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 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.

2021г.

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 api 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'))
]
```

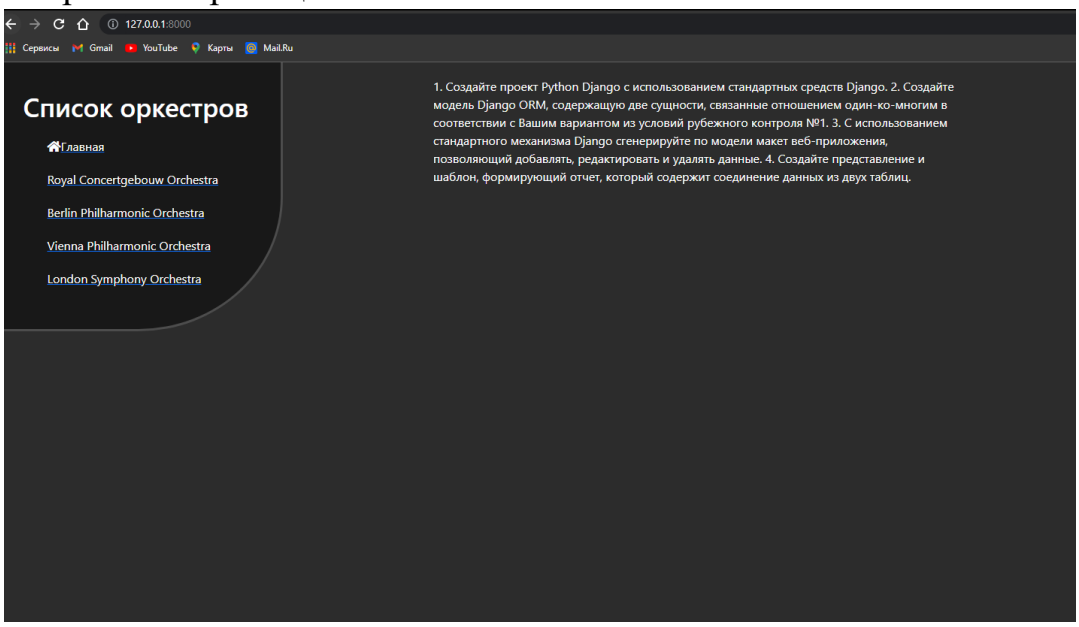
views_api.py

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

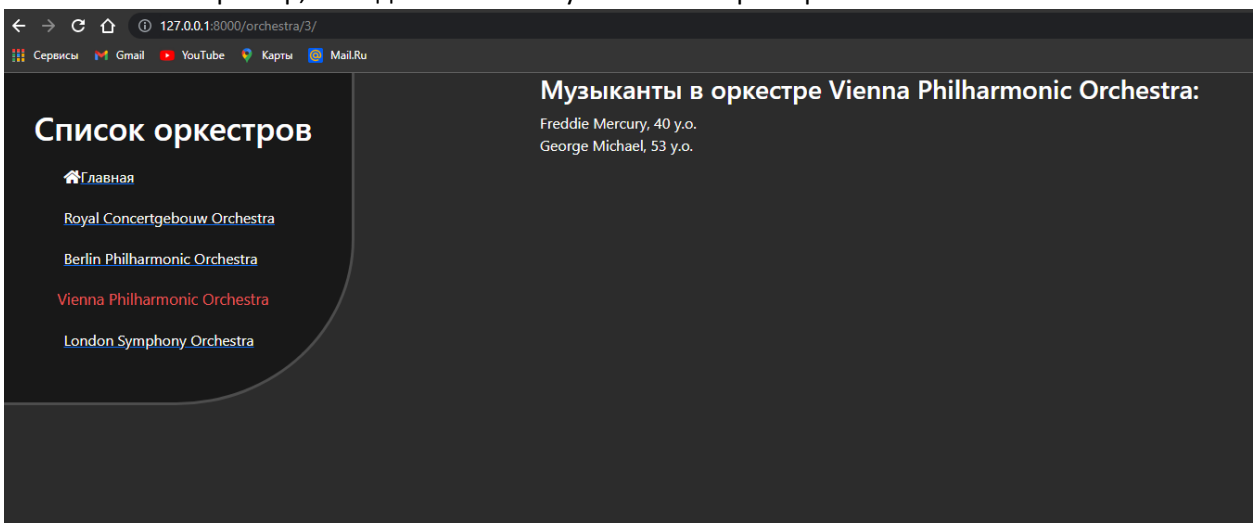
```
class OrchestraListAPIView(viewsets.ModelViewSet):

    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: