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 вычислительная техника»
Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Разработка интернет-приложений»

Отчет по РК№2

Выполнил:
студент группы ИУ5-52Б

Олейников И.И.
Подпись и дата:

Проверил:
преподаватель каф. ИУ5

Гапанюк Ю.Е.
Подпись и дата:

Москва, 2021 г.

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.

17	Дирижер	Оркестр

Код программы:

models.py

```python
from django.db import models


class Conductor(models.Model):
    id = models.IntegerField(primary_key=True)
    name = models.CharField(max_length=20)

    class Meta:
        db_table = 'conductor'


class Orcestra(models.Model):
    id = models.IntegerField(primary_key=True)
    name = models.TextField()
    size = models.FloatField()
    cond_id = models.ForeignKey(Conductor, on_delete=models.PROTECT)

    class Meta:
        db_table = 'orcestra'
```

views.py

```python
from django.shortcuts import render
from rest_framework import viewsets

from dir.models import Conductor, Orcestra
from dir.serializers import ConductorSerializer, OrcestraSerializer


class ConductorViewSet(viewsets.ModelViewSet):
    queryset = Conductor.objects.all()
    serializer_class = PCSerializer


class OrcestraViewSet(viewsets.ModelViewSet):
    queryset = Orcestra.objects.all()
    serializer_class = DisplaySerializer


def report(request):
    return render(request, 'report.html', {'data': {
        'orcestra': Orcestra.objects.select_related('conductor')
    }})
```

settings.py

```python
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql_psycopg2',
        'NAME': 'rk2_bd',
        'USER': 'iilya',
        'PASSWORD': '12345',
        'HOST': 'localhost',
        'PORT': 3306,
    }
}
```

urls.py

```python
router = routers.DefaultRouter()
router.register('orcestra', views.ConductorViewSet)
router.register('conductor', views.OrcestraViewSet)

urlpatterns = [

    path('', include(router.urls)),
    path('report/', views.report),
    path('api-auth/', include('rest_framework.urls', namespace='rest_framework')),
    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>
```

details.html

```
{% extends 'base.html' %}

{% block title %}Компьютер{% endblock %}

{% block content %}
    <div>
        <h1>Дисплей:</h1>
    </div>
    <ul>
        {% for disp in data.display %}
            <li>Матрица:  <i>{{ disp.matrix_type }} </i></li>
            <li>Диагональ: <i>{{ disp.diagonal }}</i></li>
            <li>Название компьютера: <i>{{ disp.pc.name }}</i></li>
            <br>
        {% empty %}
            <li>Список пуст</li>
        {% endfor %}
    </ul>
{% endblock %}
```

Результат работы программы:

Оркестры:

- Название: zolip
  Размер: 6
  Дирижер: Илья
- Название: malin
  Размер: 12
  Дирижер: Александр
- Название: golic
  Размер: 64
  Дирижер: Илья
- Название: zoya
  Размер: 18
  Дирижер: Адам
- Название: bolca
  Размер: 3
  Дирижер: Михаил