Model):  
 name = models.TextField('Название фильма', default="Название")  
 country = models.CharField('Страна', max\_length=150, default="США")  
 seasons = models.CharField('Количество сезонов', max\_length=50, default="1")  
 filmtype = models.CharField('Тип', max\_length=20, default="Фильм")  
 year = models.CharField('Год выпуска', max\_length=10, default="2020")  
 description = models.TextField('Описаное', default="НЕТУ")  
 genre = models.ManyToManyField(Genre, help\_text="Выберите жанр для фильма", verbose\_name="Жанр", blank=True)  
 img = models.ImageField(upload\_to="images/films", default="static/mult/image/default.jpg", blank=True, null=True)  
 img\_url = models.TextField('ссылка на исходную картинку')  
 unformated\_name = models.TextField("Имя папки", default=name)  
 mult = models.BooleanField("Является ли мультиком", default=False)  
 isShown = models.BooleanField("Показывать ли на странице", default=True)  
 create\_date = models.DateTimeField("Дата создания", auto\_now\_add=True)  
 likes = models.ManyToManyField(User, blank=True, related\_name='film\_likes')  
 dislikes = models.ManyToManyField(User, blank=True, related\_name='film\_dislikes')  
  
 def \_\_str\_\_(self):  
 return self.name  
  
 class Meta:  
 verbose\_name = 'Фильм'  
 verbose\_name\_plural = 'Фильмы'  
 ordering = ['name']  
  
  
class SeriesFilms(models.Model):  
 name = models.ForeignKey(Film, on\_delete=models.CASCADE)  
 name\_serie = models.TextField('Название серии')  
 href = models.TextField('Ссылка на серию')  
 full\_name = models.TextField('Полное название серии')  
  
 def \_\_str\_\_(self):  
 return self.name\_serie  
  
 class Meta:  
 verbose\_name = 'Серия'  
 verbose\_name\_plural = 'Серии Фильмов'  
 ordering = ['name\_serie']  
  
  
class Mult(models.Model):  
 name = models.TextField('Название мультика')  
 episodes = models.TextField('Количество эпизодов')  
 status = models.TextField('Вышло или нет')  
 description = models.TextField('Описаное')  
 img\_url = models.TextField('Картинка')  
 img = models.ImageField(upload\_to="images/mults", default="static/mult/image/default.jpg", blank=True, null=True)  
 genre = models.ManyToManyField(Genre, help\_text="Выберите жанр для мультика", verbose\_name="Жанр", blank=True)  
 unformated\_name = models.TextField("Имя папки")  
 mult = models.BooleanField("Является ли мультиком", default=True)  
 isShown = models.BooleanField("Показывать ли на странице", default=True)  
 create\_date = models.DateTimeField("Дата создания", auto\_now\_add=True)  
 likes = models.ManyToManyField(User, blank=True, related\_name='mult\_likes')  
 dislikes = models.ManyToManyField(User, blank=True, related\_name='mult\_dislikes')  
  
 def \_\_str\_\_(self):  
 return self.name  
  
 class Meta:  
 verbose\_name = 'Мультфильм'  
 verbose\_name\_plural = 'Мультфильмы'  
 ordering = ['name']  
  
  
class Series(models.Model):  
 name = models.ForeignKey(Mult, on\_delete=models.CASCADE)  
 name\_serie = models.TextField('Название серии')  
 href = models.TextField('Ссылка на серию')  
 full\_name = models.TextField('Полное название серии')  
  
 def \_\_str\_\_(self):  
 return self.name\_serie  
  
 class Meta:  
 verbose\_name = 'Серия'  
 verbose\_name\_plural = 'Серии Мультфильмов'  
 ordering = ['name\_serie']  
  
  
class Subs(models.Model):  
 name = models.ForeignKey(Series, on\_delete=models.CASCADE)  
 mult = models.ForeignKey(Mult, on\_delete=models.CASCADE)  
 name\_sub = models.TextField('Название субтитра')  
 autor = models.TextField('Автор субтитров')  
 href = models.TextField('Ссылка на субтитр')  
  
 def \_\_str\_\_(self):  
 return self.name\_sub  
  
 class Meta:  
 verbose\_name = 'Субтитры'  
 verbose\_name\_plural = 'Субтитры Мультфильмов'  
 ordering = ['name\_sub']  
  
  
class Audio(models.Model):  
 name = models.ForeignKey(Series, on\_delete=models.CASCADE)  
 mult = models.ForeignKey(Mult, on\_delete=models.CASCADE)  
 name\_audio = models.TextField('Название озвучки')  
 autor = models.TextField('Автор озвучки')  
 href = models.TextField('Ссылка на озвучку')  
  
 def \_\_str\_\_(self):  
 return self.name\_audio  
  
 class Meta:  
 verbose\_name = 'Озвучка'  
 verbose\_name\_plural = 'Озвучка Мультфильмов'  
 ordering = ['name\_audio']

**Описание классов:**

***Film***

|  |  |  |
| --- | --- | --- |
| *Название поля* | *Тип данных* | *Обязательное* |
| *name* | *TextField* | *Да* |
| *country* | *CharField(150)* | *Да* |
| *seasons* | *CharField(50)* | *Да* |
| *filmtype* | *CharField(20)* | *Да* |
| *year* | *CharField(10)* | *Да* |
| *description* | *TextField* | *Да* |
| *genre* | *ManyToManyField(Genre)* | *Нет* |
| *img* | *ImageField* | *Нет* |
| *img\_url* | *TextField* | *Да* |
| *unformated\_name* | *TextField* | *Да* |
| *mult* | *BooleanField* | *Да* |
| *isShown* | *BooleanField* | *Да* |
| *create\_date* | *DateTimeField* | *Нет(Авто)* |
| *likes* | *ManyToManyField(User)* | *Нет* |
| *dislikes* | *ManyToManyField(User)* | *Нет* |

***SeriesFilms***

|  |  |  |
| --- | --- | --- |
| *Название поля* | *Тип данных* | *Обязательное* |
| *name* | *ForeignKey(Film)* | *Да* |
| *name\_serie* | *CharField(150)* | *Да* |
| *href* | *CharField(50)* | *Да* |
| *full\_name* | *CharField(20)* | *Да* |

***Genre***

|  |  |  |
| --- | --- | --- |
| *Название поля* | *Тип данных* | *Обязательное* |
| *name* | *CharField(150)* | *Да* |

