



**Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)**

ФАКУЛЬТЕТ _____ Радиотехнический (РТ) _____

КАФЕДРА _____ Системы обработки информации и управления (ИУ5) _____

**Рубежный контроль №2
по курсу «Разработка интернет-приложений»
17 вариант**

Студентка РТ5-51Б
(Группа)

Стадник Е.Р.
(Фамилия И.О.)

Преподаватель:

Гапанюк Ю. Е.
(Фамилия И.О.)

2021 г.

Вариант задания:

№ варианта	Класс 1	Класс 2
17	Дирижер	Оркестр

Задание:

Рубежный контроль представляет собой разработку веб-приложения с использованием фреймворка Django. Веб-приложение должно выполнять следующие функции:

1. Создайте проект Python Django с использованием стандартных средств Django.
2. Создайте модель Django ORM, содержащую две сущности, связанные отношением один-ко-многим в соответствии с Вашим вариантом из условий рубежного контроля №1.
3. С использованием стандартного механизма Django сгенерируйте по модели макет веб-приложения, позволяющий добавлять, редактировать и удалять данные.
4. Создайте представление и шаблон, формирующий отчет, который содержит соединение данных из двух таблиц.

Текст программы:

views.py

```
from django.shortcuts import render
from rest_framework import viewsets
from orch.serializers import OrchSerializer
from orch.serializers import DirSerializer
from orch.models import Orch
from orch.models import Dir
```

```
class OrchViewSet(viewsets.ModelViewSet):
```

```
    """
    API endpoint, который позволяет просматривать и редактировать акции
    компаний
    """
```

```
    # queryset всех пользователей для фильтрации по дате последнего изменения
    queryset = Orch.objects.all()
    serializer_class = OrchSerializer # Сериализатор для модели
```

```
class DirViewSet(viewsets.ModelViewSet):
```

```
    """
    API endpoint, который позволяет просматривать и редактировать акции
    компаний
    """
```

```
    # queryset всех пользователей для фильтрации по дате последнего изменения
```

```

queryset = Dir.objects.all()
serializer_class = DirSerializer # Сериализатор для модели

def report(request):
    return render(request, 'page.html',
    {'data': {'Dirs': Dir.objects.select_related('idDirOrch').order_by('idDir')}})

```

serializers.py

```

from orch.models import Orch
from orch.models import Dir
from rest_framework import serializers

```

```

class DirSerializer(serializers.ModelSerializer):
    class Meta:
        # Модель, которую мы сериализуем
        model = Dir
        # Поля, которые мы сериализуем
        fields = ["idDir", "fioDir", "salDir", "idDirOrch"]

```

```

class OrchSerializer(serializers.ModelSerializer):
    class Meta:
        model = Orch
        fields = ['idOrch', 'nameOrch']

```

models.py

```

from django.db import models

```

```

class Dir(models.Model):
    idDir = models.IntegerField(db_column='idDir', primary_key=True) # Field
name made lowercase.
    fioDir = models.CharField(db_column='fioDir', max_length=100) # Field name
made lowercase.
    salDir = models.FloatField(db_column='salDir') # Field name made lowercase.
    idDirOrch = models.ForeignKey('Orch', models.DO_NOTHING,
db_column='idDirOrch') # Field name made lowercase.

    class Meta:
        managed = False
        db_table = 'dir'

```

```

class DjangoMigrations(models.Model):
    id = models.BigAutoField(primary_key=True)
    app = models.CharField(max_length=255)
    name = models.CharField(max_length=255)
    applied = models.DateTimeField()

    class Meta:
        managed = False
        db_table = 'django_migrations'

class Orch(models.Model):
    idorch = models.IntegerField(db_column='idOrch', primary_key=True) # Field
name made lowercase.
    nameorch = models.CharField(db_column='nameOrch', max_length=45,
blank=True, null=True) # Field name made lowercase.

    class Meta:
        managed = False
        db_table = 'orch'

```

settings.py

```

INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',

    # DRF
    'rest_framework',

    # Наше приложение
    'orch'
]

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': 'MyDB',
        'USER': 'root',
        'PASSWORD': '123',
        'HOST': 'localhost',

```

```

        'PORT': 3306, # Стандартный порт MySQL
        'OPTIONS': {'charset': 'utf8'},
        'TEST_CHARSET': 'utf8',
    }
}

```

urls.py

```

from django.contrib import admin
from orch import views as dir_views
from django.urls import include, path
from rest_framework import routers

router = routers.DefaultRouter()
router.register(r'dirs', dir_views.DirViewSet)
router.register(r'orchs', dir_views.OrchViewSet)

# Wire up our API using automatic URL routing.
# Additionally, we include login URLs for the browsable API.
urlpatterns = [
    path("", include(router.urls)),
    path('api-auth/', include('rest_framework.urls', namespace='rest_framework')),
    path('page/', dir_views.report),
    path('admin/', admin.site.urls),
]

