	<p align="center"> 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) </p>
---	--

ФАКУЛЬТЕТ ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ
 КАФЕДРА СИСТЕМЫ ОБРАБОТКИ ИНФОРМАЦИИ И УПРАВЛЕНИЯ

Отчет по РК2

Студент Драгун Илья Алексеевич
фамилия, имя, отчество
 Группа ИУ5-54Б

Студент 18.12.2021 Драгун И.А.
подпись, дата *фамилия, и.о.*
 Преподаватель Гапанюк Ю.Е.
подпись, дата *фамилия, и.о.*

Задание:

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.

Код:

Код проекта rk:

models.py:

```
from django.db import models

class Orchestra(models.Model):
    name = models.CharField(max_length=255)

    class Meta:
        managed = False
        db_table = 'tbl_orchestra'

class Musician(models.Model):
    name = models.CharField(max_length=255)
    age = models.IntegerField()
    orchestra_id = models.IntegerField()

    class Meta:
        managed = False
        db_table = 'tbl_musicians'
```

urls.py:

```
from django.urls import path, include
from . import views

urlpatterns = [
    path('', views.index, name='home'),
    path('api/', include('api.urls')),
    path('orchestra/<int:id>/', views.GetOrchestra, name='orchestra_url')
]
```

views.py:

```
from django.shortcuts import render
from .models import Orchestra, Musician

def GetOrchestra(request, id):
    return render(request, 'main/orchestra.html', {'data' : {
        'orchestra': Orchestra.objects.filter(id=id)[0],
        'musicians': Musician.objects.filter(orchestra_id=id),
        'orchestras': Orchestra.objects.all(),
    }})
```

```
def index(request):
    return render(request, 'main/index.html', {'data' : {
        'orchestras': Orchestra.objects.all()
    }})
```

index.html:

```
{% extends 'main/layout.html' %}

{% block title %}Главная страница{% endblock %}
{% block content %}
    <div class="features">
        <p>
            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.
        </p>
    </div>
{% endblock %}
```

layout.html:

```
{% load static %}
<!doctype html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport"
        content="width=device-width, user-scalable=no, initial-scale=1.0,
        maximum-scale=1.0, minimum-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>{% block title %} {% endblock %}</title>
    <link rel="stylesheet"
        href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css"
        ">
    <link rel="stylesheet" href="{% static 'main/css/main.css' %}">
    <link rel="stylesheet"
        href="https://use.fontawesome.com/releases/v5.8.2/css/all.css">
</head>
<body>
    <aside>
        <h3>Список оркестров</h3>
        <ul>
            <a href="{% url 'home' %}"><li><i class="fas fa-
            home"></i>Главная</li></a>
            {% for orchestra in data.orchestras %}
            <a href="{% url 'orchestra_url'
            orchestra.id %}"><li>{{orchestra.name}}</li></a>
            {% endfor %}
        </ul>
    </aside>
    <main>
        {% block content %}
        {% endblock %}
    </main>
```

```
</body>
</html>
```

orchestra.html:

```
{% extends 'main/layout.html' %}

{% block title %}{{data.orchestra.name}}{% endblock %}
{% block content %}

    <div class="features">
        <h3>Музыканты в оркестре {{data.orchestra.name}}:</h3>
        {% for musician in data.musicians %}
            <div>{{musician.name}}, {{musician.age}} y.o.</div>
        {% endfor %}

    </div>
{% endblock %}
```