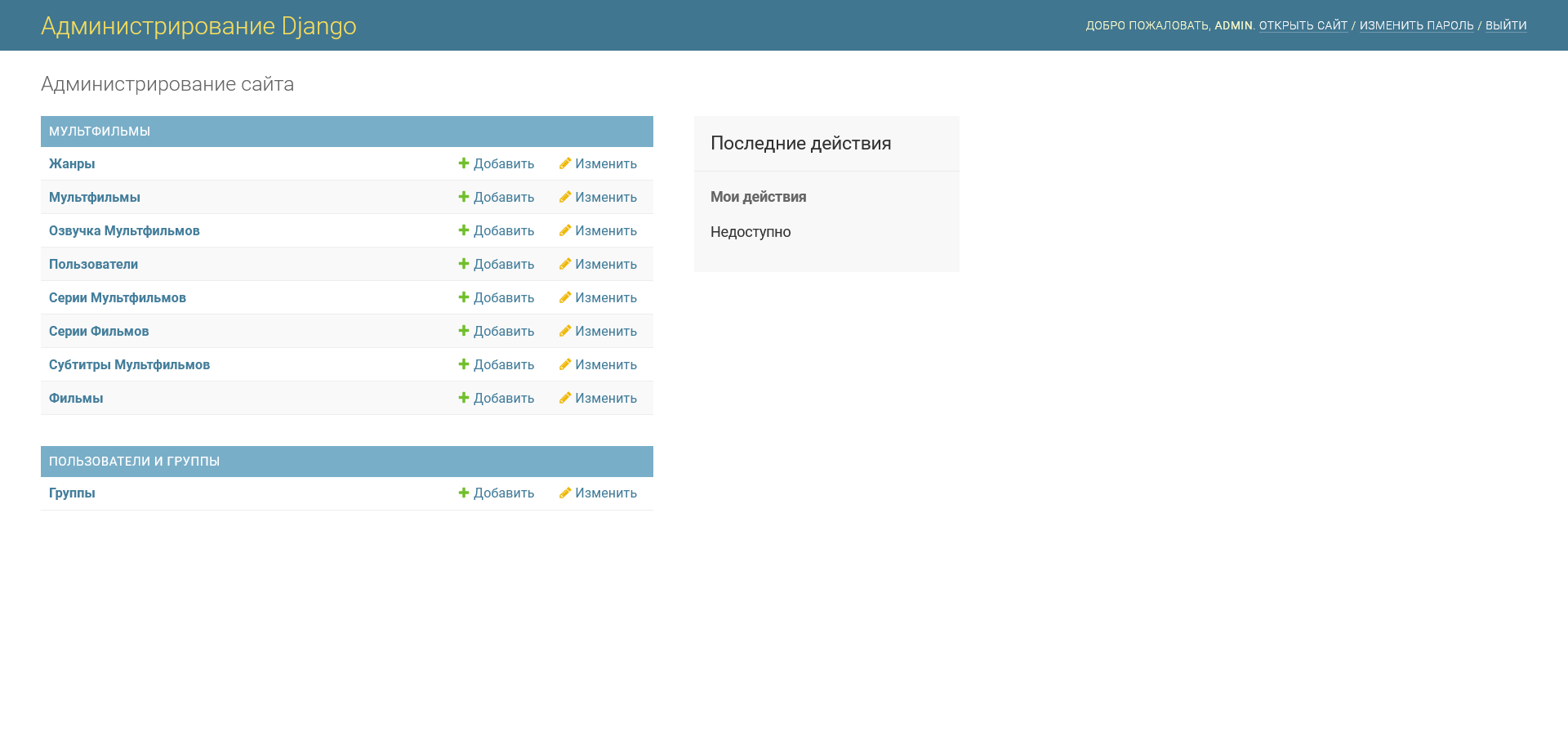
***Subs***

|  |  |  |
| --- | --- | --- |
| *Название поля* | *Тип данных* | *Обязательное* |
| *name* | *ForeignKey(Series)* | *Да* |
| *mult* | *ForeignKey(Mult)* | *Да* |
| *name\_sub* | *TextField* | *Да* |
| *autor* | *TextField* | *Да* |
| *href* | *TextField* | *Да* |

**Скриншоты работы**

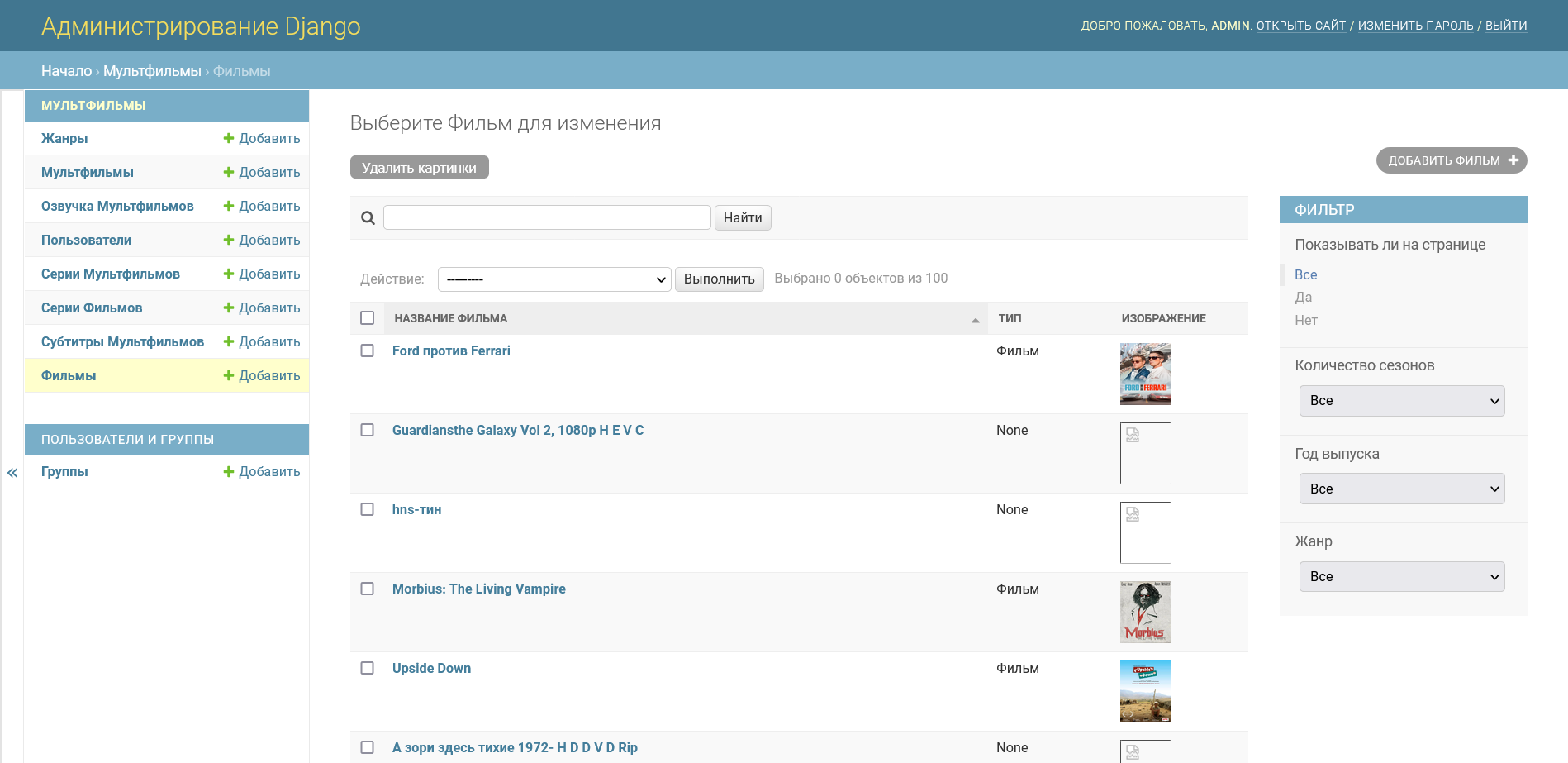
После создания классов в администрировании Django стало доступно изменение и добавления элементов в базу данных.