```

base.html

```

<!doctype html>
<html lang="en" class="h-100">
<head>
    <meta charset="utf-8">
    <title>{% block title %}{% endblock %}</title>
</head>
<body>
    {% block content %}{% endblock %}
</body>
</html>

```

page.html

```

{% extends 'base.html' %}
{% block title %} Дирижеры {% endblock %}
{% block content %}
<table>

```

```

<tr>
  <th>ID</th>
  <th>ФИО</th>
  <th>Зарплата</th>
  <th>Оркестр</th>
</tr>
{% for dir in data.Dirs %}
<tr>
  <th>{{ dir.iddir }}</th>
  <td>{{ dir.fiodir }}</td>
  <td>{{ dir.saldir }}</td>
  <td>{{ dir.iddirorch.nameorch }}</td>
</tr>
{% endfor %}
</table>
{% endblock %}

```

Результат выполнения программы:

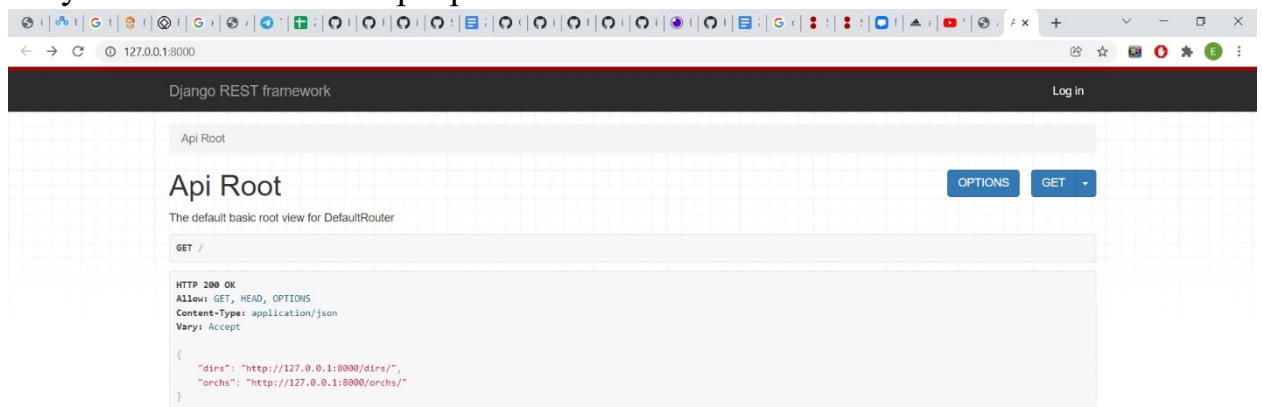


Рис. 1. Главная страница

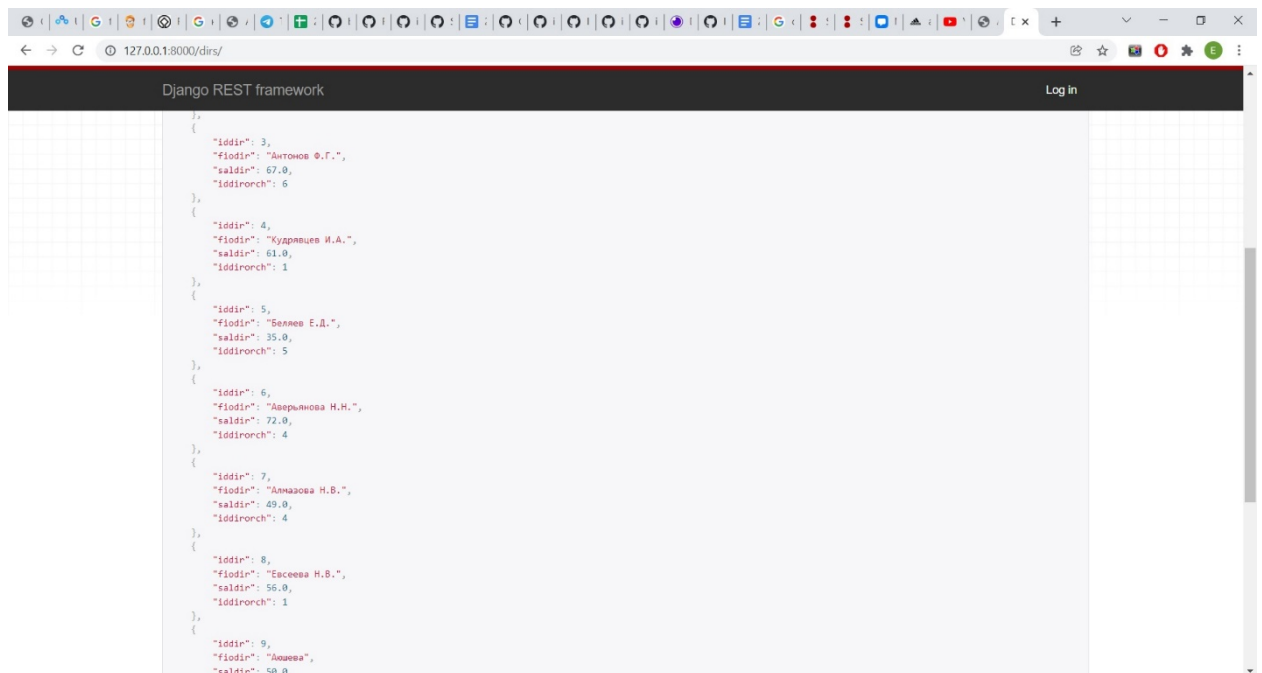


Рис. 2. Дирижеры

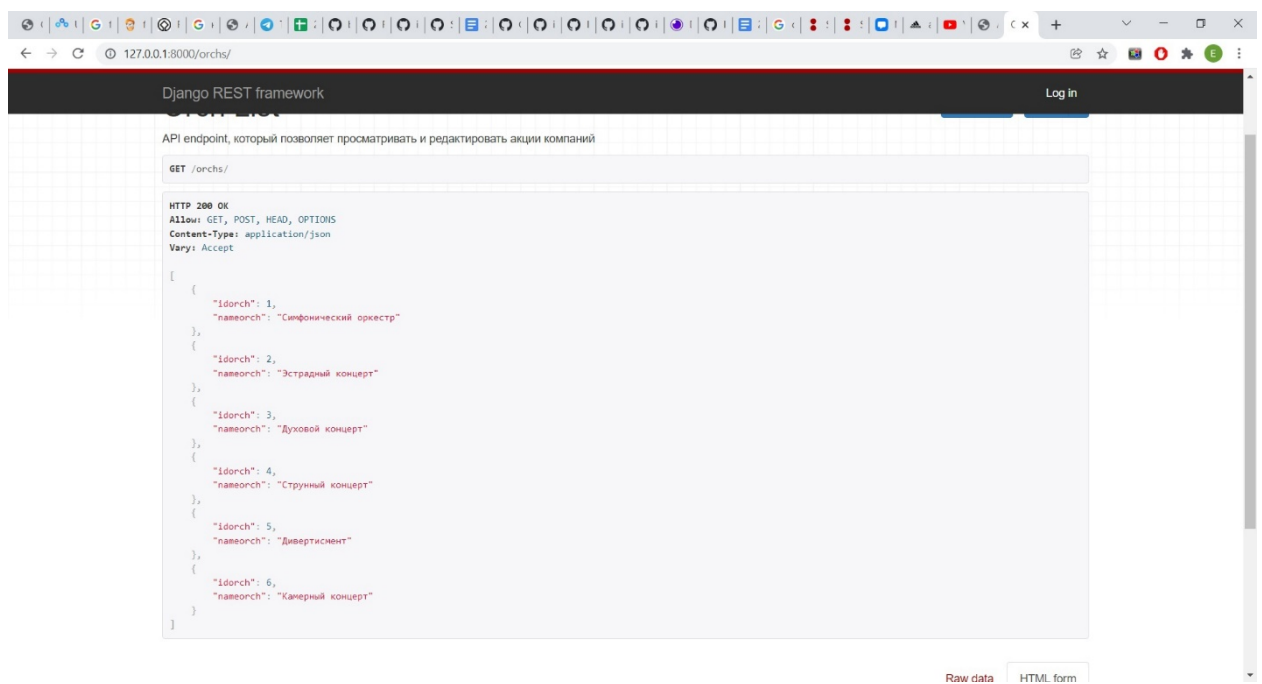


Рис. 3. Оркестры

