Отчет по РК № 2 по курсу "Разработка Интернет-Приложений"

Выполнил: Студент группы

ИУ5-55Б

Зубарева А. М.

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.

«Музыкант» и «Оркестр» связаны соотношением один-ко-многим. Выведите список всех Музыкантов, у которых фамилия заканчивается на «ов», и названия их Оркестров.

«Музыкант» и «Оркестр» связаны соотношением один-ко-многим. Выведите список Оркестр со средней зарплатой Музыкантов в каждом Оркестре, отсортированный по средней зарплате.

№ варианта	Класс 1	Класс 2
5	Музыкант	Оркестр

Текст программы:

rk2/urls.py

```
from django.contrib import admin
from django.urls import path, include
from musicians import views

urlpatterns = [
   path('admin/', admin.site.urls),
   path('', views.index, name="main"),
   path('musician/', include([
        path('', views.read_musician, name='read_musician'),
        path('create/', views.create_musician, name="create_musician"),
        path('update/<int:musician_id>/', views.update_musician,
name="update_musician"),
```

musicians/models.py

```
class Orchestra(models.Model):
    name = models.CharField(max_length=256, verbose_name="Orchestra name")

def __str__(self):
    return self.name

class Musician (models.Model):
    name = models.CharField(max_length=256, verbose_name="Musician name")
    salary = models.PositiveIntegerField(verbose_name="Musician's salary")
    orchestra = models.ForeignKey(
        Orchestra,
        on_delete=models.SET_DEFAULT,
        null=True,
        default=None,
        related_name="musicians"
)

def __str__(self):
    return self.name
```

musicians/views.py

```
from django.db.models.aggregates import Avg
from django.http.response import HttpResponse
from django.shortcuts import get_list_or_404, get_object_or_404, redirect, render
from .models import Orchestra, Musician
from django.db import models

def index(request):
    return render(request, 'index.html')

# musicians
```

```
def read musician(request):
def create_musician(request):
       return render(request, 'musician/create musician.html', {"orchestras": orchestras})
def update_musician(request, musician_id):
       return render(request, 'musician/update musician.html', {"musician": musician,
               setattr(musician, key, request.POST[key])
  return redirect(request.META.get('HTTP REFERER'))
def read orchestra(request):
  return render(request, 'orchestra/orchestra list.html', {'orchestras': orchestras})
```

```
return render(request, 'orchestra/create orchestra.html')
      new orchestra = Orchestra(name=request.POST['name'])
def update orchestra(request, orchestra id):
       return render(request, 'orchestra/update_orchestra.html', {"orchestra": orchestra})
              setattr(orchestra, key, request.POST[key])
  orchestra = get object or 404 (Orchestra, pk=orchestra id)
  return redirect(request.META.get('HTTP REFERER'))
def report(request):
       average salaries.append({"orchestra": orchestra, "salary":
round(Musician.objects.filter(
          orchestra=orchestra.pk).aggregate(Avg('salary'))['salary avg'] or 0)})
  return render(request, 'report.html', {"found musicians": found musicians,
'average_salaries": sorted(average_salaries, key=lambda salary: salary['salary'])})
```

templates/inc/base.html

```
<meta http-equiv="cache-control" content="no-cache";</pre>
  <title>
  </title>
  </header>
          </div>
crossorigin="anonymous"></script>
```

templates/inc/navbar.html

templates/musician/musician_card.html

templates/musician/musician_form.html

```
<div class="mb-3">
     <label for="musicianName" class="form-label">Musician Name</label>
```

templates/musician/musician_list.html

templates/musician/create_musician.html

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

templates/musician/update musician.html

templates/orchestra/orchestra_card.html

templates/orchestra/orchestra_form.html

templates/orchestra/orchestra_list.html

```
{% for orchestra in orchestras %} {% include 'orchestra/orchestra_card.html' with orchestra=orchestra %} {% endfor %} 
</div>
{% endblock content %}
```

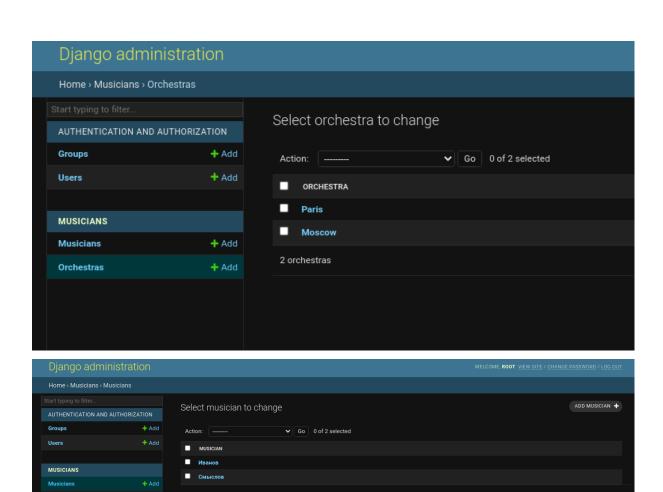
templates/orchestra/create orchestra.html

templates/orchestra/update_orchestra.html

templates/report.html

templates/index.html

Результат работы:



Main Page All musicians Create musician All orchestras Create orchestra Report

Main Page



