**Министерство транспорта Российской Федерации**

**Федеральное государственное автономное образовательное**

**учреждение высшего образования**

**«РУТ (МИИТ)»**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Институт транспортной техники и систем управления**

**Кафедра «Управление и защита информации»**

Отчет по практической работе

по дисциплине

**«Основы построения защищенных баз данных»**

на тему

**«Приложение с базой данных библиотеки»**

Выполнил: ст. гр. ТКИ-541

Иванов А.К.

Проверил: доцент, к.т.н.

Васильева М.А.

Москва, 2024

Оглавление

[Цель работы 3](#_Toc163466463)

[Постановка задачи 4](#_Toc163466464)

[Листинг программы 5](#_Toc163466465)

[Описание классов 7](#_Toc163466466)

[Схема БД «Библиотека» 10](#_Toc163466467)

[Скриншоты работы 11](#_Toc163466468)

[Вывод 20](#_Toc163466469)

# Цель работы

Изучить современные технологии ORM, разработать приложение с базой данных, умеющее отрабатывать операции CRUD.

# Постановка задачи

При помощи языка программирования Python и фреймворка Django создать базу данных библиотеки.

# Листинг программы

**from django.db import models  
  
class Series(models.Model):  
 title\_series = models.CharField(max\_length=255)  
 note = models.TextField(default='', blank=True)  
  
 def \_\_str\_\_(self):  
 return f"{self.title\_series}"  
  
 class Meta:  
 verbose\_name = "Серия книг"  
 verbose\_name\_plural = "Серии книг"  
  
class City(models.Model):  
 city = models.CharField(max\_length=255)  
  
 def \_\_str\_\_(self):  
 return f"{self.city}"  
  
 class Meta:  
 verbose\_name = "Город"  
 verbose\_name\_plural = "Города"  
  
class Publishing(models.Model):  
 id\_city = models.ForeignKey(City, on\_delete=models.CASCADE)  
 publishining\_office = models.CharField(max\_length=255)  
 note = models.TextField(default='', blank=True)  
  
 def \_\_str\_\_(self):  
 return f"{self.publishining\_office}"  
  
 class Meta:  
 verbose\_name = "Издатель"  
 verbose\_name\_plural = "Издатели"  
class Room(models.Model):  
 room\_name = models.CharField(max\_length=255)  
 note = models.TextField(default='', blank=True)  
  
 def \_\_str\_\_(self):  
 return f"{self.room\_name}"  
  
 class Meta:  
 verbose\_name = "Комната"  
 verbose\_name\_plural = "Комнаты"  
  
class Shelf(models.Model):  
 id\_room = models.ForeignKey(Room, on\_delete=models.CASCADE)  
 number\_shelf = models.PositiveSmallIntegerField()  
 note = models.TextField(default='', blank=True)  
  
 def \_\_str\_\_(self):  
 return f"№{self.number\_shelf} {self.id\_room}"  
  
 class Meta:  
 verbose\_name = "Полка"  
 verbose\_name\_plural = "Полки"  
  
class Author(models.Model):  
 first\_name = models.CharField(max\_length=255)  
 middle\_name = models.CharField(max\_length=255, default='', blank=True)  
 last\_name = models.CharField(max\_length=255)  
 birth\_date = models.DateField()  
 death\_date = models.DateField(default='', blank=True)  
 note = models.TextField(default='', blank=True)  
  
 def \_\_str\_\_(self):  
 return f"{self.first\_name} {self.last\_name}"  
  
 class Meta:  
 verbose\_name = "Автор"  
 verbose\_name\_plural = "Авторы"  
  
class Book(models.Model):  
 title\_book = models.CharField(max\_length=255)  
 year\_of\_publishing = models.PositiveSmallIntegerField()  
 edition = models.PositiveSmallIntegerField()  
 isbn = models.CharField(max\_length=17)  
 number\_of\_copies = models.PositiveSmallIntegerField()  
 volume = models.PositiveSmallIntegerField()  
 id\_publishing = models.ForeignKey(Publishing, on\_delete=models.CASCADE)  
 id\_shelf = models.ForeignKey(Shelf, on\_delete=models.CASCADE)  
 id\_series = models.ForeignKey(Series, on\_delete=models.CASCADE)  
 id\_author = models.ForeignKey(Author, on\_delete=models.CASCADE)  
 note = models.TextField(default='', blank=True)  
  
 def \_\_str\_\_(self):  
 return f"{self.id\_author}, \"{self.title\_book}\", {self.id\_publishing} Изд. {self.edition}, {self.year\_of\_publishing} г. - {self.volume} стр."  
  
 class Meta:  
 verbose\_name = "Книга"  
 verbose\_name\_plural = "Книги"  
  
class AuthorBook(models.Model):  
 id\_author = models.ForeignKey(Author, on\_delete=models.CASCADE)  
 id\_book = models.ForeignKey(Book, on\_delete=models.CASCADE)  
  
 def \_\_str\_\_(self):  
 return f"{self.id\_book}"  
  
 class Meta:  
 verbose\_name = "Автор-книга"  
 verbose\_name\_plural = "Авторы-книги"**

# Описание классов

*Author*

|  |  |  |
| --- | --- | --- |
| Название поля | Тип данных | Обязательное |
| *first\_name* | *CharField(255)* | *Да* |
| *middle\_name* | *CharField(255)* | *Нет* |
| *last\_name* | *CharField(255)* | *Да* |
| *birth\_date* | *DateField()* | *Да* |
| *death\_date* | *DateField()* | *Нет* |
| *note* | *TextField()* | *Нет* |

*Book*

|  |  |  |
| --- | --- | --- |
| Название поля | Тип данных | Обязательное |
| *title\_book* | *CharField(255)* | *Да* |
| *year\_of\_publishing* | *PositiveSmallIntegerField()* | *Да* |
| *edition* | *PositiveSmallIntegerField()* | *Да* |
| *isbn* | *CharField(17)* | *Да* |
| *number\_of\_copies* | *PositiveSmallIntegerField()* | *Да* |
| *volume* | *PositiveSmallIntegerField()* | *Да* |
| *id\_publishing* | *ForeignKey(Publishing)* | *Да* |
| *id\_shelf* | *ForeignKey(Shelf)* | *Да* |
| *id\_series* | *ForeignKey(Series)* | *Да* |
| *id\_author* | *ForeignKey(Author)* | *Да* |
| *note* | *TextField()* | *Нет* |

*AuthorBook*

|  |  |  |
| --- | --- | --- |
| Название поля | Тип данных | Обязательное |
| *id\_author* | *ForeignKey(Author)* | *Да* |
| *id\_book* | *ForeignKey(Book)* | *Да* |

*Series*

|  |  |  |
| --- | --- | --- |
| Название поля | Тип данных | Обязательное |
| *title\_series* | *CharField(255)* | *Да* |
| *note* | *TextField()* | *Нет* |

*Publishing*

|  |  |  |
| --- | --- | --- |
| Название поля | Тип данных | Обязательное |
| *id\_city* | *ForeignKey(City)* | *Да* |
| *publishining\_office* | *CharField(255)* | *Да* |
| *note* | *TextField()* | *Нет* |

*City*

|  |  |  |
| --- | --- | --- |
| Название поля | Тип данных | Обязательное |
| *city* | *CharField(255)* | *Да* |

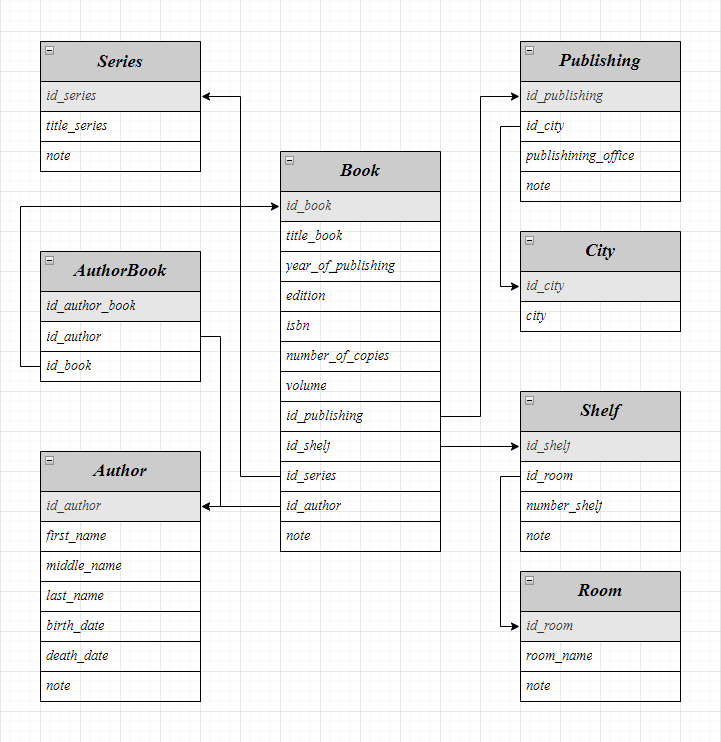
*Shelf*

|  |  |  |
| --- | --- | --- |
| Название поля | Тип данных | Обязательное |
| *id\_room* | *ForeignKey(Room)* | *Да* |
| *number\_shelf* | *PositiveSmallIntegerField()* | *Да* |
| *note* | *TextField()* | *Нет* |

