Московский государственный технический университет

им. Н. Э. Баумана

Факультет «Информатика и системы управления»

Отчёт по рубежному контролю №2
по курсу «Разработка интернет-приложений»

Выполнил:

студент группы ИУ5-51Б
Коротков Н.К.

Проверил:

преподаватель каф. ИУ5
Гапанюк Ю. Е.

Подпись и дата:

Подпись и дата:

Москва, 2021 г.

Описание задания:

1. Создайте проект Python Django с использованием стандартных средствDjango.
2. Создайте модель Django ORM, содержащую две сущности, связанныеотношением 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, редактироватьи удалять данные.
4. Создайте представление и шаблон, формирующий отчет, которыйсодержит соединение данных из двух таблиц.

№ варианта	Класс 1	Класс 2
11	Программа	Компьютер

Ход выполнения работы:

models.py:

```
from django.db import models

class Computer(models.Model):
    name = models.CharField(max_length=255)
    cost = models.IntegerField()

    class Meta:
        managed = False
        db_table = 'computers'

class Program(models.Model):
    name = models.CharField(max_length=255)
    size = models.IntegerField()
    computer_id = models.IntegerField()

    class Meta:
        managed = False
```

urls.py:

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

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

views.py:

```
from django.shortcuts import render
from .models import Computer, Program

def GetComputer(request, id):
    return render(request, 'main/computer.html', {'data' : {
        'computer': Computer.objects.filter(id=id)[0],
        'programs': Program.objects.filter(computer_id=id),
        'computers': Computer.objects.all(),
    }})

def index(request):
    return render(request, 'main/index.html', {'data' : {
        'computers': Computer.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 компьютеров</h3>
        <ul>
            <a href="{% url 'home'%}"><li><i class="fas fa-
home"></i>Главная</li></a>
            {% for computer in data.computers %}
            <a href="{% url 'computer_url'
computer.id%}"><li>{{computer.name}}</li></a>
            {% endfor %}
        </ul>
    </aside>
    <main>
    {% block content %}
    {% endblock %}
    </main>
</body>
</html>
```

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

computer.html:

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

serializers.py

```
from rest_framework import serializers
from rk.models import Computer, Program

class ComputerSerializer(serializers.ModelSerializer):
    name = serializers.CharField(max_length=255)
    cost = serializers.IntegerField()
    class Meta:
        model = Computer
        fields =[
            'id', 'name', 'cost'
        ]

class ProgramSerializer(serializers.ModelSerializer):
    name = serializers.CharField(max_length=255)
    age = serializers.IntegerField()
    computer_id = serializers.IntegerField()

    class Meta:
        model = Program
        fields =[
            'id', 'name','size', 'computer_id'
        ]
```

urls.py

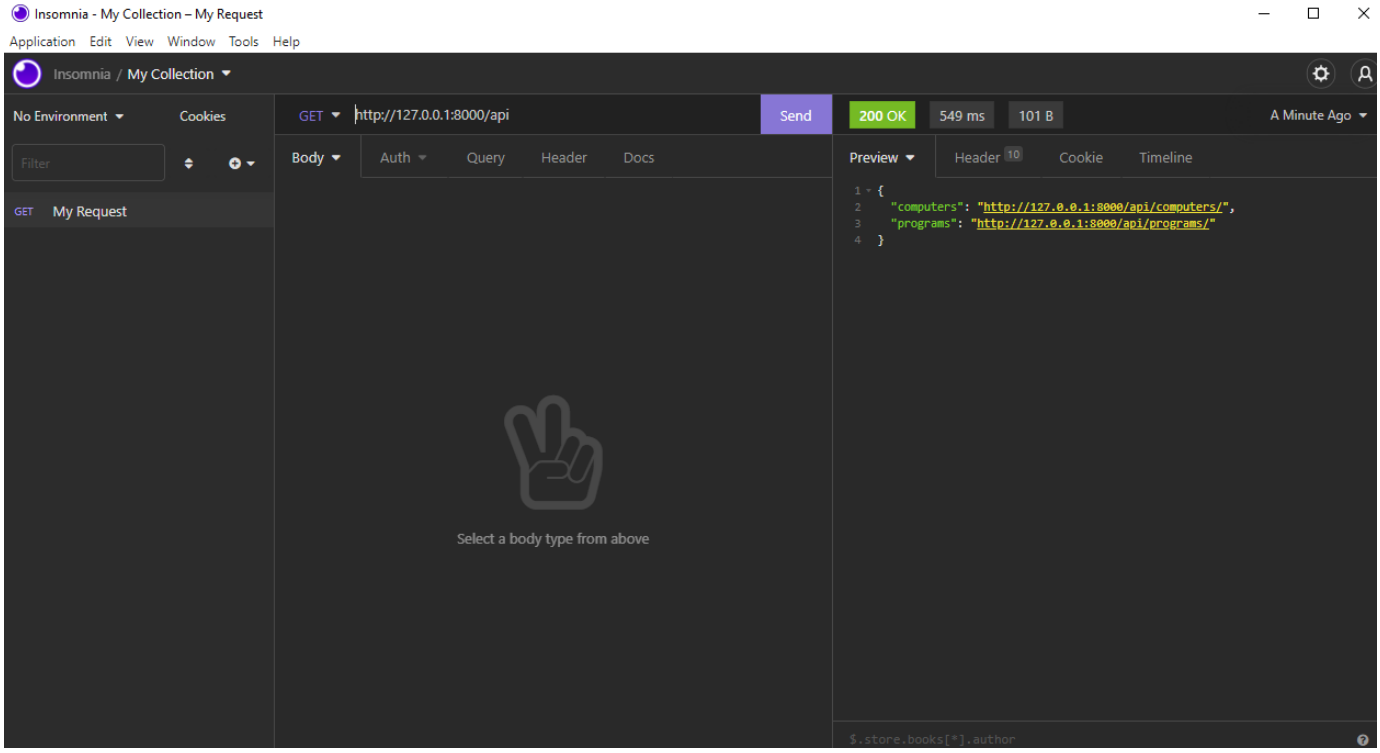
```
from django.urls import path, include
from .views_api import ComputerListAPIView, ProgramListAPIView
from rest_framework import routers

router = routers.DefaultRouter()
router.register(r'computers', ComputerListAPIView)
router.register(r'programs', ProgramListAPIView)
urlpatterns = [
    path('', include(router.urls)),
    path('api-auth/', include('rest_framework.urls',
namespace='rest_framework'))
]
```