**Изменение данных**

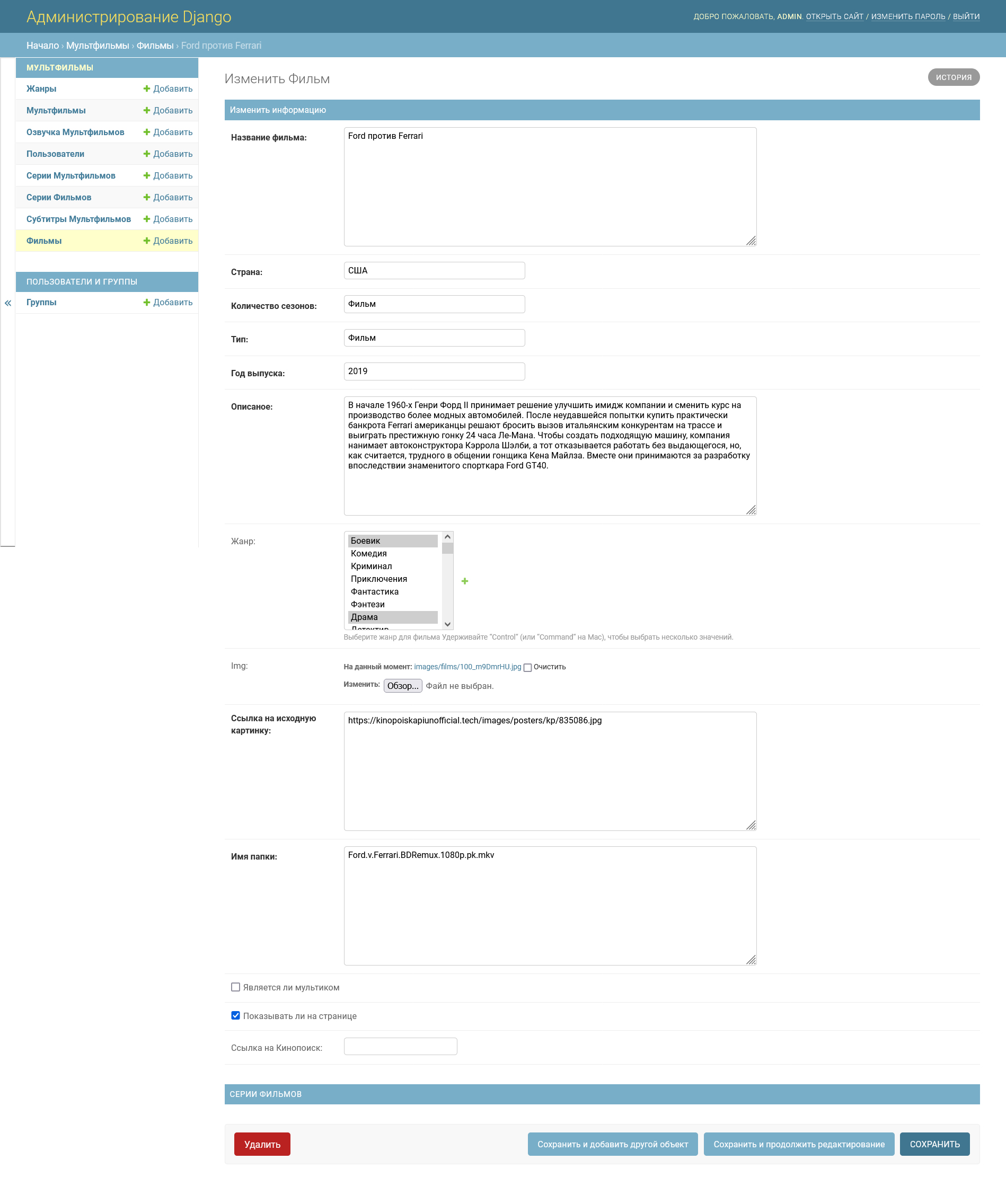


*Рисунок 1 страница администратора*

**Изменение фильмов**

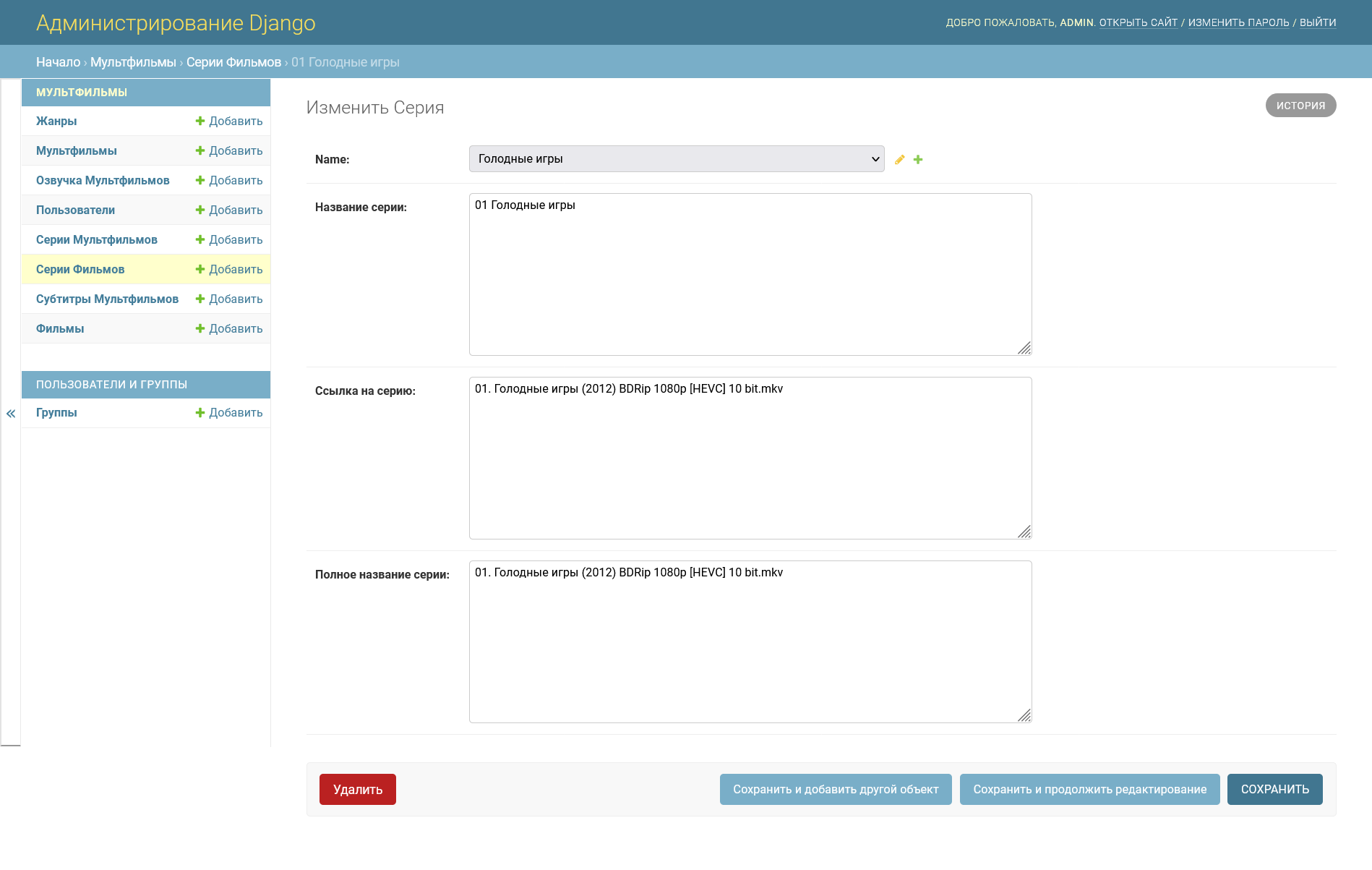


*Рисунок 2 Страница изменения фильмов*