*Room*

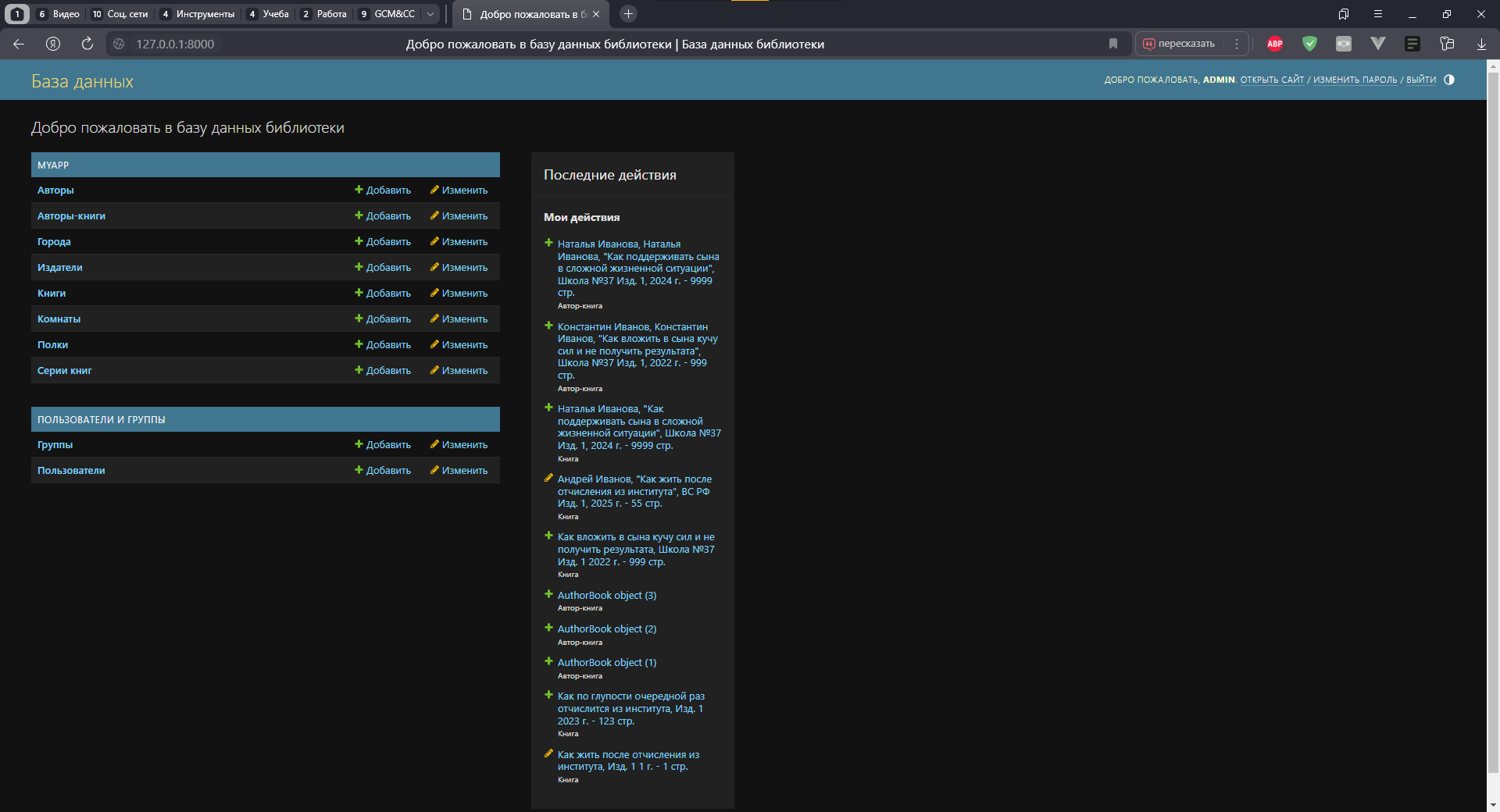
|  |  |  |
| --- | --- | --- |
| Название поля | Тип данных | Обязательное |
| *room\_name* | *CharField(255)* | *Да* |
| *note* | *TextField()* | *Нет* |