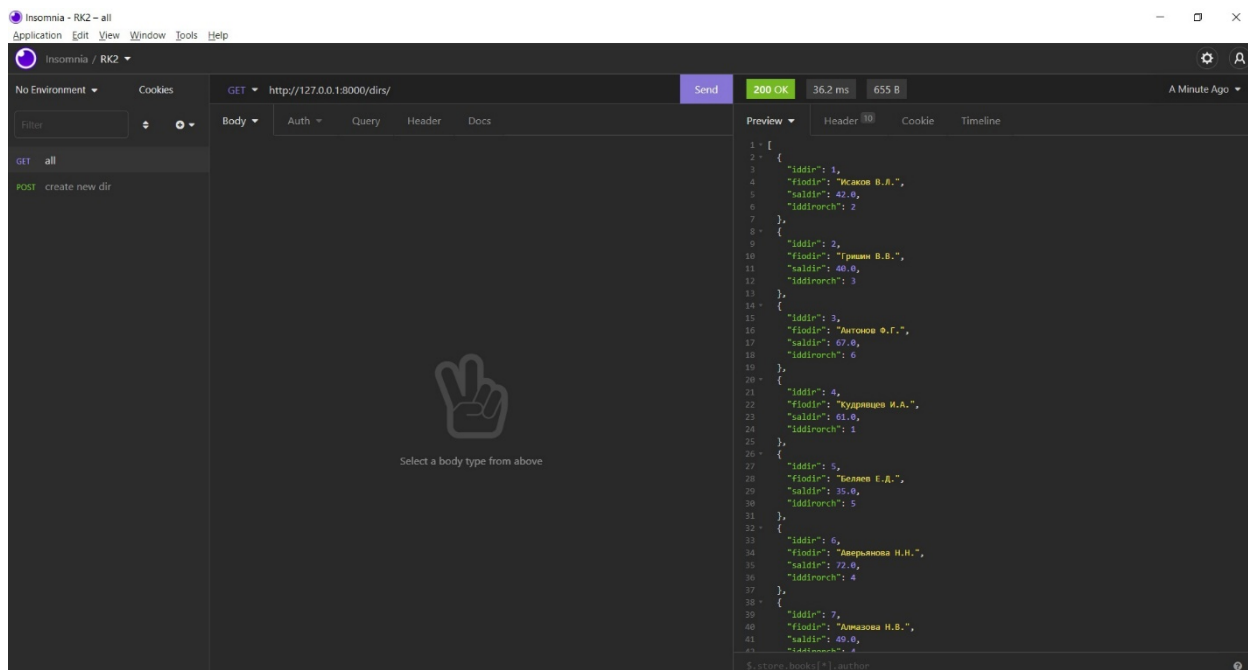


Рис. 3. Список всех дирижеров через Insomnia

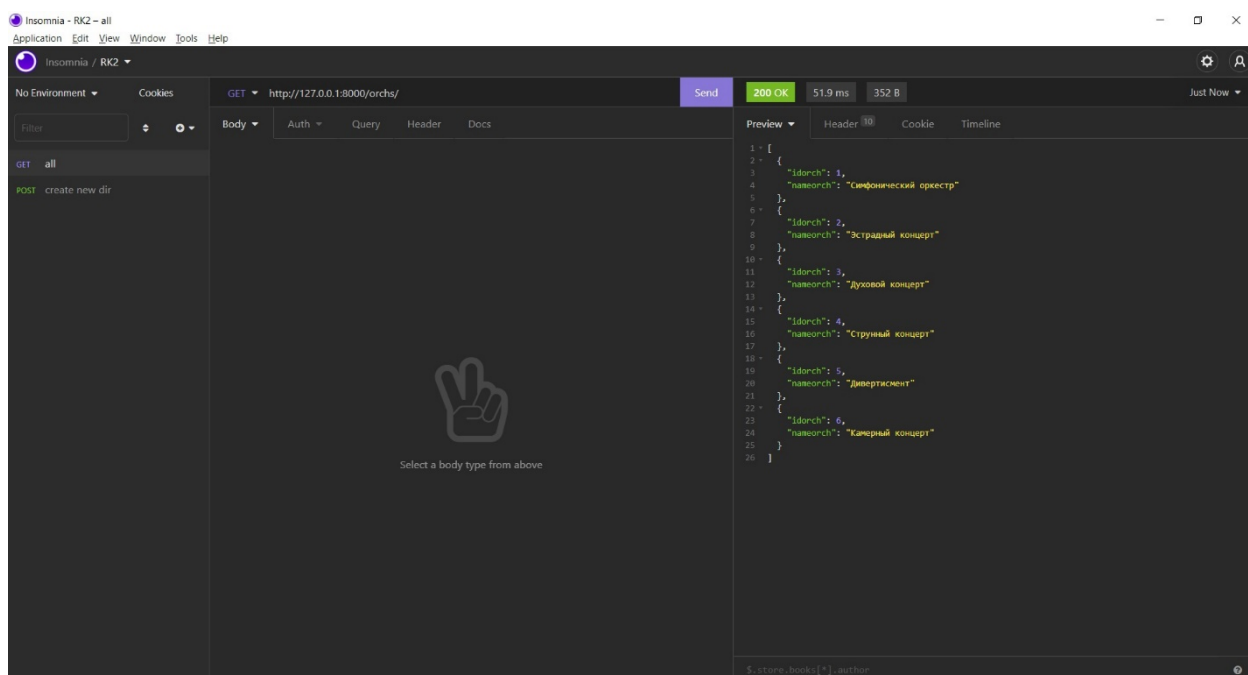


Рис. 4. Список всех оркестров через Insomnia

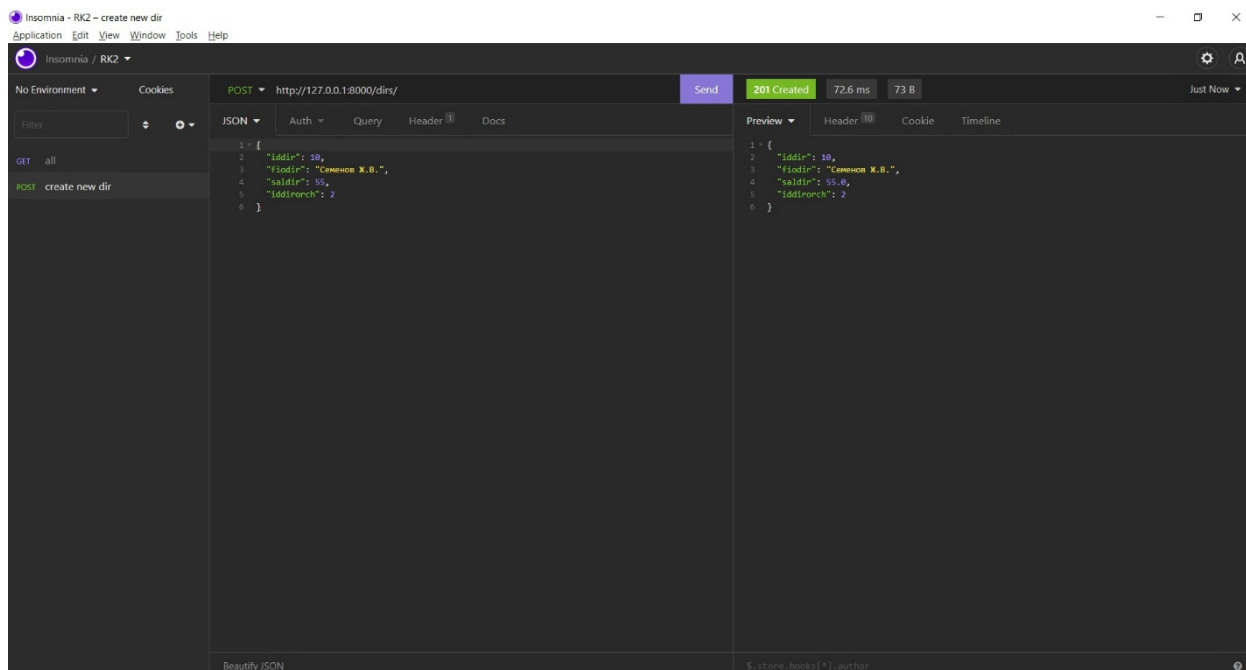


Рис. 5. Добавление нового дирижера

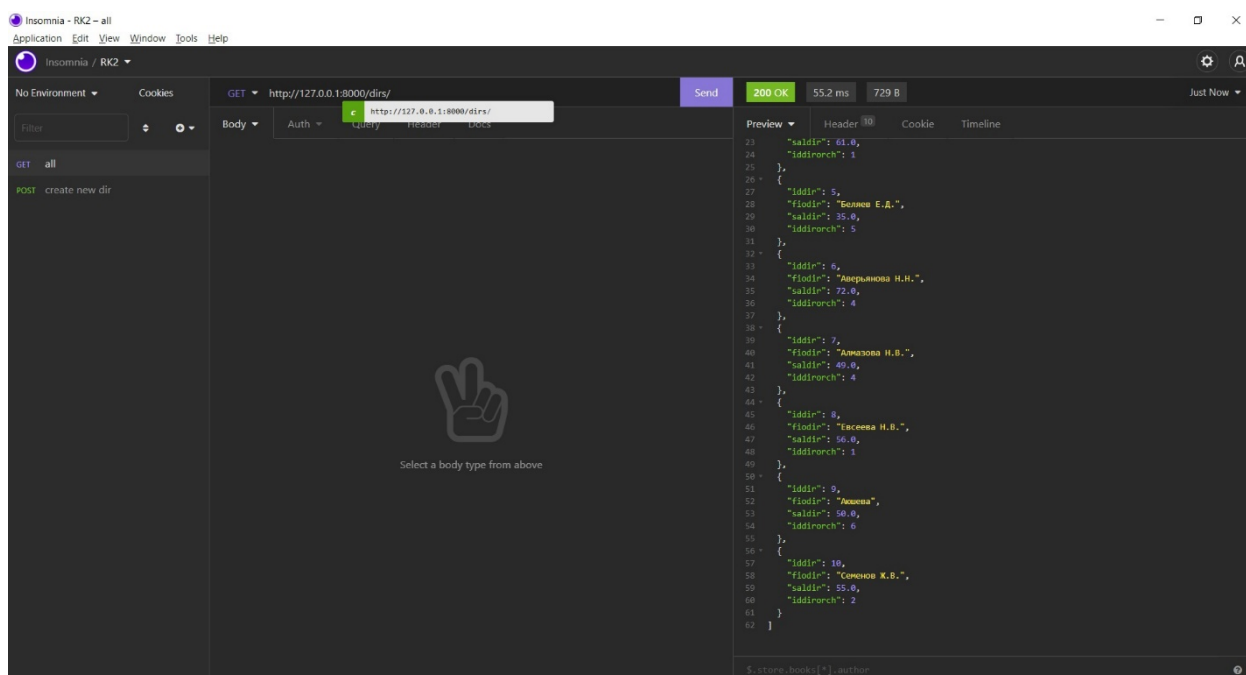


Рис. 6. Список дирижеров после добавления нового

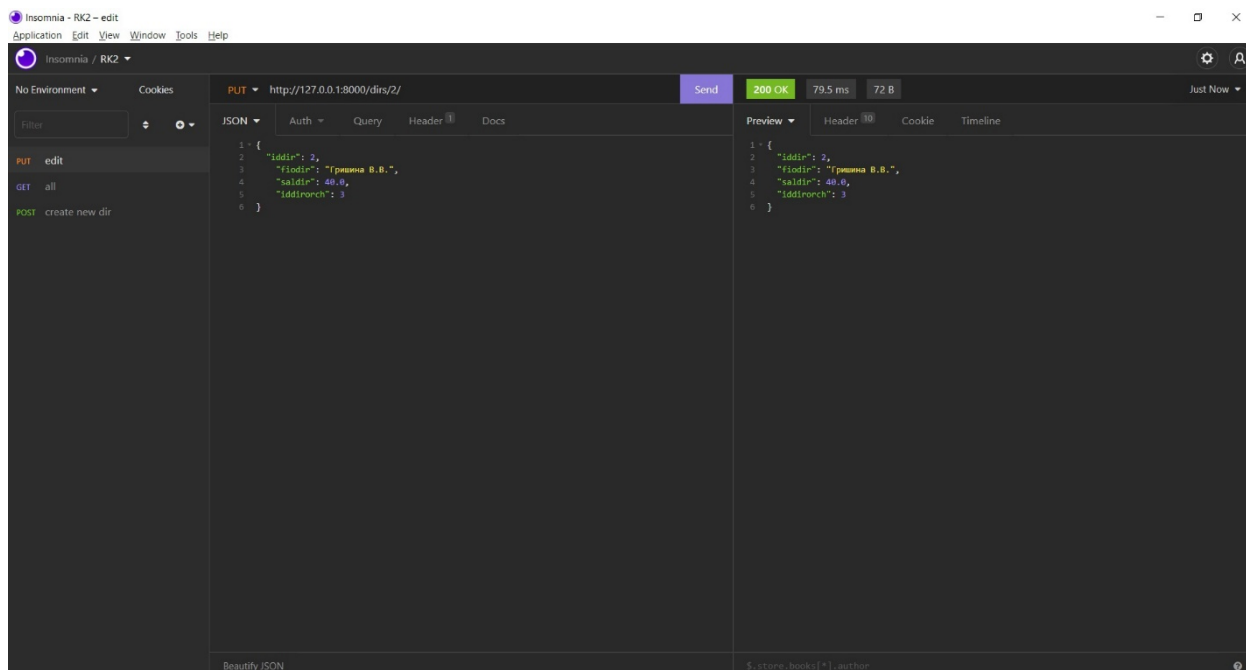


Рис. 7. Изменение 2-ого дирижера

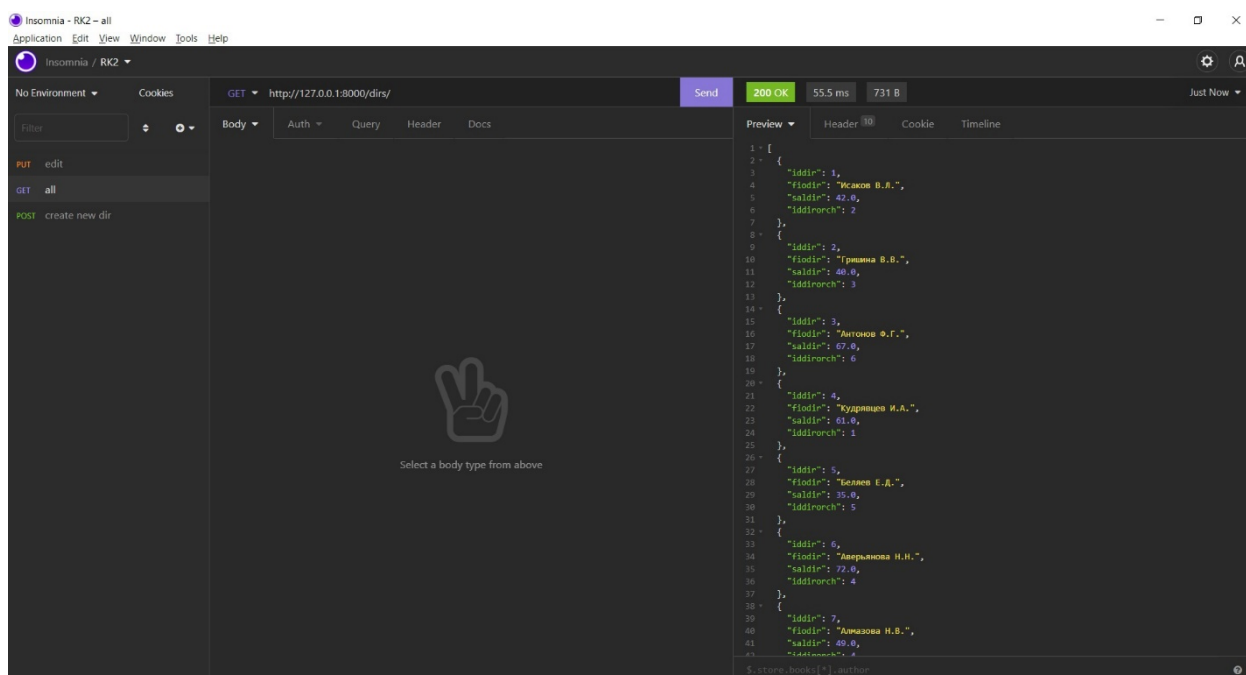


Рис. 8. Список дирижеров после изменения 2 дирижера

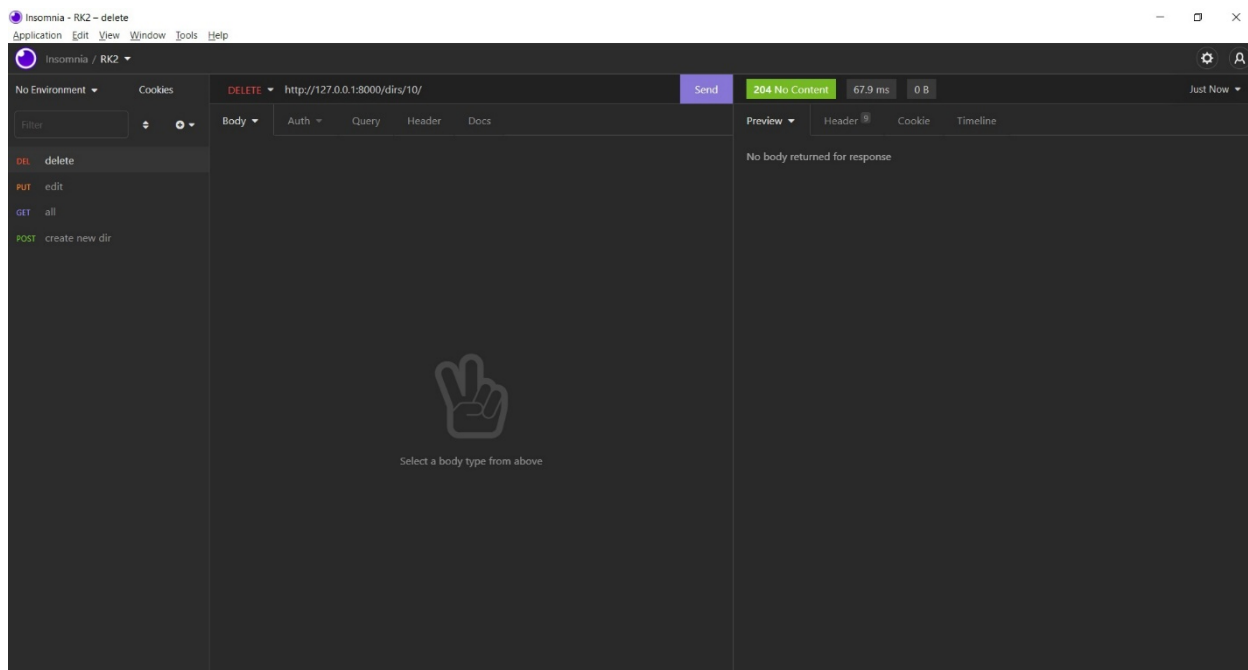