Код арі части:

serializers.py

```
from rest_framework import serializers
from rk.models import Orchestra, Musician

class OrchestraSerializer(serializers.ModelSerializer):
    name = serializers.CharField(max_length=255)

    class Meta:
        model = Orchestra
        fields = [
            'id', 'name'
        ]

class MusicianSerializer(serializers.ModelSerializer):
    name = serializers.CharField(max_length=255)
    age = serializers.IntegerField()
    orchestra_id = serializers.IntegerField()

    class Meta:
        model = Musician
        fields = [
            'id', 'name', 'age', 'orchestra_id'
        ]
```

urls.py

```
from django.urls import path, include
from .views_api import OrchestraListAPIView, MusicianListAPIView
from rest_framework import routers

router = routers.DefaultRouter()
router.register(r'orchestras', OrchestraListAPIView)
router.register(r'musicians', MusicianListAPIView)
urlpatterns = [
    path('', include(router.urls)),
    path('api-auth/', include('rest_framework.urls',
namespace='rest_framework'))
]
```

views_api.py

```
from rest_framework import viewsets
from rk.models import Orchestra, Musician
from .serializers import OrchestraSerializer, MusicianSerializer
```

```
class OrchestraListAPIView(viewsets.ModelViewSet):

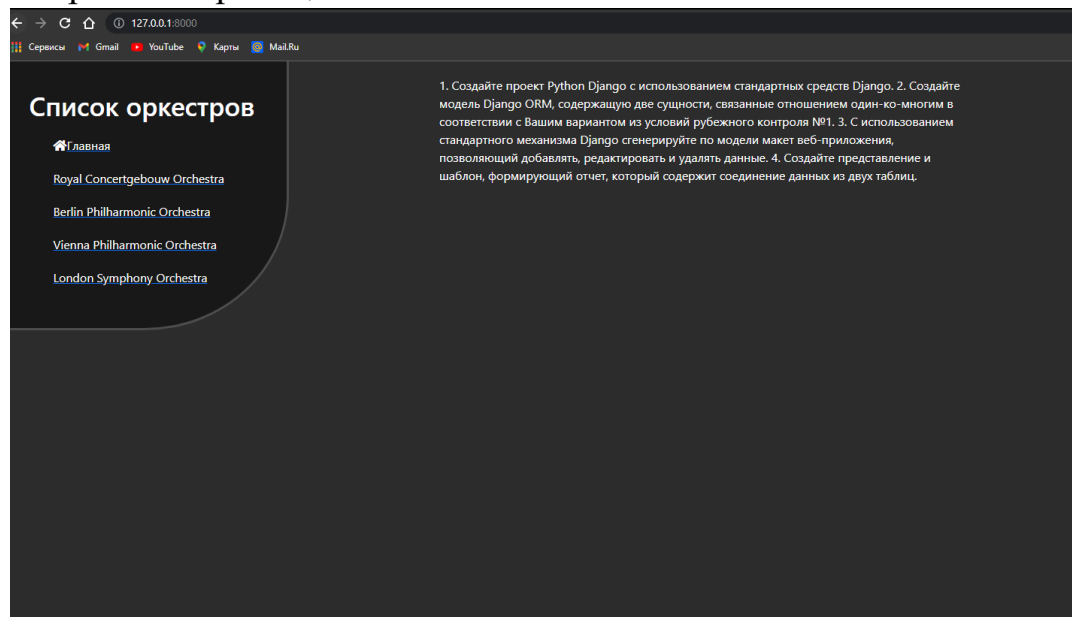
    serializer_class = OrchestraSerializer
    queryset = Orchestra.objects.all()

class MusicianListAPIView(viewsets.ModelViewSet):

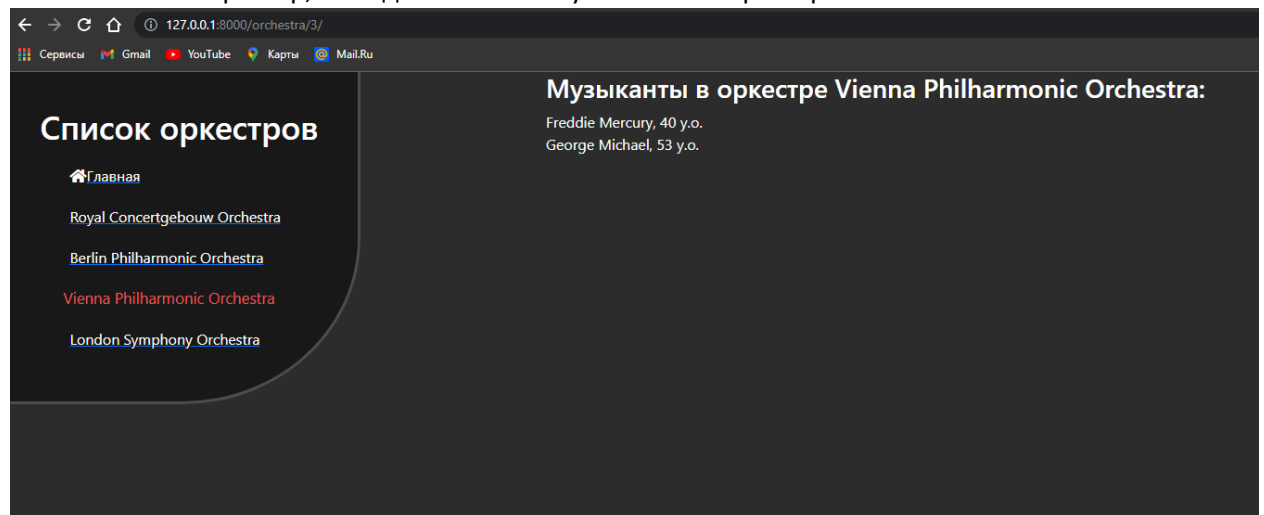
    serializer_class = MusicianSerializer
    queryset = Musician.objects.all()
```