# Схема БД «Библиотека»



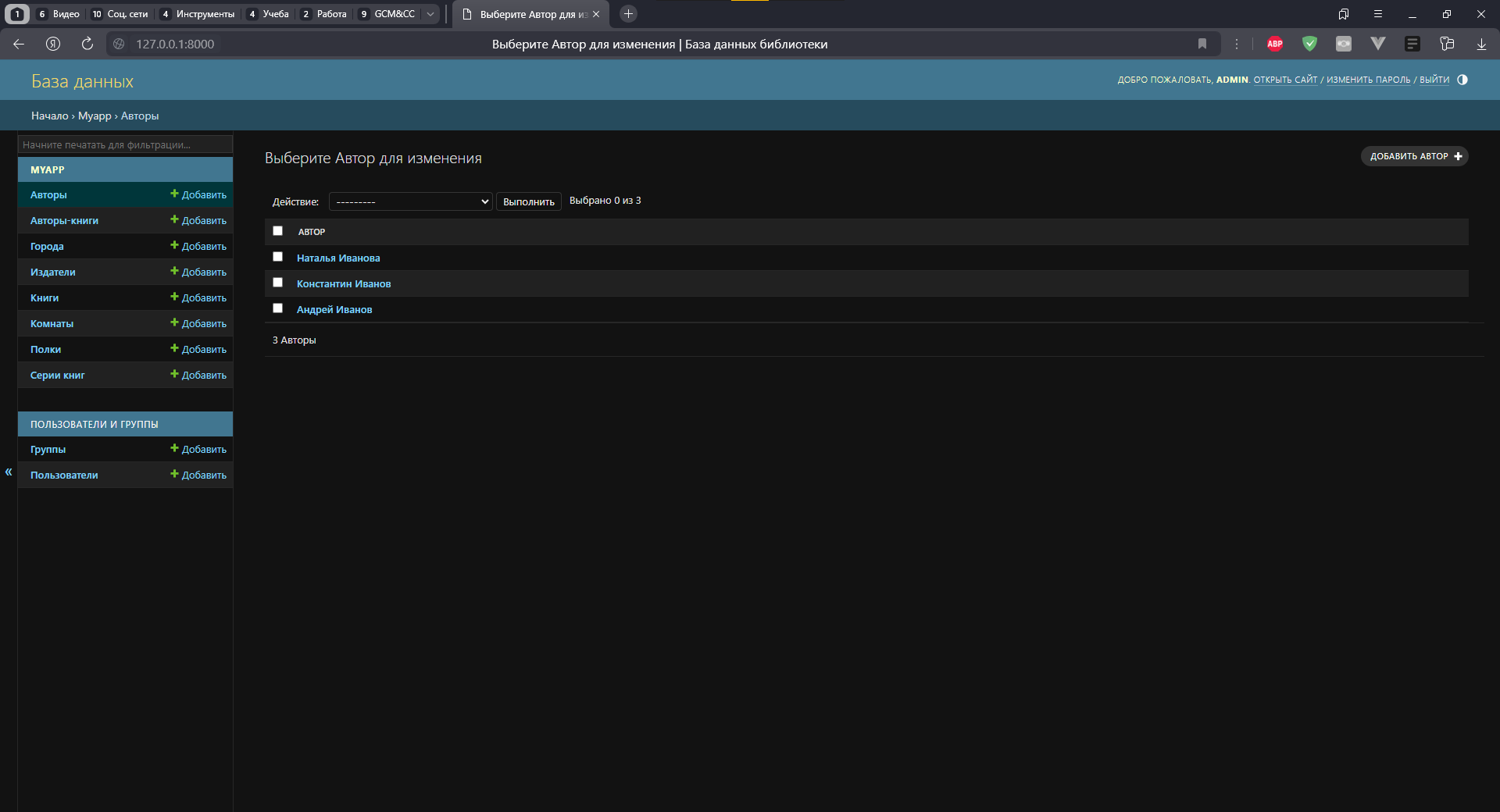
# Скриншоты работы

*Страница администратора*

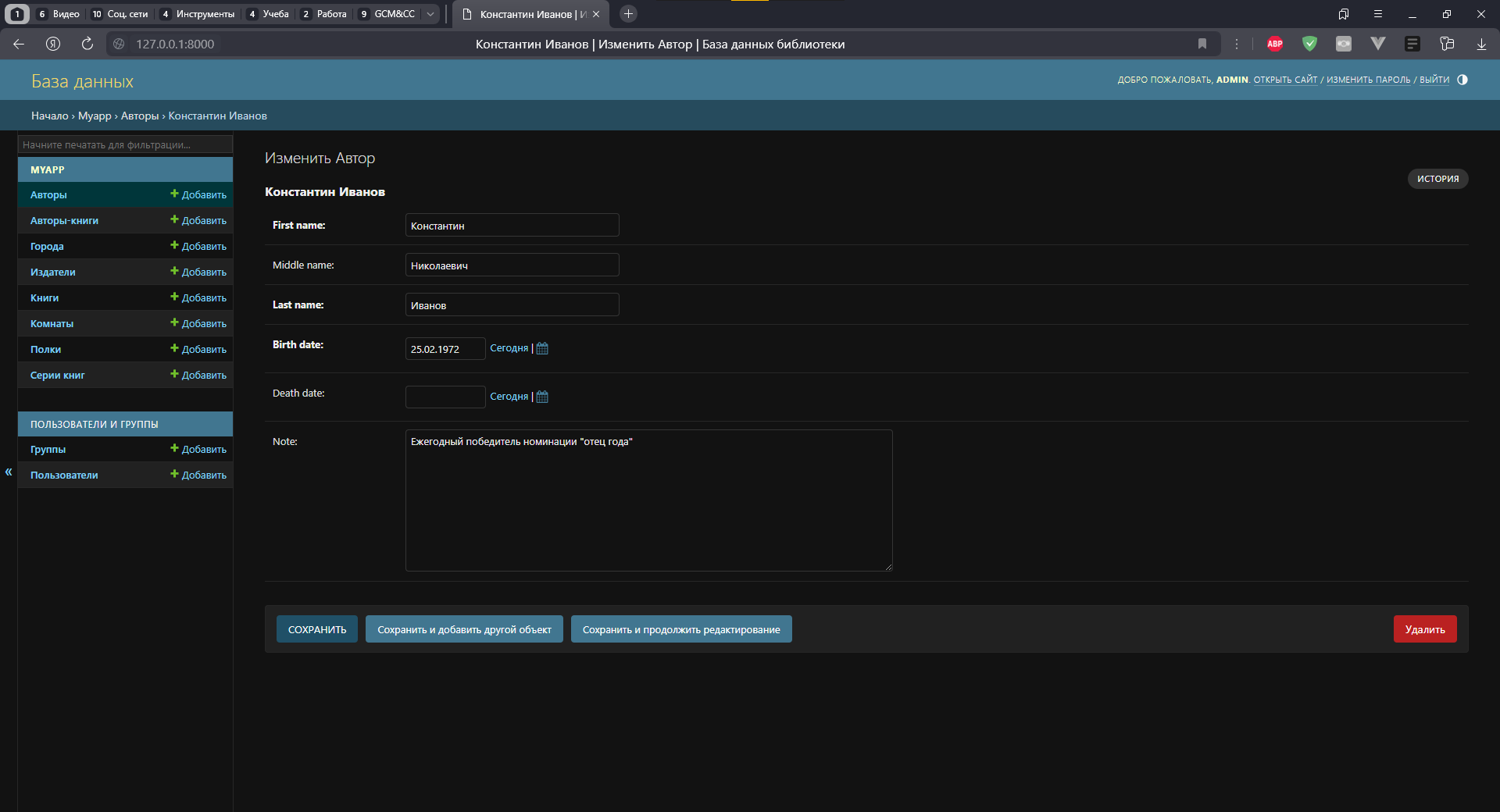


*Рисунок 1 – страница администратора*

*Страница со списком авторов*

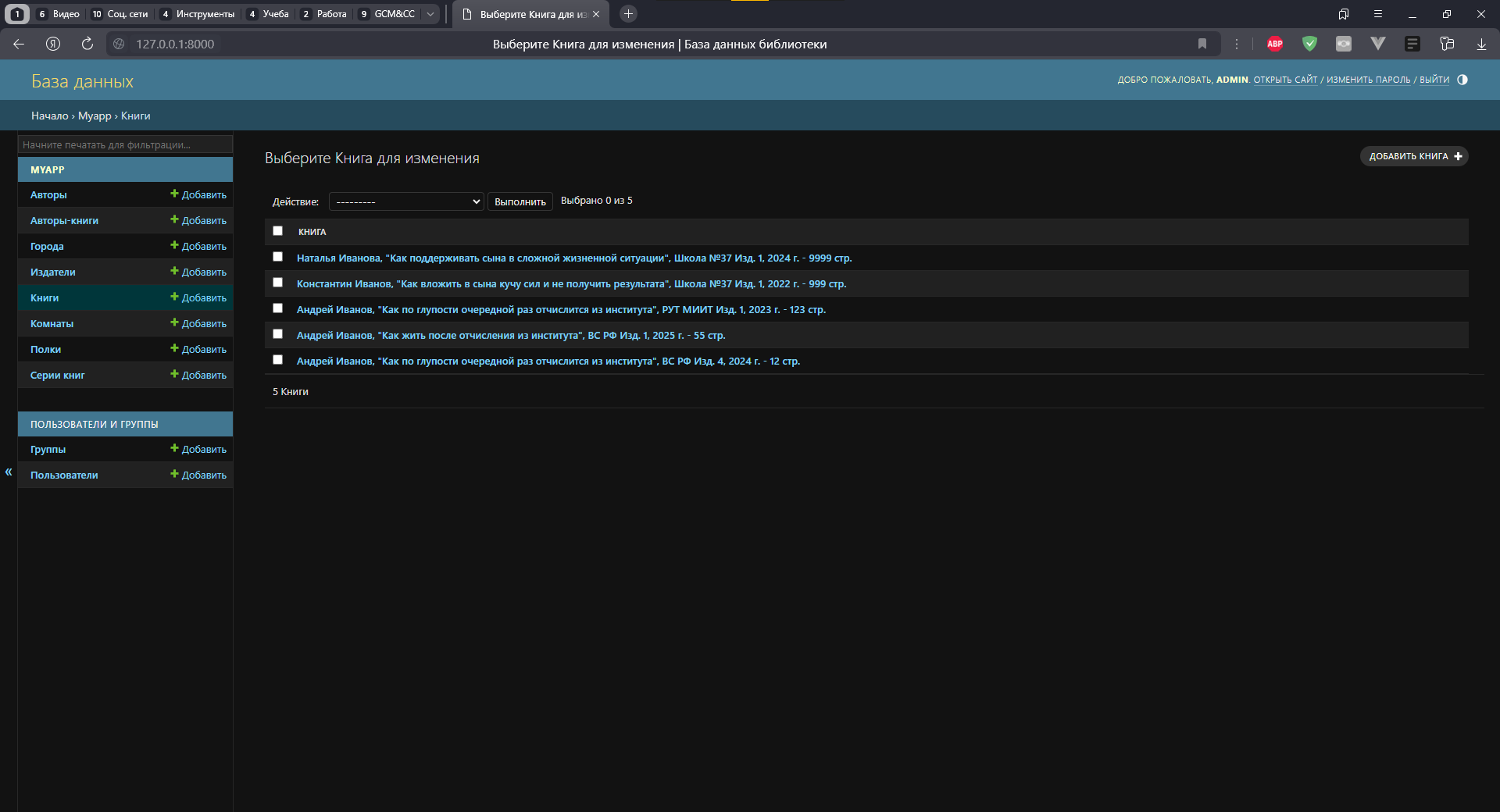


*Рисунок 2 – список авторов*