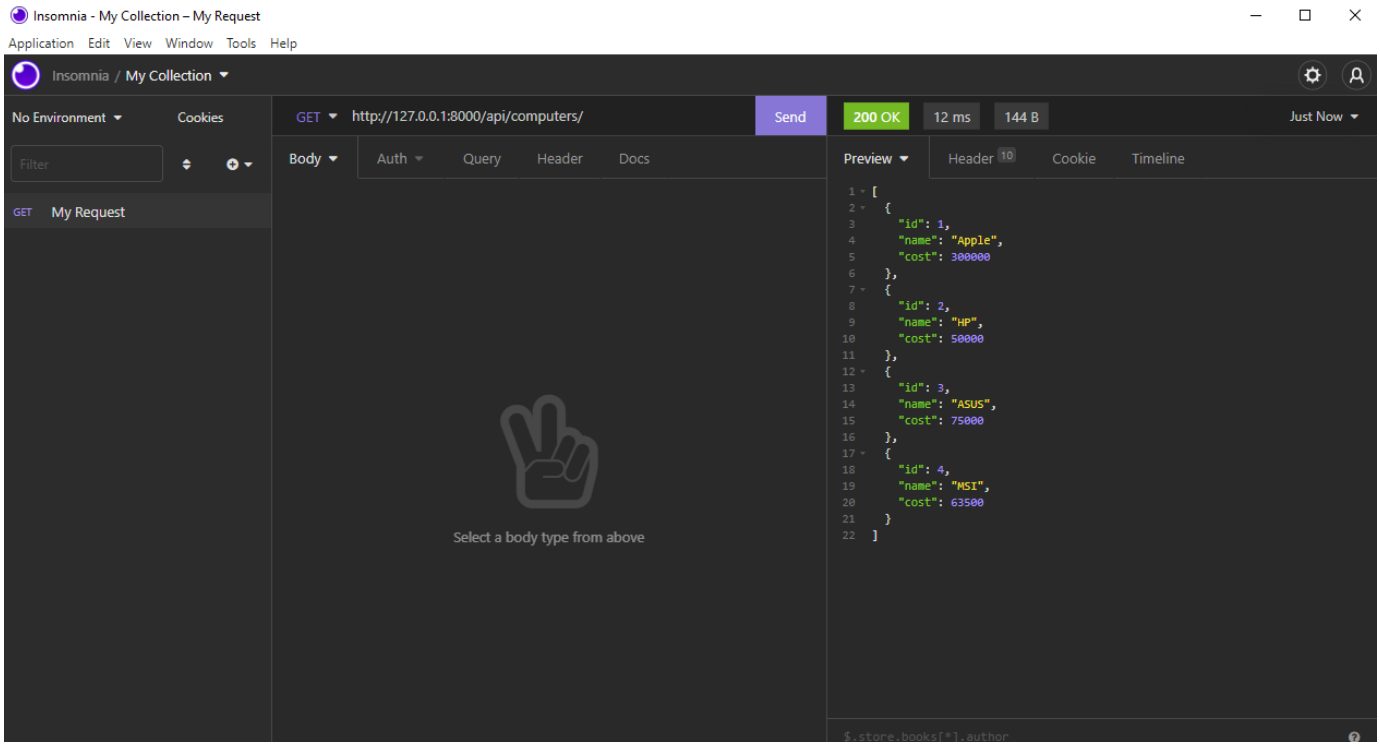
views_api.py

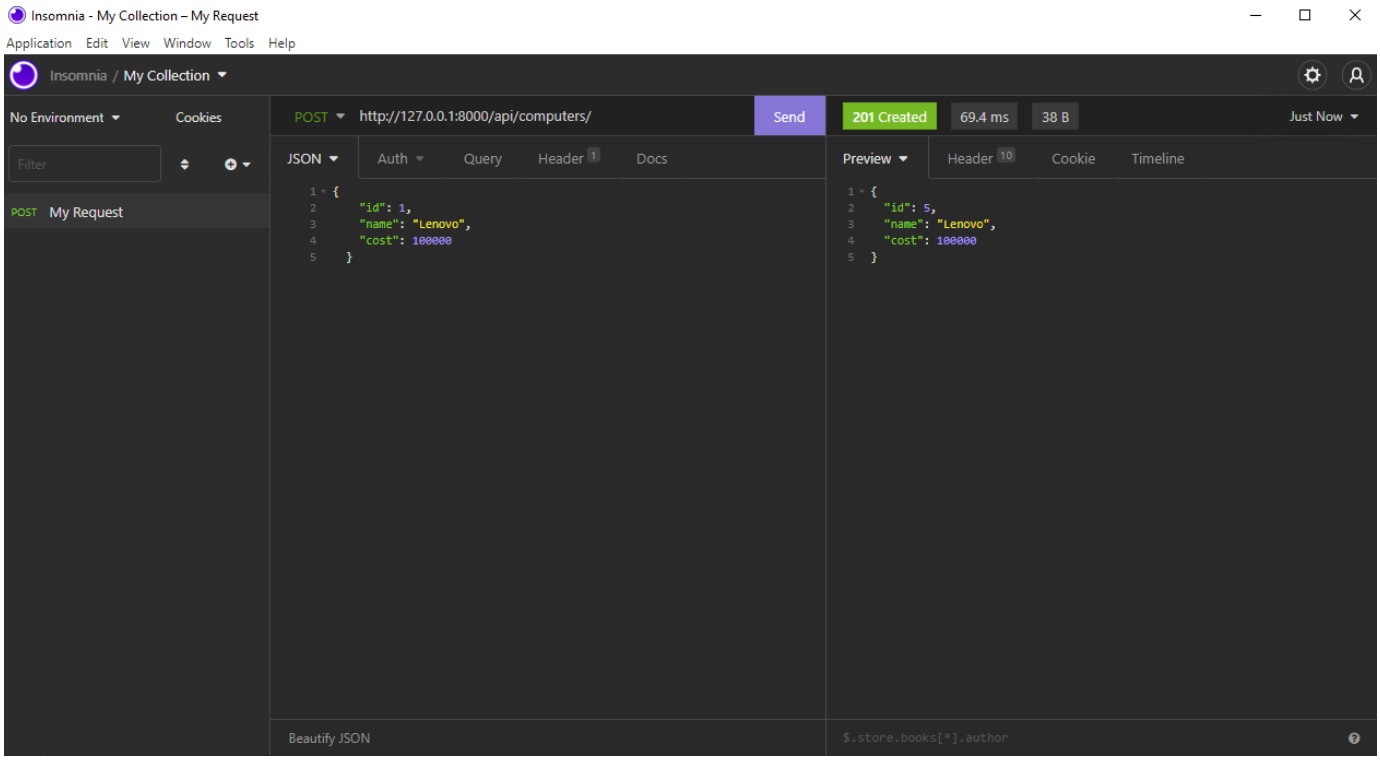
```
from rest_framework import viewsets
from rk.models import Computer, Program
```

```
from .serializers import ComputerSerializer, ProgramSerializer
class ComputerListAPIView(viewsets.ModelViewSet):

    serializer_class = ComputerSerializer
    queryset = Computer.objects.all()

class ProgramListAPIView(viewsets.ModelViewSet):

    serializer_class = ProgramSerializer
    queryset = Program.objects.all()
```