Демонстрация:

Стартовая страница



По нажатию на оркестр, выводится список музыкантов в оркестре



Запросы GET и POST к api:

GET

http://127.0.0.1:8000/api/musicians/

Send

200 OK

123 ms

462 B

Just Now

Body

Auth

Query

Header

Docs

Preview

Header

Cookie

Timeline

Select a body type from above

```
1 * [  
2 * {  
3   "id": 1,  
4   "name": "Muse",  
5   "age": 30,  
6   "orchestra_id": 4  
7 },  
8 * {  
9   "id": 2,  
10  "name": "Coldplay",  
11  "age": 25,  
12  "orchestra_id": 2  
13 },  
14 * {  
15   "id": 3,  
16   "name": "Freddie Mercury",  
17   "age": 40,  
18   "orchestra_id": 3  
19 },  
20 * {  
21   "id": 4,  
22   "name": "Michael Philip Jagger",  
23   "age": 55,  
24   "orchestra_id": 1  
25 },  
26 * {  
27   "id": 5,  
28   "name": "George Michael",  
29   "age": 53,  
30   "orchestra_id": 3  
31 },  
32 ]
```

POST

http://127.0.0.1:8000/api/orchestras/

Send

201 Created

22.2 ms

68 B

JSON

Auth

Query

Header

Docs

Preview

Header

Cookie

Timeline

```
1 * {  
2   "id": 1,  
3   "name": "Оркестр Мариинского театра"  
4 }
```

```
1 * {  
2   "id": 5,  
3   "name": "Оркестр Мариинского театра"  
4 }
```

GET

http://127.0.0.1:8000/api/orchestras/

Send

200 OK

10.1 ms

258 B

JSON

Auth

Query

Header

Docs

Preview

Header

Cookie

Timeline

```
1 * {  
2   "id": 1,  
3   "name": "Оркестр Мариинского театра"  
4 }
```

```
1 * [  
2 * {  
3   "id": 1,  
4   "name": "Royal Concertgebouw Orchestra"  
5 },  
6 * {  
7   "id": 2,  
8   "name": "Berlin Philharmonic Orchestra"  
9 },  
10 * {  
11   "id": 3,  
12   "name": "Vienna Philharmonic Orchestra"  
13 },  
14 * {  
15   "id": 4,  
16   "name": "London Symphony Orchestra"  
17 },  
18 * {  
19   "id": 5,  
20   "name": "Оркестр Мариинского театра"  
21 }  
22 ]
```