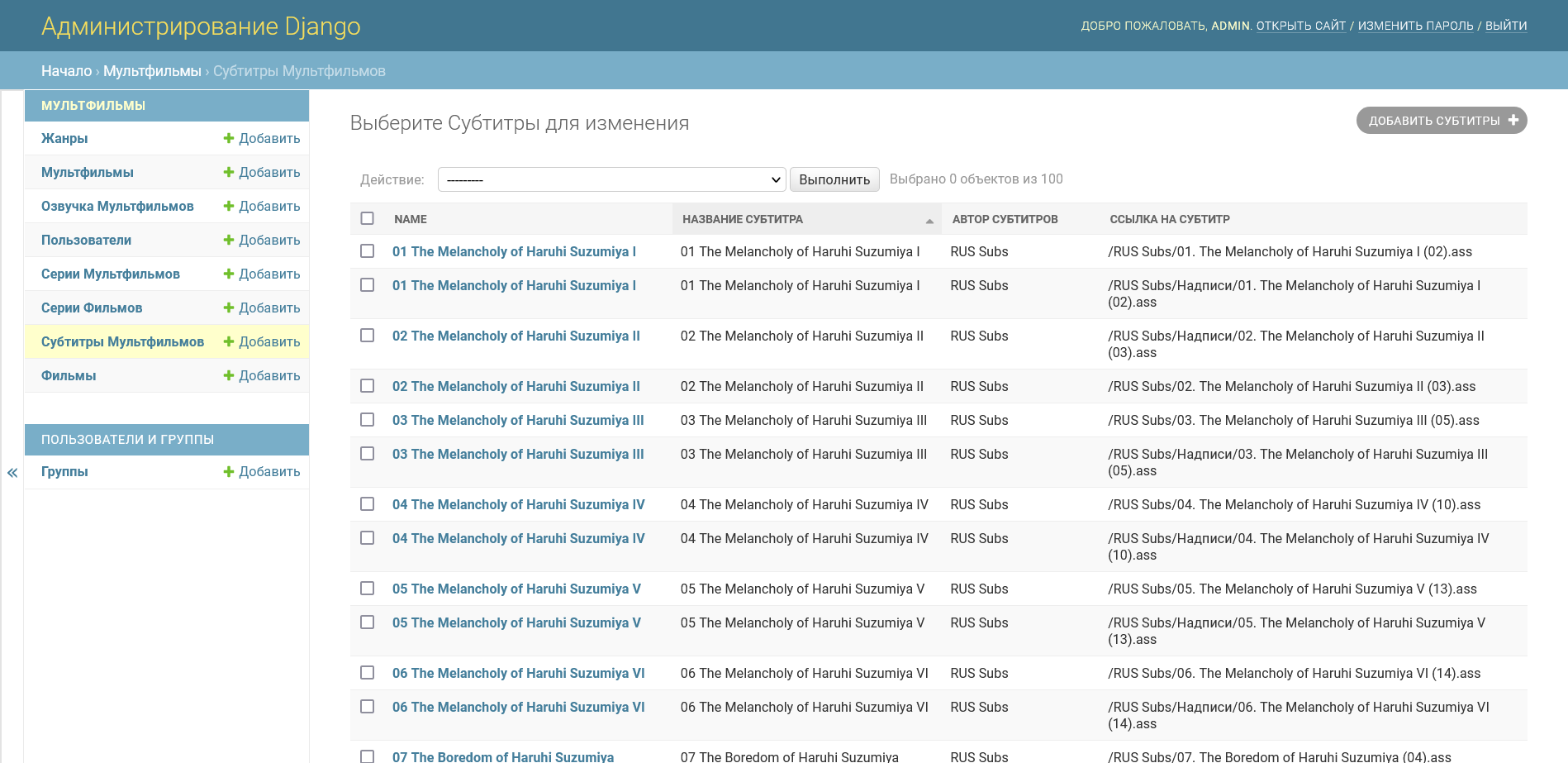


*Рисунок 3 страница изменения фильма*

**Изменение серий**

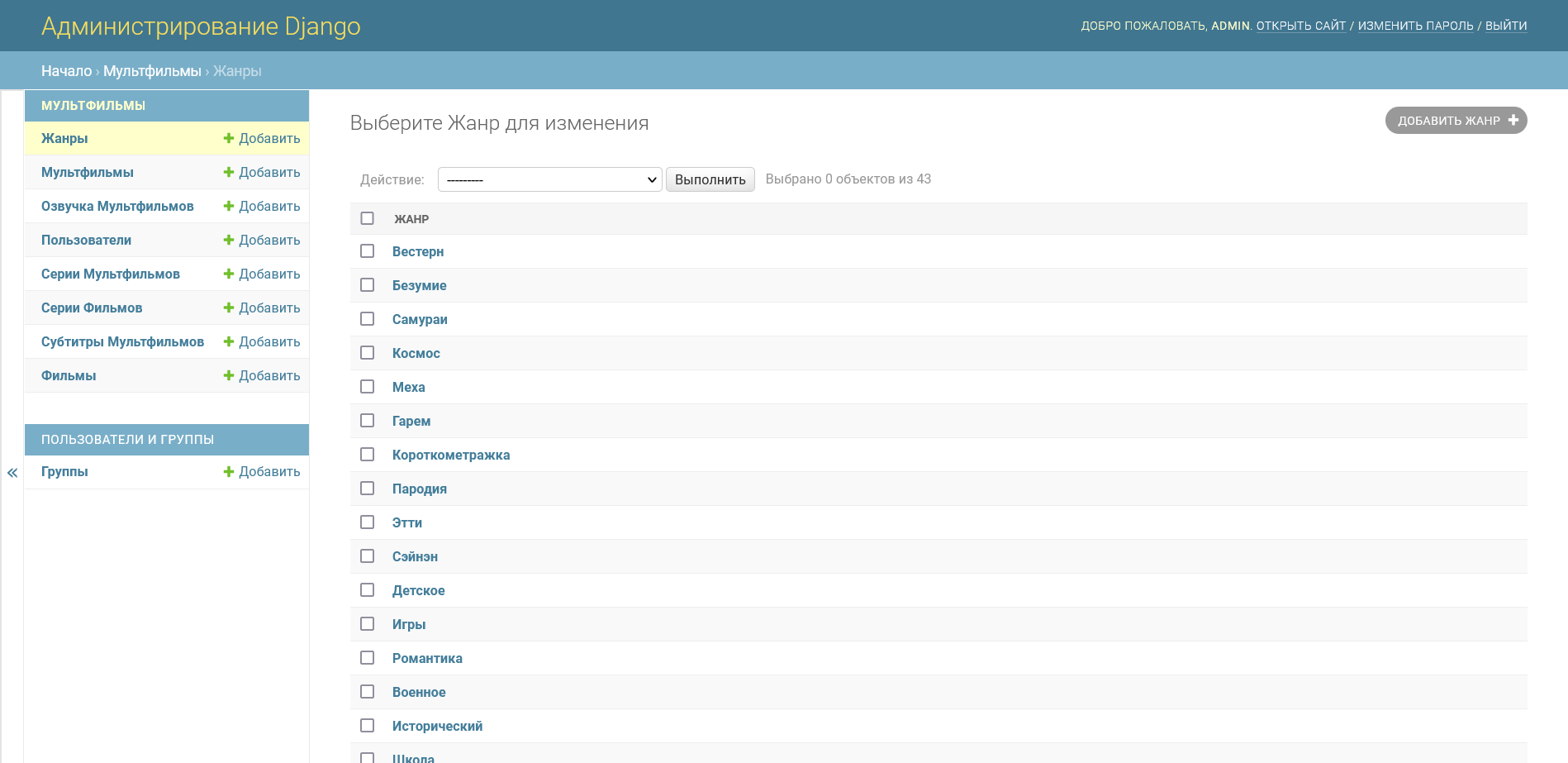
*Рисунок 4 страница изменения серий*

**Изменение субтитров**



*Рисунок 