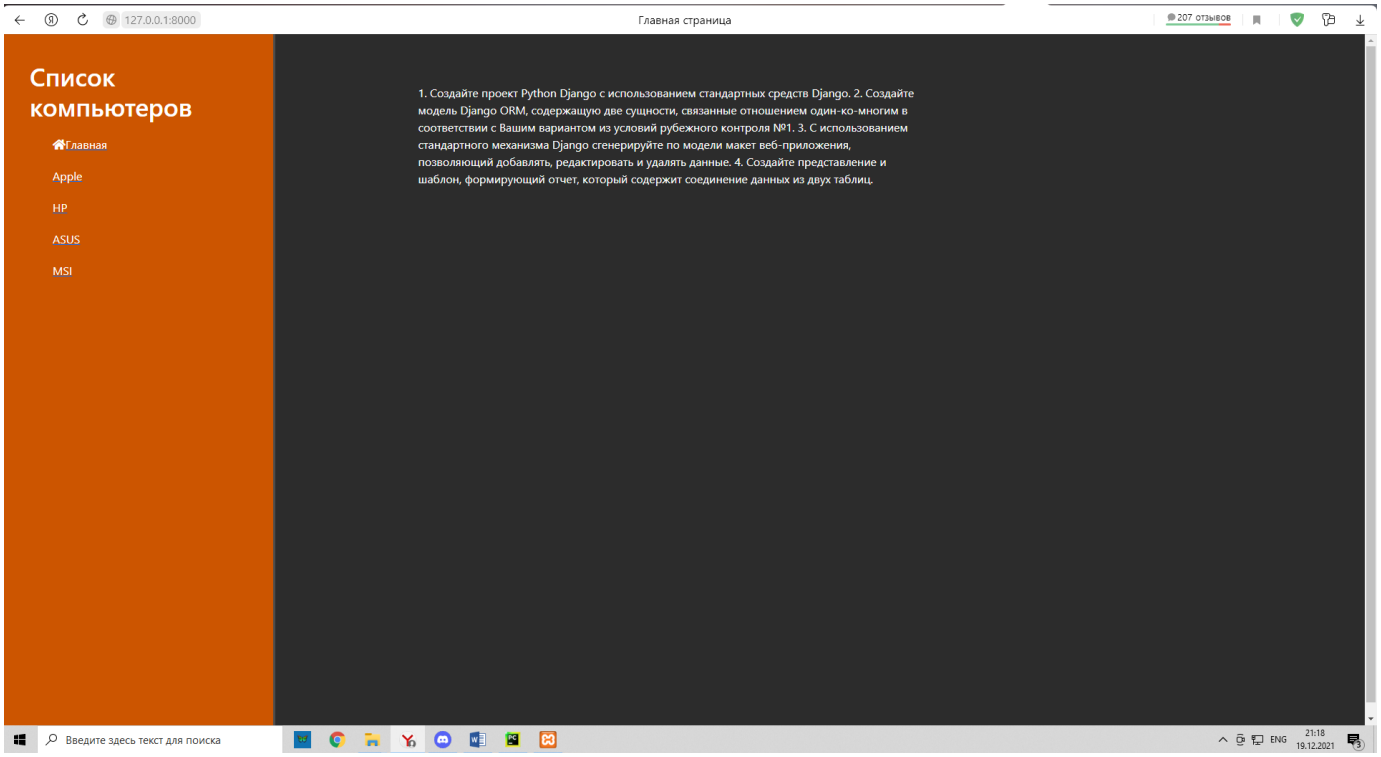
Добавление данных с помощью Insomnia

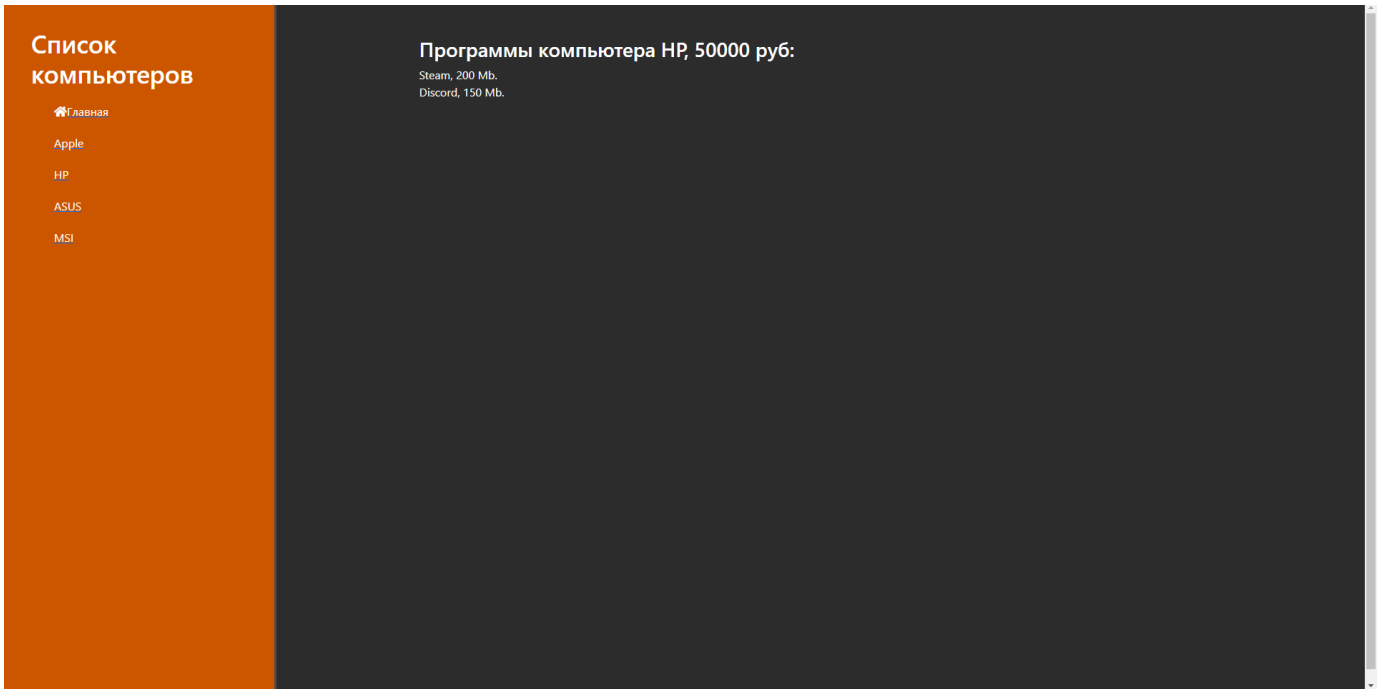


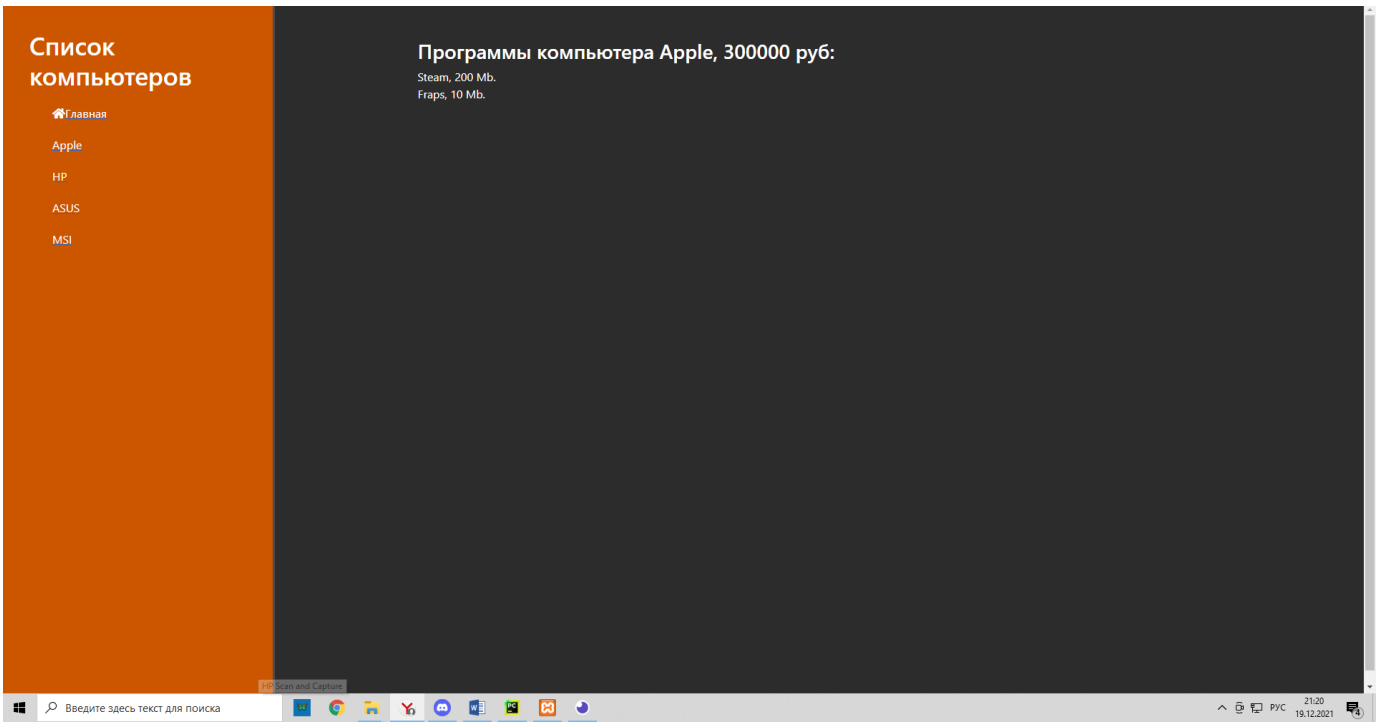



Insomnia - My Collection – My Request
Application Edit View Window Tools Help
Insomnia / My Collection
No Environment
Cookies
Filter
POST My Request
POST http://127.0.0.1:8000/api/computers/
Send
JSON
Auth
Query
Header 1
Docs
{
"id": 1,
"name": "Lenovo",
"cost": 100000
}
Beautify JSON
201 Created
69.4 ms
38 B
Just Now
Preview
Header 10
Cookie
Timeline
{
"id": 5,
"name": "Lenovo",
"cost": 100000
}
$.store.books[*].author

Скриншоты


127.0.0.1:8000
Главная страница
207 отзывов
Список компьютеров
Главная
Apple
HP
ASUS
MSI
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.
Введите здесь текст для поиска
ENG
21:18
19.12.2021


Список компьютеров
Главная
Apple
HP
ASUS
MSI
Программы компьютера HP, 50000 руб:
Steam, 200 Mb.
Discord, 150 Mb.


Список компьютеров
Главная
Apple
HP
ASUS
MSI
Программы компьютера Apple, 300000 руб:
Steam, 200 Mb.
Fraps, 10 Mb.
Введите здесь текст для поиска
РУС
21:20
19.12.2021