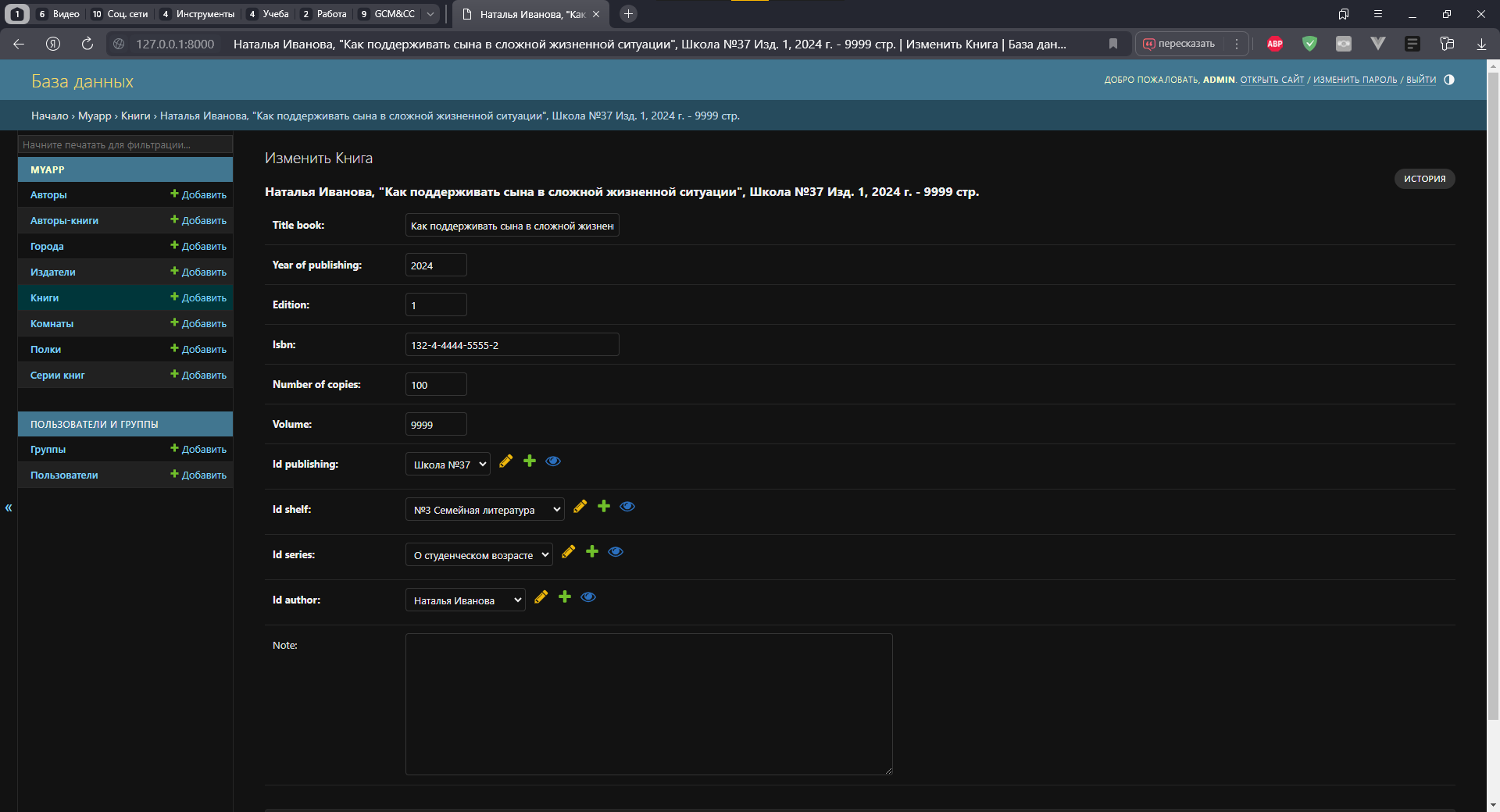


*Рисунок 3 – изменение списка авторов*

*Страница со списком книг*

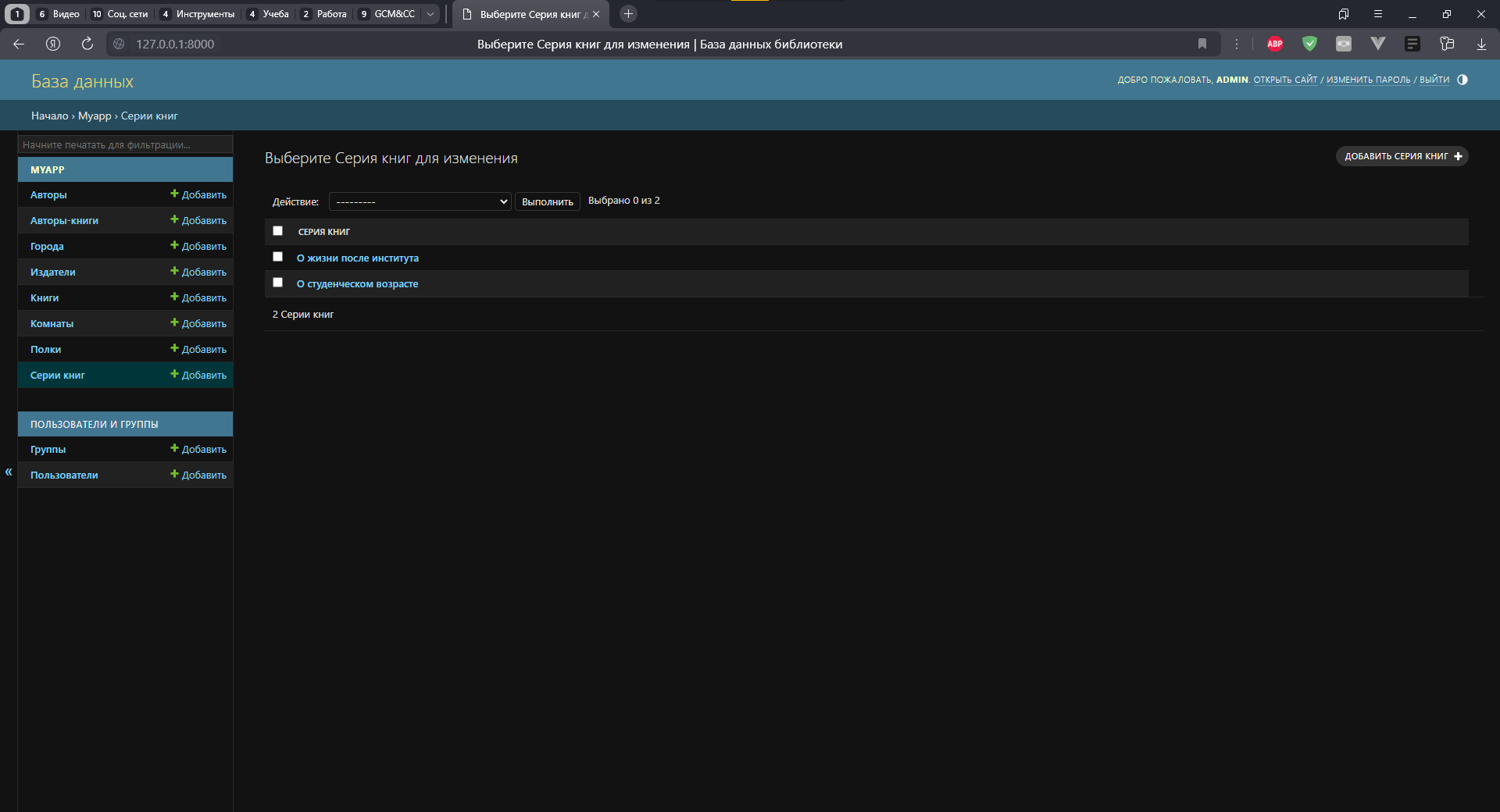


*Рисунок 4 – список книг*

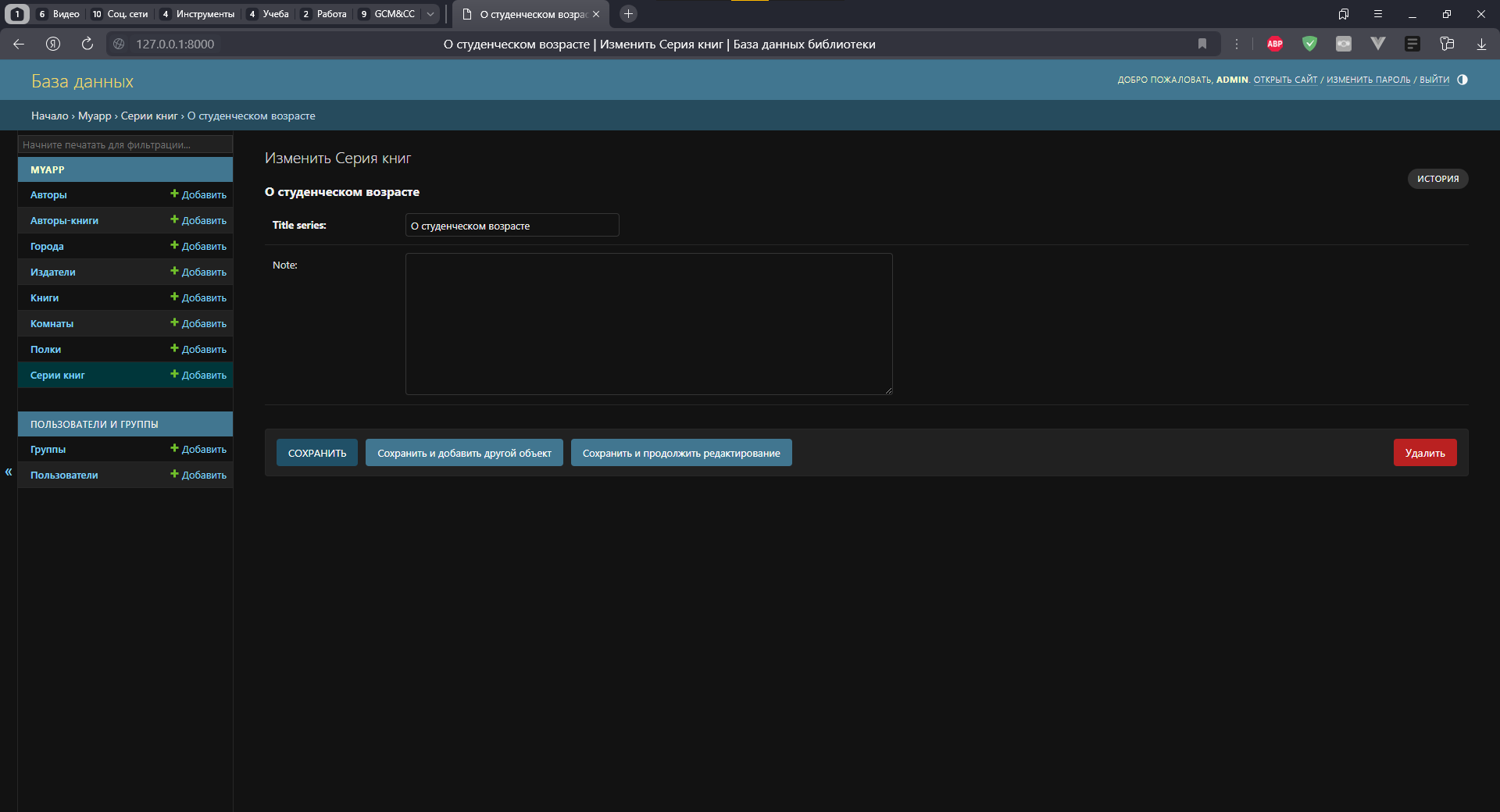


*Рисунок 5 – изменение списка книг*

*Страница со списком серий книг*