5 страница изменения субтитров*

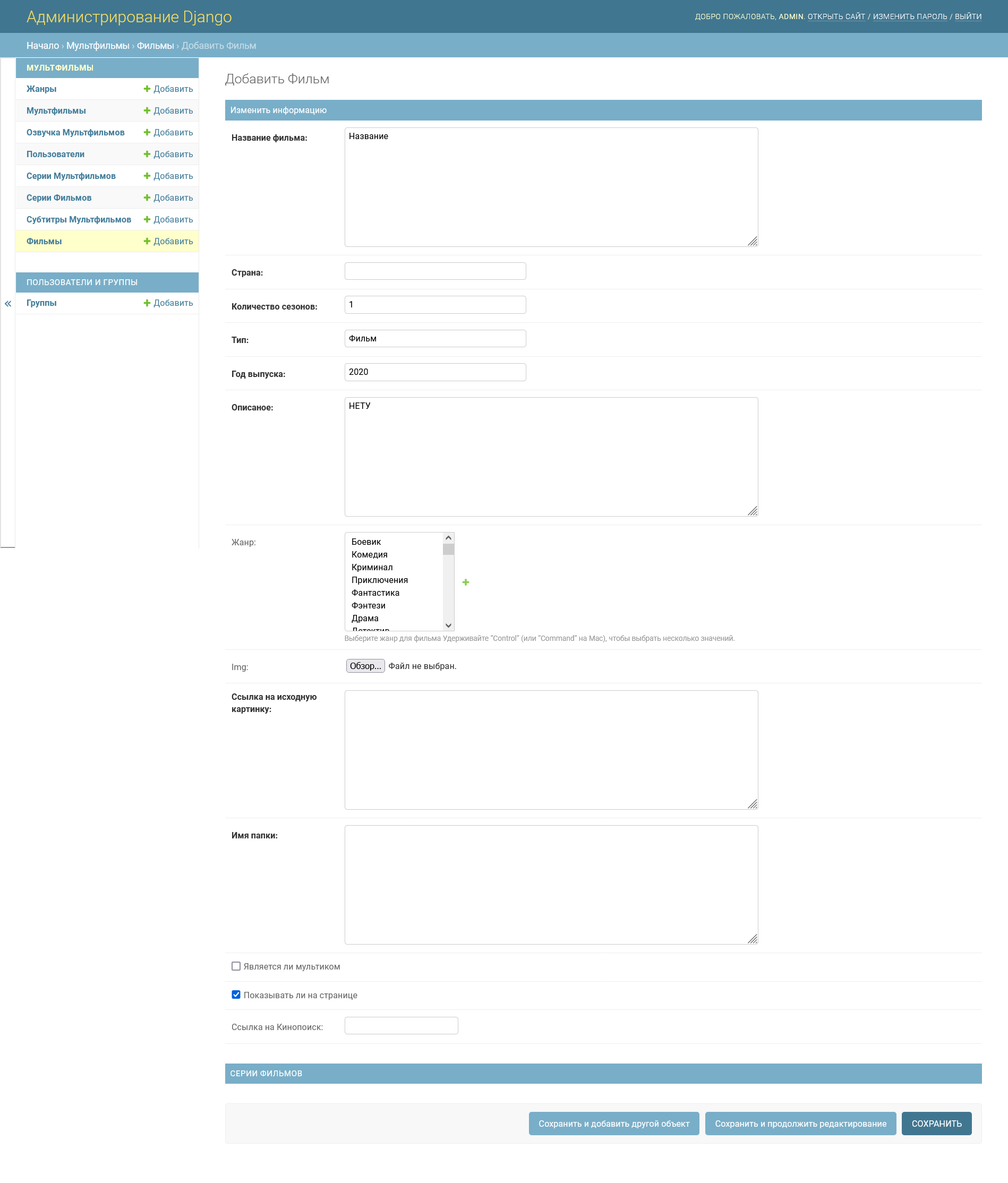
**Изменение жанров**



*Рисунок 6 страница изменения жанров*

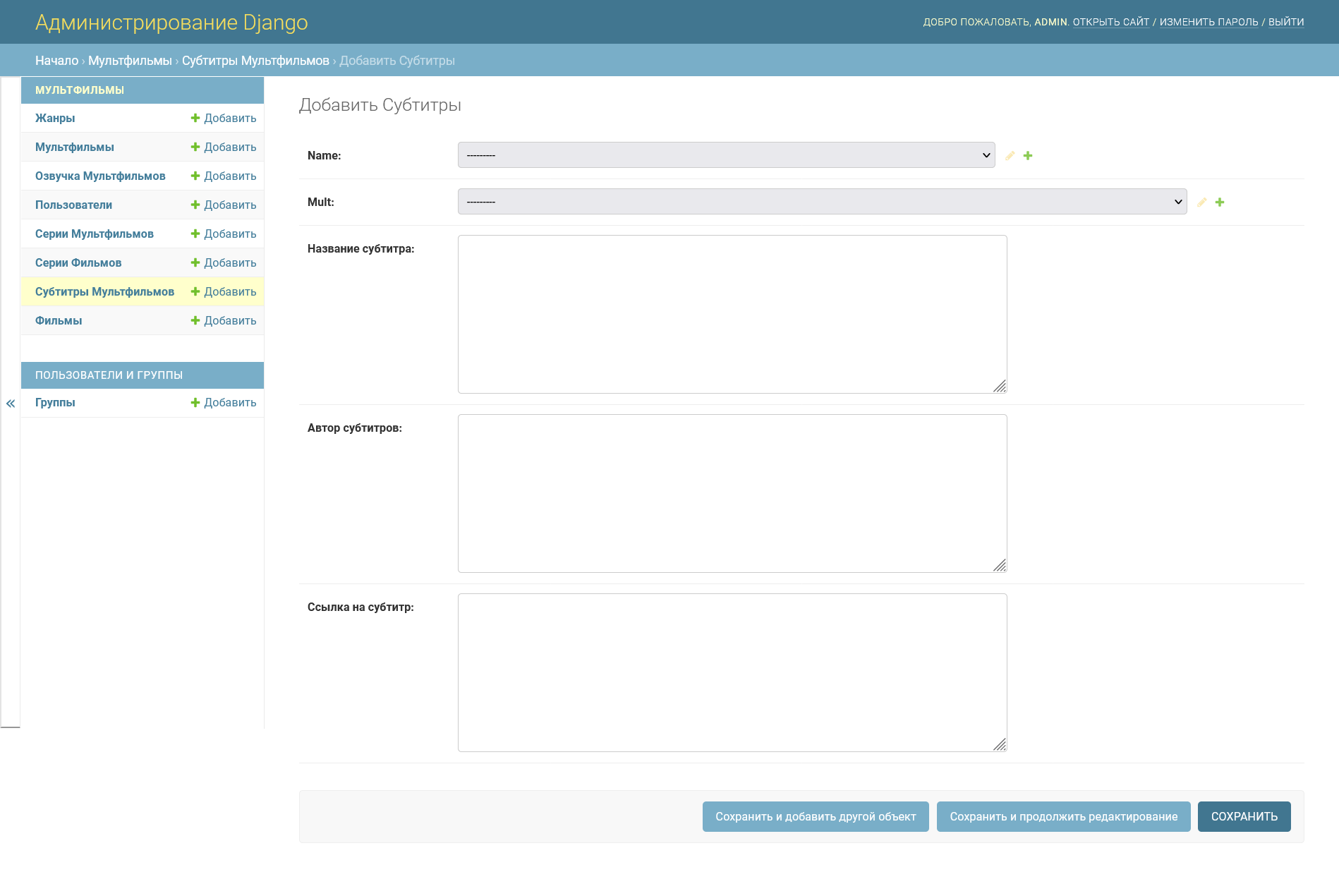
