**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 лабораторной работе №6

«Разработка REST API с использованием Django REST Framework.»

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-52Б |  | преподаватель каф. ИУ5 |
| Веревкина Диана В. |  | Гапанюк Ю.Е. |
| Подпись и дата: |  | Подпись и дата: |

Москва, 2021 г.

# ****Цель лабораторной работы****

# Изучение возможностей разработки REST API с использованием Django REST Framework.

# Описание задания

С использованием Django REST Framework разработайте REST API для одной модели (одной таблицы базы данных).

# Текст программы

**initial.py**

# Generated by Django 4.0 on 2021-12-22 19:40  
  
from django.db import migrations, models  
import django.db.models.deletion  
  
  
class Migration(migrations.Migration):  
  
 initial = True  
  
 dependencies = [  
 ]  
  
 operations = [  
 migrations.CreateModel(  
 name='Type',  
 fields=[  
 ('id', models.BigAutoField(auto\_created=True, primary\_key=True, serialize=False, verbose\_name='ID')),  
 ('TypeName', models.CharField(max\_length=100)),  
 ],  
 ),  
 migrations.CreateModel(  
 name='Cat',  
 fields=[  
 ('id', models.BigAutoField(auto\_created=True, primary\_key=True, serialize=False, verbose\_name='ID')),  
 ('CatName', models.CharField(max\_length=100)),  
 ('CatBirthday', models.IntegerField()),  
 ('TypeId', models.ForeignKey(on\_delete=django.db.models.deletion.CASCADE, to='siteDiana.type')),  
 ],  
 ),  
 ]

**admin.py**

from django.contrib import admin  
from .models import Cat, Type  
  
admin.site.register(Cat)  
admin.site.register(Type)

**apps.py**

from django.apps import AppConfig  
  
  
class SitedianaConfig(AppConfig):  
 default\_auto\_field = 'django.db.models.BigAutoField'  
 name = 'siteDiana'

**models.py**

from django.db import models  
  
  
class Type(models.Model):  
 TypeName = models.CharField(max\_length=100)  
 def \_\_str\_\_(self):  
 return self.TypeName  
  
  
class Cat(models.Model):  
 CatName = models.CharField(max\_length=100)  
 CatBirthday = models.IntegerField()  
 TypeId = models.ForeignKey(Type, on\_delete=models.CASCADE)  
 def \_\_str\_\_(self):  
 return self.CatName

**serializers.py**

from .models import Type  
from rest\_framework import serializers  
  
  
class ModelmySerializer(serializers.ModelSerializer):  
 class Meta:  
 # Модель, которую мы сериализуем  
 model = Type  
 # Поля, которые мы сериализуем  
 fields = ["TypeName"]

**urls.py**

from django.urls import include, path  
from rest\_framework import routers  
from rest\_framework.urlpatterns import format\_suffix\_patterns  
  
from . import views  
  
router = routers.DefaultRouter()  
router.register(r'model', views.ModelmyViewSet)  
  
# Wire up our API using automatic URL routing.  
# Additionally, we include login URLs for the browsable API.  
urlpatterns = [  
 path('', include(router.urls)),  
 path('api-auth/', include('rest\_framework.urls', namespace='rest\_framework'))  
]

**views.py**

from django.shortcuts import render  
from .models import Cat, Type  
  
def typeList(request):  
 data = {'tp': Type.objects.all()}  
 return render(request, 'index.html', data)  
  
def catInfo(request, \_id):  
 data = {'t': Type.objects.get(id=\_id), 'cat': Cat.objects.filter(TypeId=\_id)}  
 return render(request, 'aboutCat.html', data)

**settings.py**

*"""  
Django settings for lab5 project.  
  
Generated by 'django-admin startproject' using Django 3.2.9.  
  
For more information on this file, see  
https://docs.djangoproject.com/en/3.2/topics/settings/  
  
For the full list of settings and their values, see  
https://docs.djangoproject.com/en/3.2/ref/settings/  
"""*from pathlib import Path  
  
# Build paths inside the project like this: BASE\_DIR / 'subdir'.  
BASE\_DIR = Path(\_\_file\_\_).resolve().parent.parent  
  
  
# Quick-start development settings - unsuitable for production  
# See https://docs.djangoproject.com/en/3.2/howto/deployment/checklist/  
  
# SECURITY WARNING: keep the secret key used in production secret!  
SECRET\_KEY = 'django-insecure-^da0\*b601\*m48cq0x+=ya5u=$lszff#yt5czjqduy2-2&5isle'  
  
# SECURITY WARNING: don't run with debug turned on in production!  
DEBUG = True  
  
ALLOWED\_HOSTS = []  
  