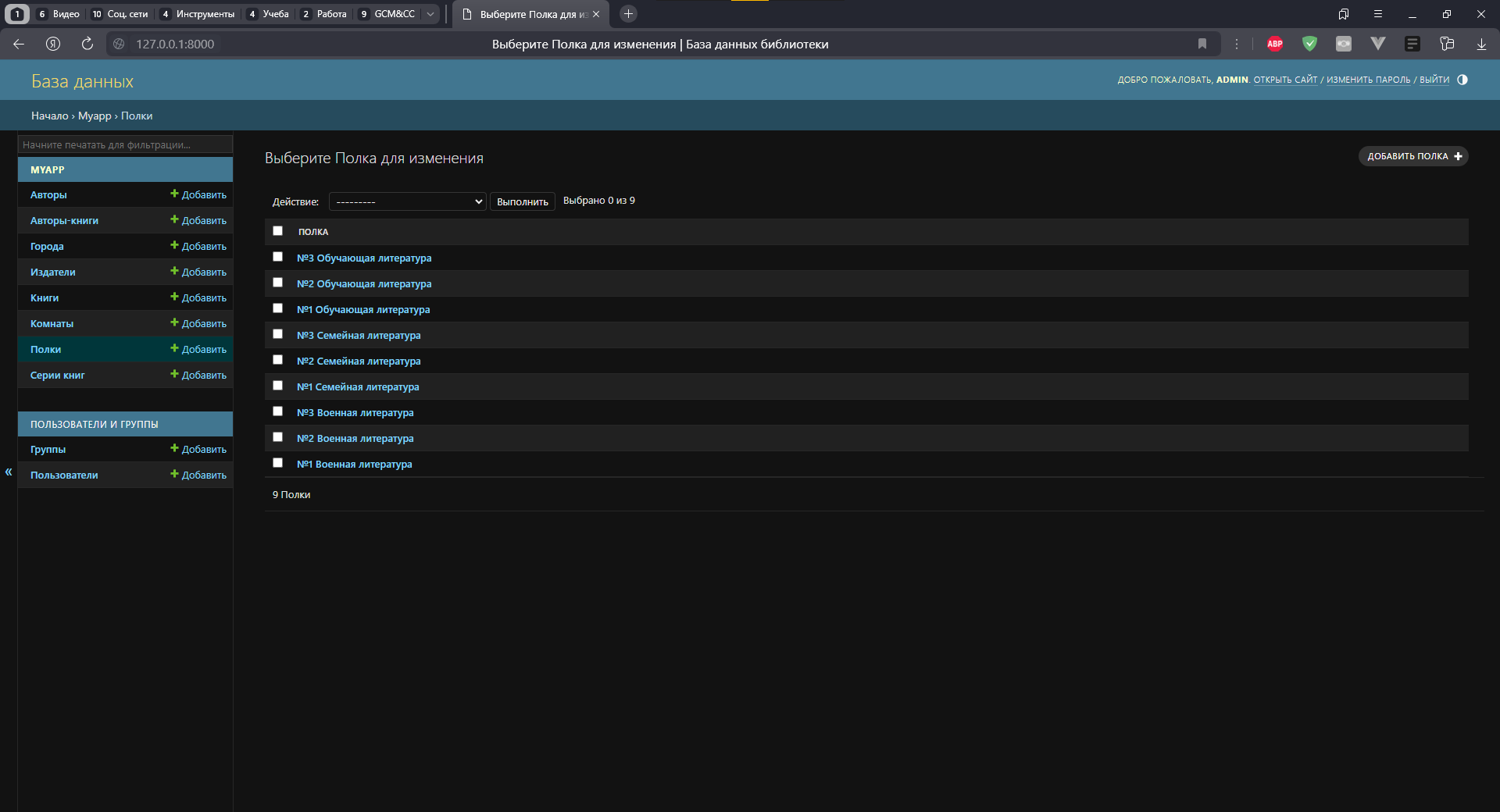


*Рисунок 6 – список серий книг*

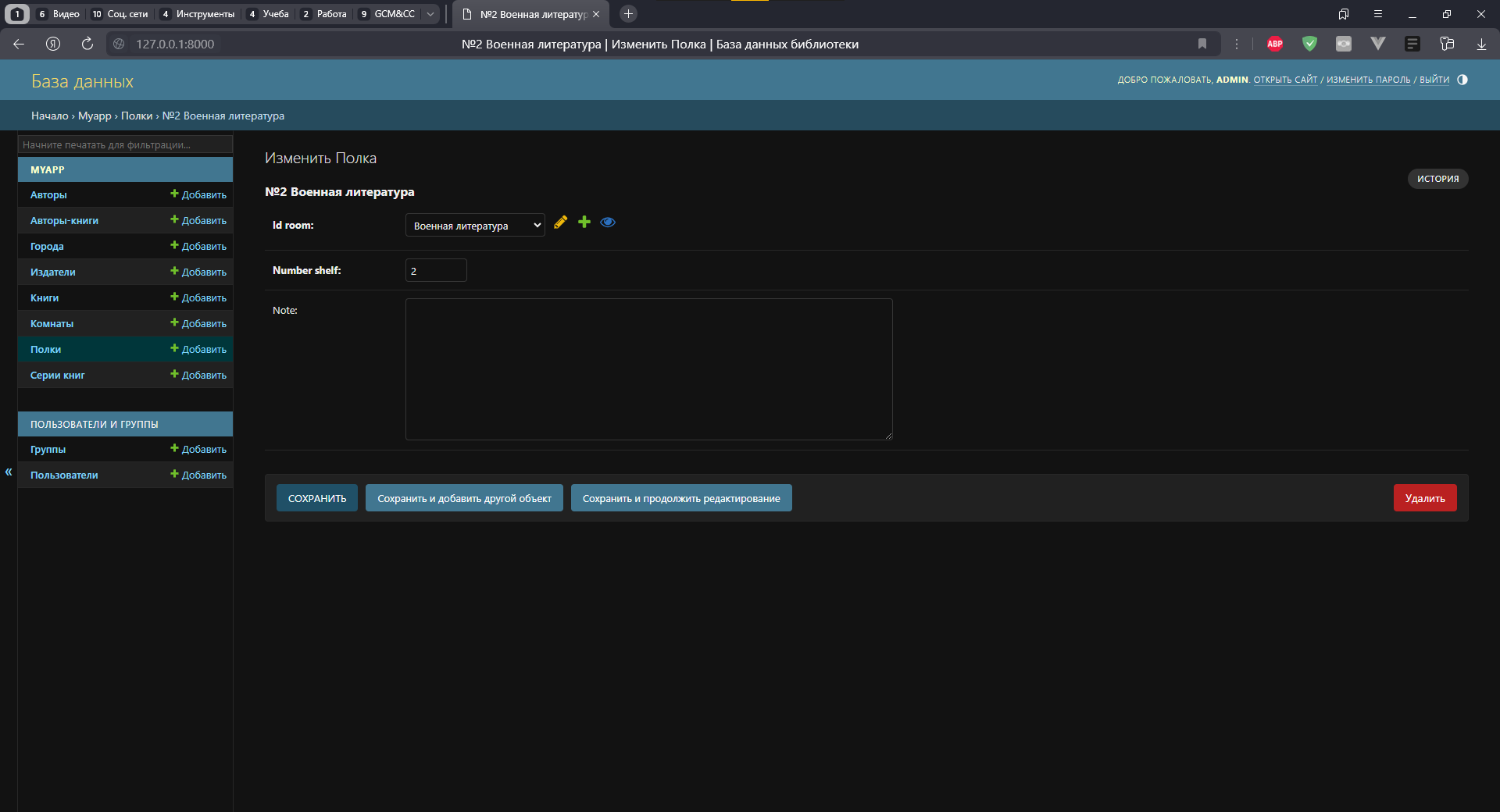


*Рисунок 7 – изменение списка серий книг*

*Страница со списком полок*

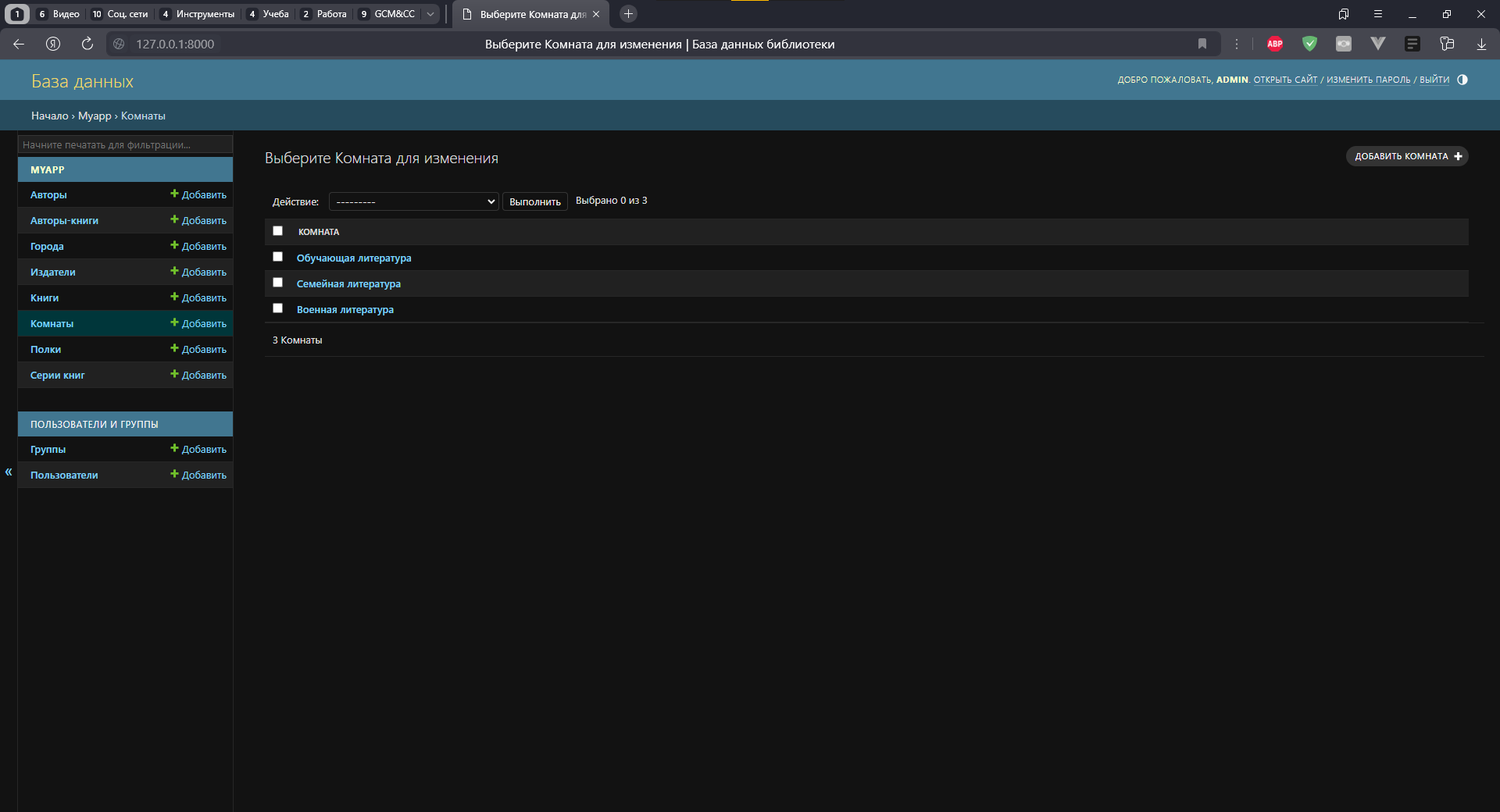


*Рисунок 8 – список полок*

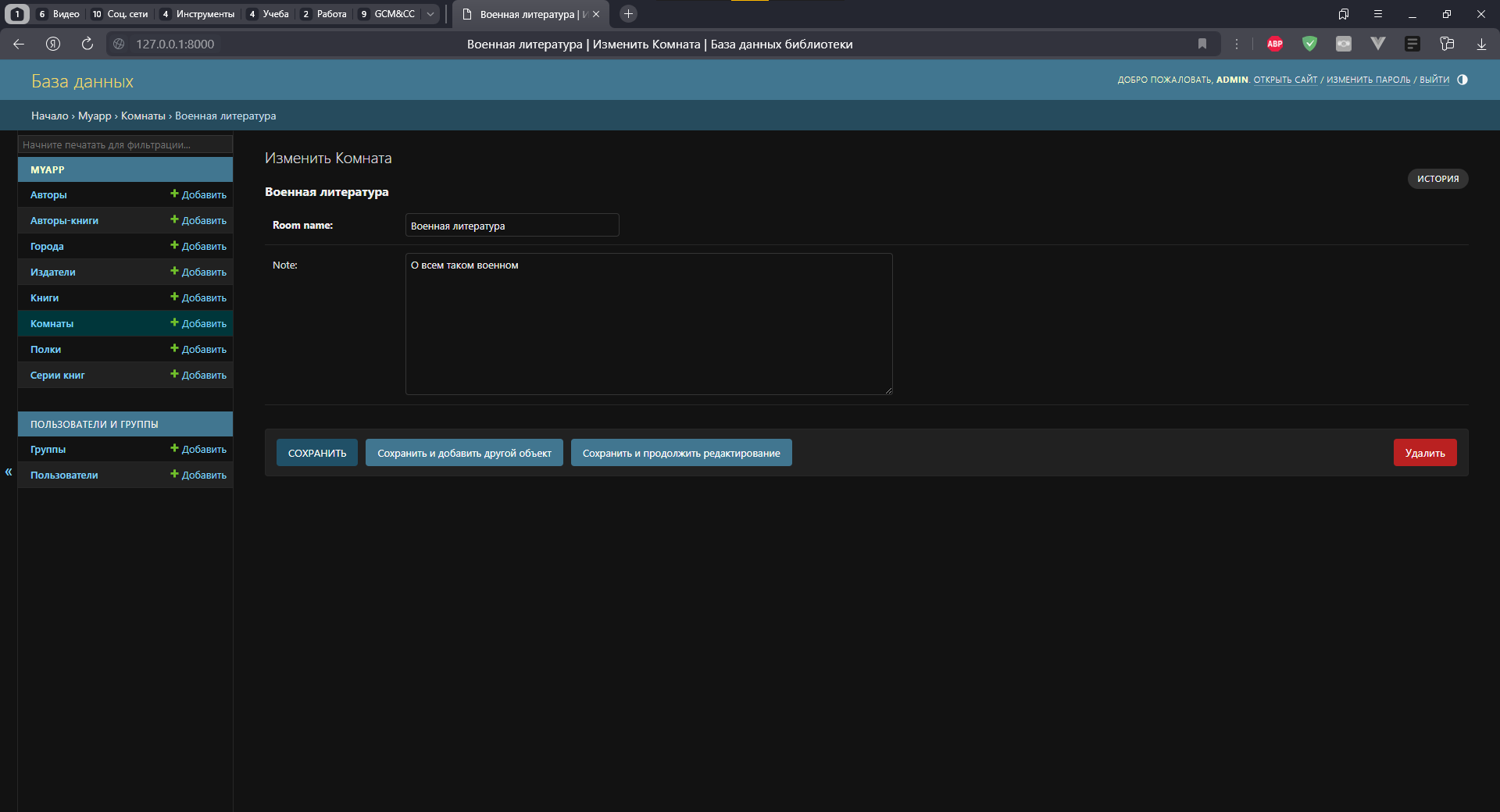


*Рисунок 9 – изменение списка полок*