**Добавление данных**

**Добавление фильмов**



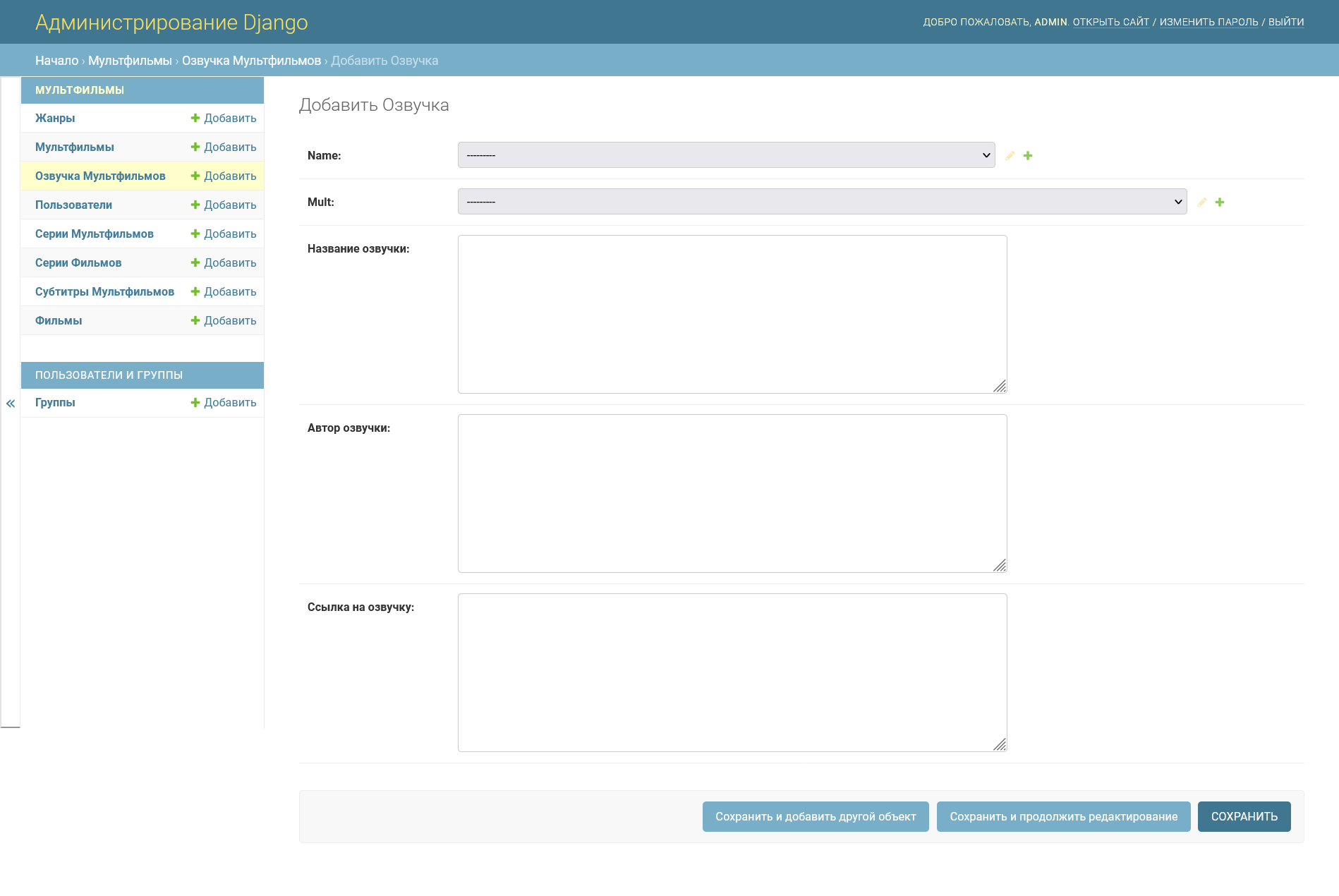
*Рисунок 7 страница добавления фильмов*

**Добавление субтитров**



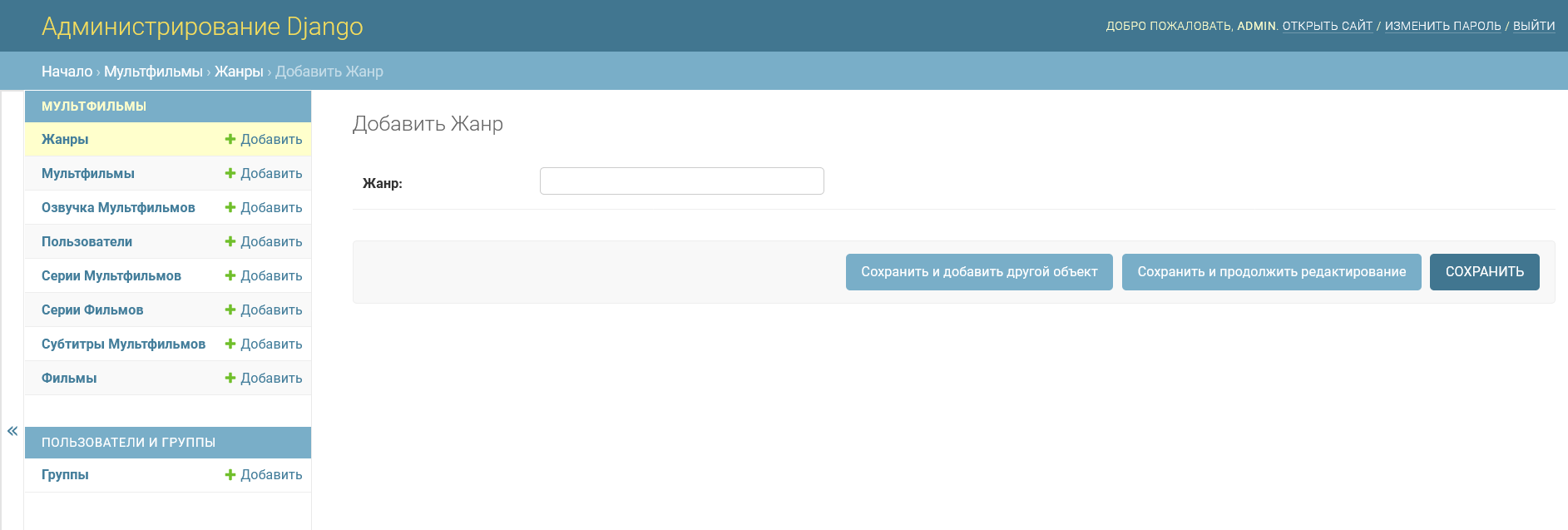
*Рисунок 8 страница добавления субтитров*