Рис. 9. Удаление 10-ого дирижера

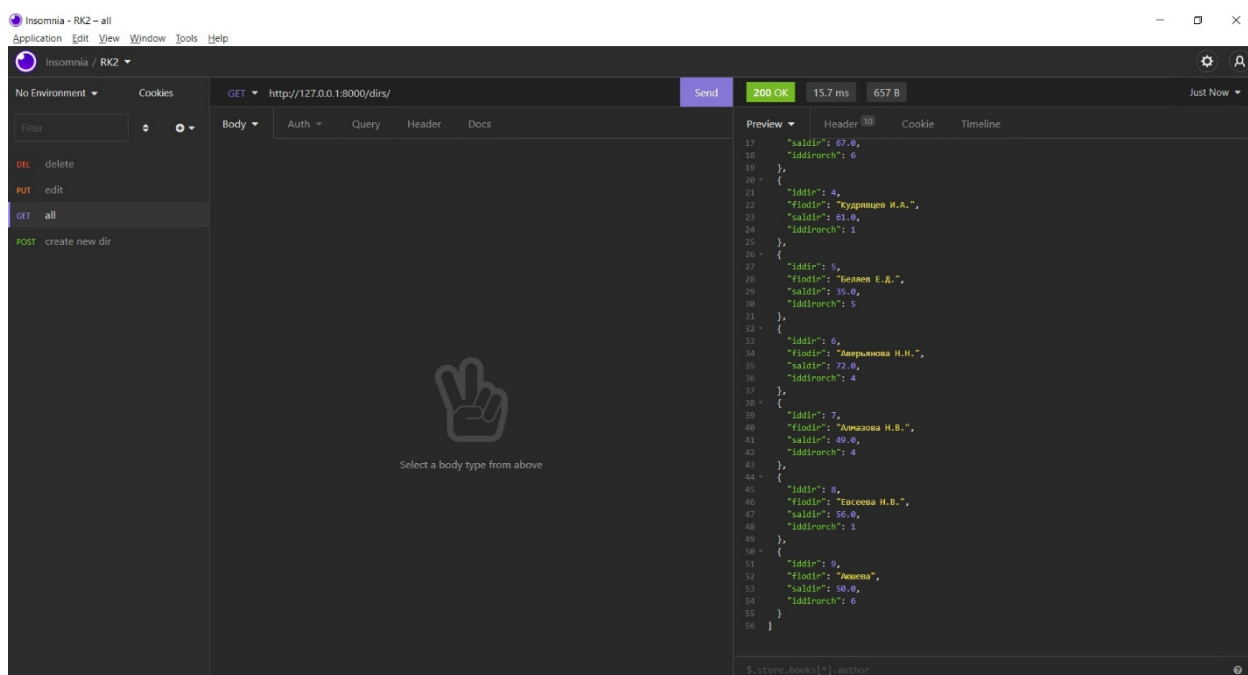
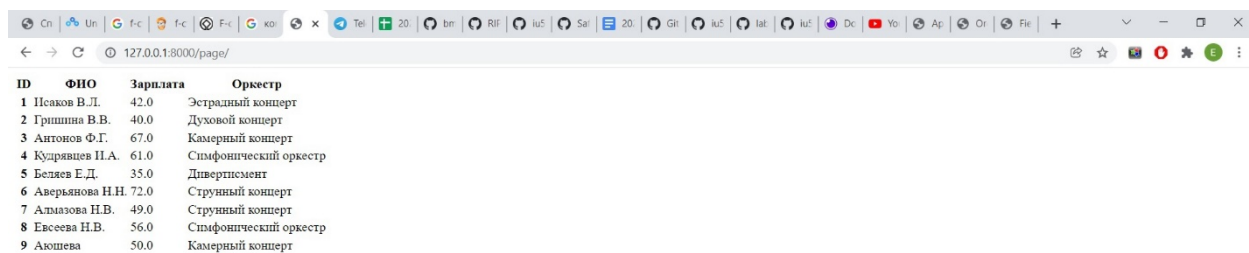


Рис. 10. Список дирижеров после удаления 10-ого дирижера



ID	ФИО	Зарплата	Оркестр
1	Исakov В.Л.	42.0	Эстрадный концерт
2	Гришина В.В.	40.0	Духовой концерт
3	Антонов Ф.Г.	67.0	Камерный концерт
4	Кудрявцев П.А.	61.0	Симфонический оркестр
5	Беллев Е.Д.	35.0	Дивертисмент
6	Аверьянова Н.Н.	72.0	Струнный концерт
7	Алмазова Н.В.	49.0	Струнный концерт
8	Евсеева Н.В.	56.0	Симфонический оркестр
9	Аюшева	50.0	Камерный концерт

Рис. 11. Отчет с данными из двух таблиц