*Страница со списком комнат*

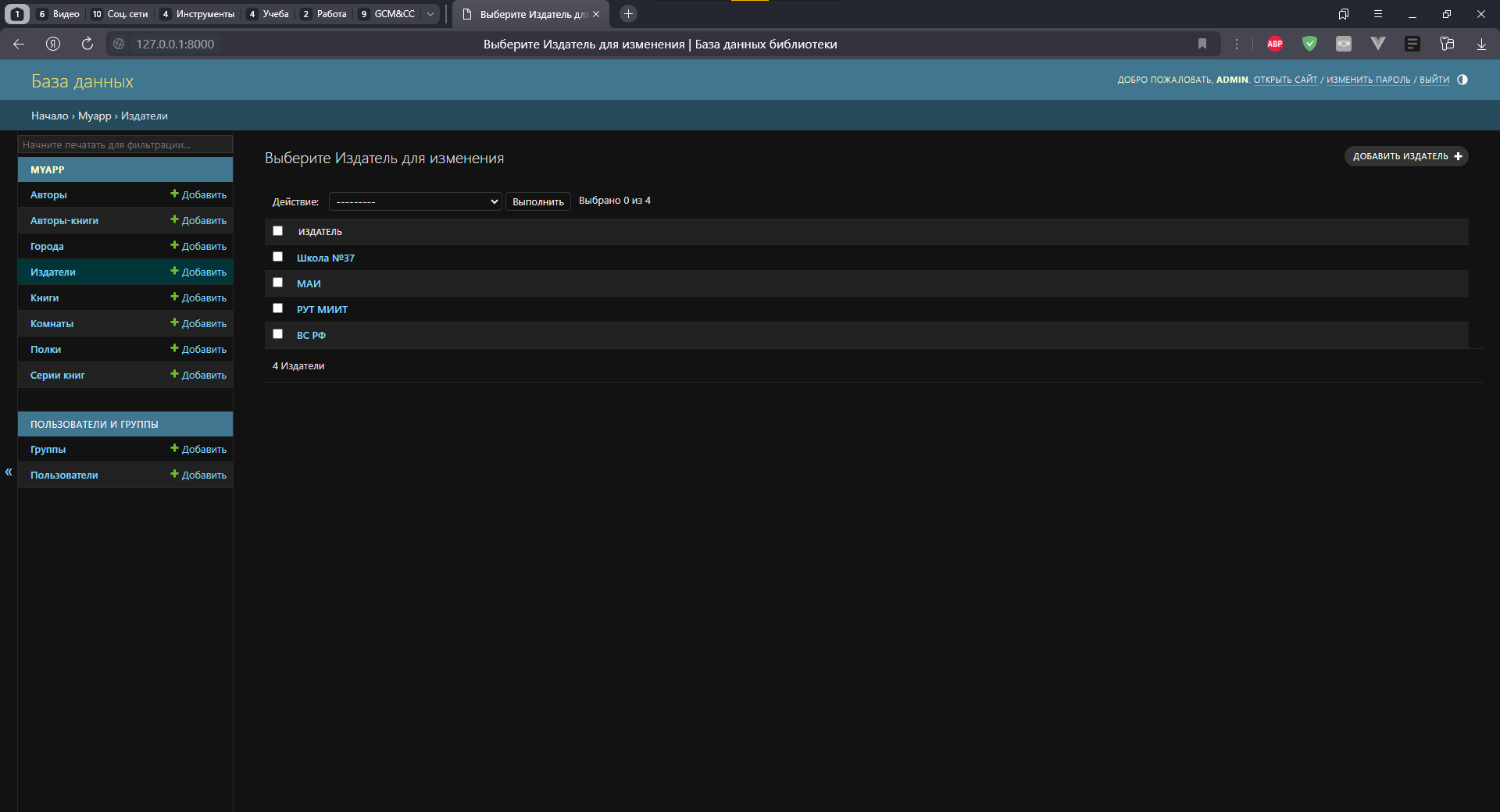


*Рисунок 10 – список комнат*

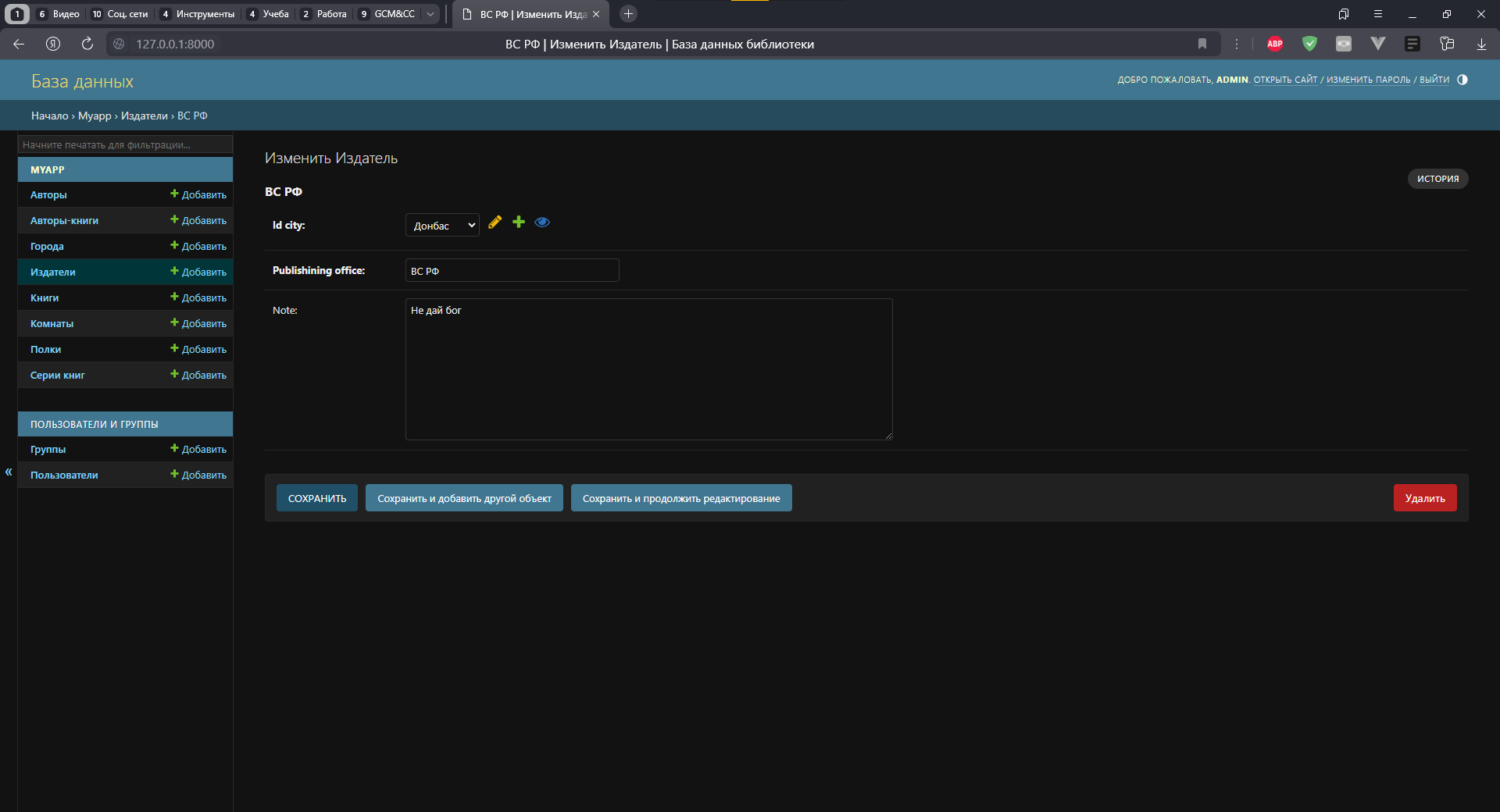


*Рисунок 11 – изменение списка комнат*

*Страница со списком издателей*

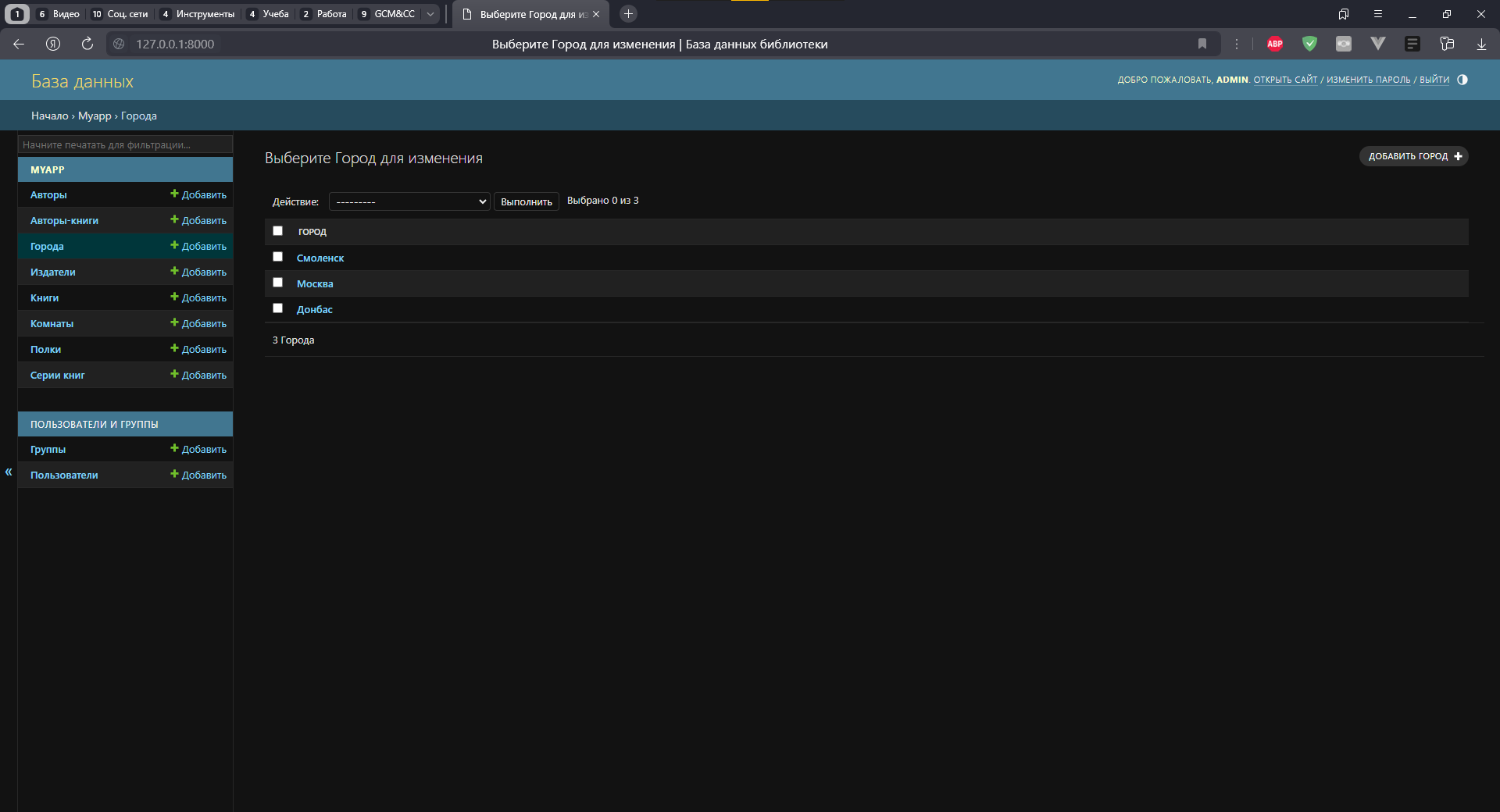


*Рисунок 12 – список издателей*