**Добавление озвучки**



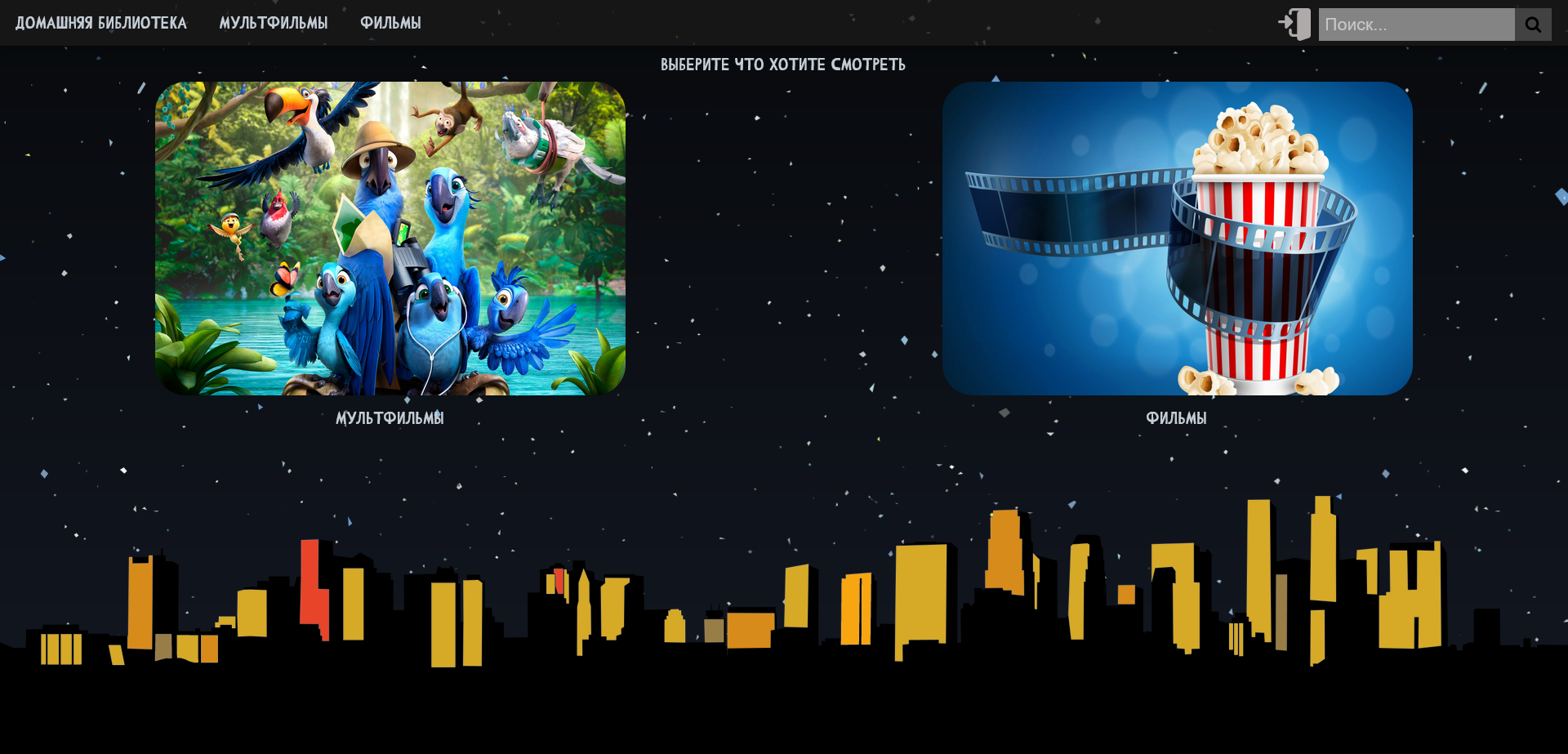
*Рисунок 9 страница добавления озвучки*

**Добавление жанров**



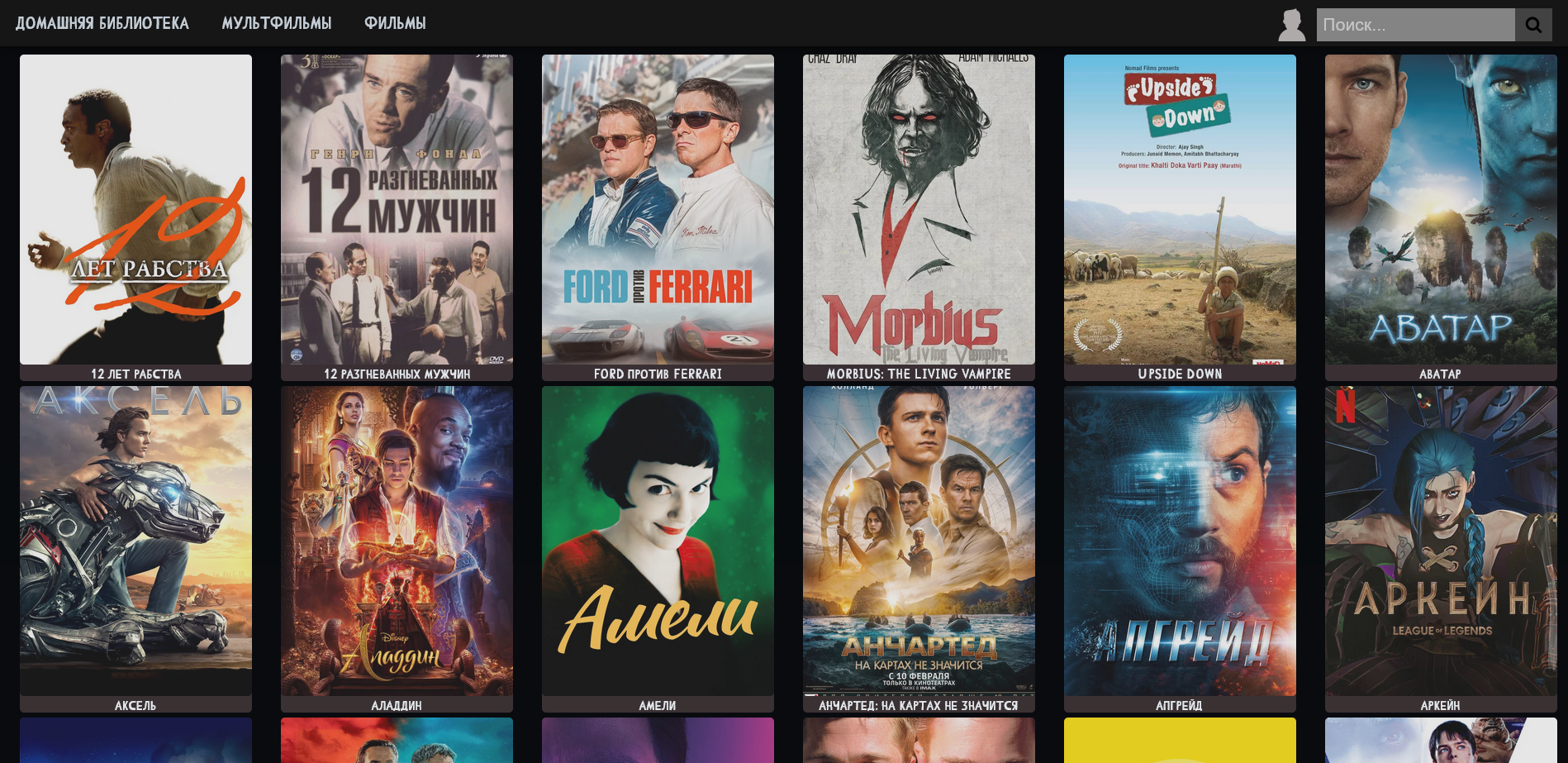
*Рисунок 10 страница добавления жанров*

**Главная страница**



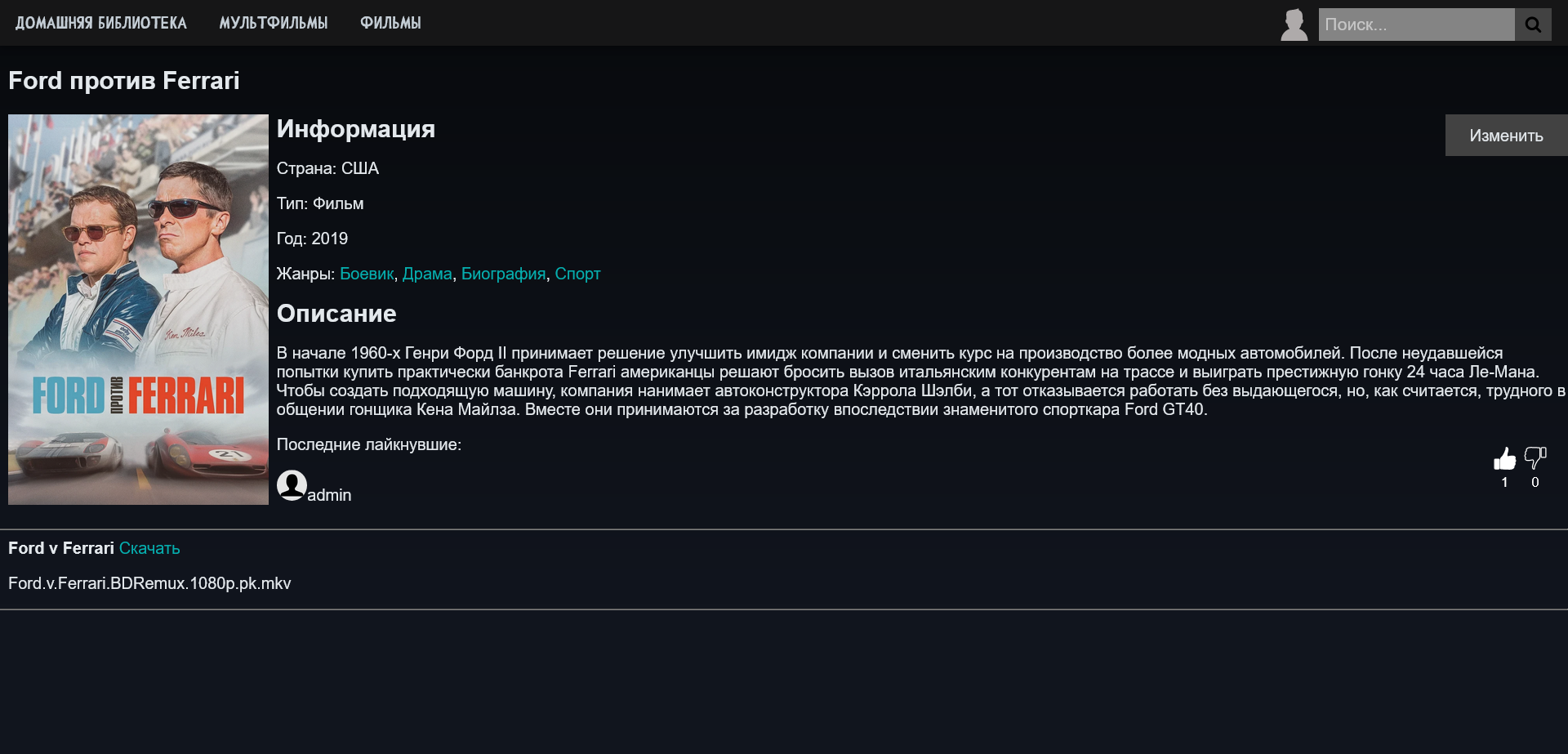
*Рисунок 11 главная страница*

**Страница выбора фильмов**



*Рисунок 12 страница выбора фильмов*

**Страница информации о фильме**



*Рисунок 11 страница информации о фильме*

**Вывод:**

Была произведена работа по созданию базы данных с помощью технологии ОРМ Python Django