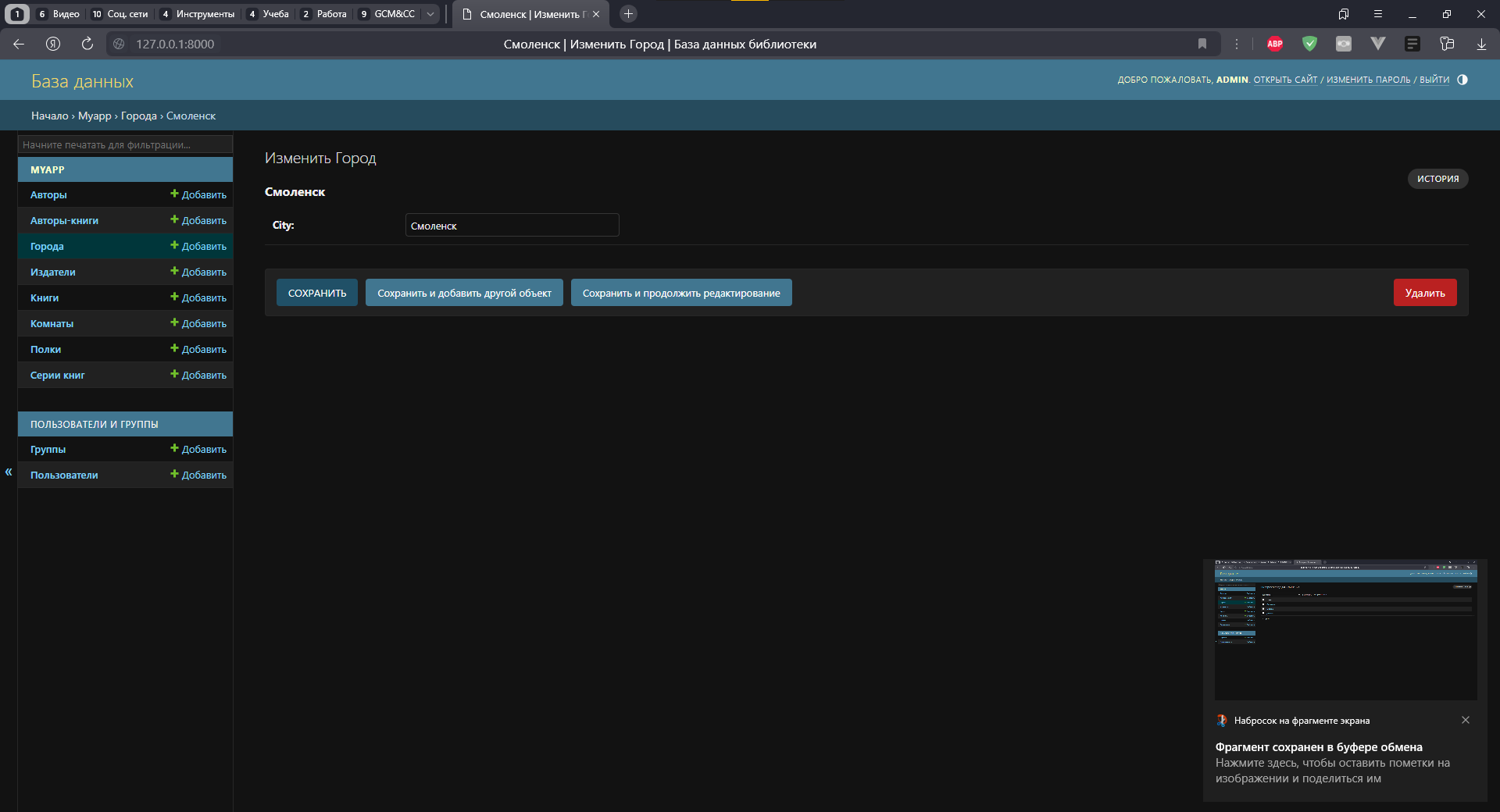


*Рисунок 13 – изменение списка издателей*

*Страница со списком городов*

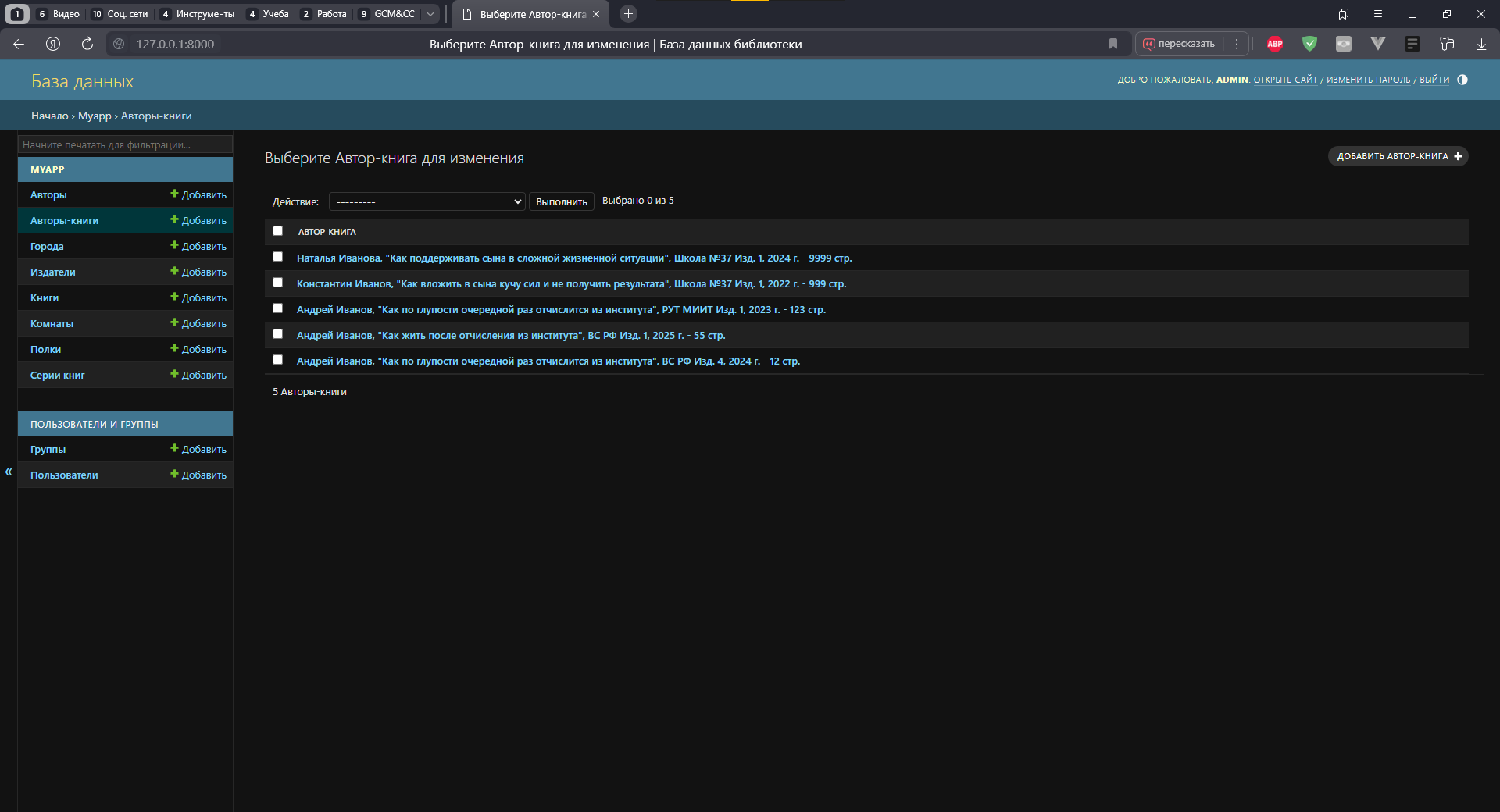


*Рисунок 14 – список городов*

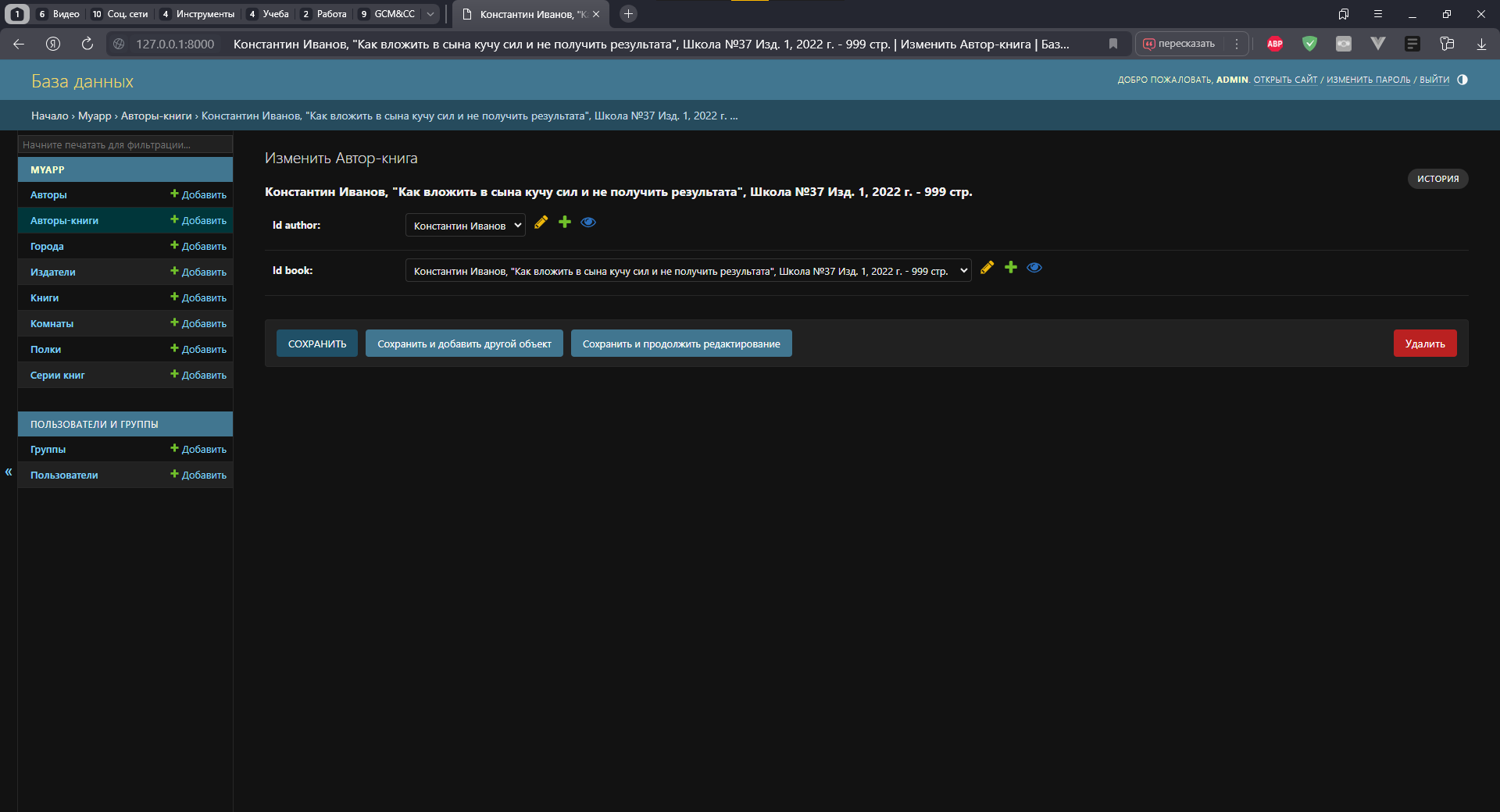


*Рисунок 15 – изменение списка городов*

*Страница со списком связей «автор-книга»*



*Рисунок 16 – список связей «автор-книга»*



*Рисунок 17 – изменение списка связей «автор-книга»*

# Вывод

Изучил современные технологии ORM, при помощи языка программирования Python и фреймворка Django разработал приложение с базой данных библиотеки, умеющее отрабатывать операции CRUD.