  
# Application definition  
  
INSTALLED\_APPS = [  
 'django.contrib.admin',  
 'django.contrib.auth',  
 'django.contrib.contenttypes',  
 'django.contrib.sessions',  
 'django.contrib.messages',  
 'django.contrib.staticfiles',  
 'siteDiana',  
 'rest\_framework',  
]  
  
MIDDLEWARE = [  
 'django.middleware.security.SecurityMiddleware',  
 'django.contrib.sessions.middleware.SessionMiddleware',  
 'django.middleware.common.CommonMiddleware',  
 'django.middleware.csrf.CsrfViewMiddleware',  
 'django.contrib.auth.middleware.AuthenticationMiddleware',  
 'django.contrib.messages.middleware.MessageMiddleware',  
 'django.middleware.clickjacking.XFrameOptionsMiddleware',  
]  
  
ROOT\_URLCONF = 'lab5.urls'  
  
TEMPLATES = [  
 {  
 'BACKEND': 'django.template.backends.django.DjangoTemplates',  
 'DIRS': ['templates'],  
 'APP\_DIRS': True,  
 'OPTIONS': {  
 'context\_processors': [  
 'django.template.context\_processors.debug',  
 'django.template.context\_processors.request',  
 'django.contrib.auth.context\_processors.auth',  
 'django.contrib.messages.context\_processors.messages',  
 ],  
 },  
 },  
]  
  
WSGI\_APPLICATION = 'lab5.wsgi.application'  
  
  
# Database  
# https://docs.djangoproject.com/en/3.2/ref/settings/#databases  
  
DATABASES = {  
 'default': {  
 'ENGINE': 'django.db.backends.mysql',  
 'NAME': 'BDlab5',  
 'USER': 'root',  
 'PASSWORD': 'Stimorol1',  
 'HOST': 'localhost',  
 'PORT': 3306, # Стандартный порт MySQL  
 'OPTIONS': {'charset': 'utf8'},  
 'TEST\_CHARSET': 'utf8',  
 }  
}  
  
  
# Password validation  
# https://docs.djangoproject.com/en/3.2/ref/settings/#auth-password-validators  
  
AUTH\_PASSWORD\_VALIDATORS = [  
 {  
 'NAME': 'django.contrib.auth.password\_validation.UserAttributeSimilarityValidator',  
 },  
 {  
 'NAME': 'django.contrib.auth.password\_validation.MinimumLengthValidator',  
 },  
 {  
 'NAME': 'django.contrib.auth.password\_validation.CommonPasswordValidator',  
 },  
 {  
 'NAME': 'django.contrib.auth.password\_validation.NumericPasswordValidator',  
 },  
]  
  
  
# Internationalization  
# https://docs.djangoproject.com/en/3.2/topics/i18n/  
  
LANGUAGE\_CODE = 'en-us'  
  
TIME\_ZONE = 'UTC'  
  
USE\_I18N = True  
  
USE\_L10N = True  
  
USE\_TZ = True  
  
  
# Static files (CSS, JavaScript, Images)  
# https://docs.djangoproject.com/en/3.2/howto/static-files/  
  
STATIC\_URL = '/static/'  
  
# Default primary key field type  
# https://docs.djangoproject.com/en/3.2/ref/settings/#default-auto-field  
  
DEFAULT\_AUTO\_FIELD = 'django.db.models.BigAutoField'

**urls.py**

*"""lab5 URL Configuration  
  
The `urlpatterns` list routes URLs to views. For more information please see:  
 https://docs.djangoproject.com/en/3.2/topics/http/urls/  
Examples:  
Function views  
 1. Add an import: from my\_app import views  
 2. Add a URL to urlpatterns: path('', views.home, name='home')  
Class-based views  
 1. Add an import: from other\_app.views import Home  
 2. Add a URL to urlpatterns: path('', Home.as\_view(), name='home')  
Including another URLconf  
 1. Import the include() function: from django.urls import include, path  
 2. Add a URL to urlpatterns: path('blog/', include('blog.urls'))  
"""*from django.contrib import admin  
from django.urls import path, re\_path, include  
from siteDiana import views  
  
urlpatterns = [  
 path('admin/', admin.site.urls),  
 path('', views.typeList),  
 path('<int:\_id>/', views.catInfo),  
 path('api/', include('siteDiana.urls')),  
  
]

**aboutCatl.html**

<!DOCTYPE html>  
<html lang="en">  
<head>  
 <meta charset="UTF-8">  
 <title>AboutCat</title>  
</head>  
<body>  
{% block content %}  
<ul>  
 <h2>Порода - {{t.TypeName}}</h2>  
 {% for i in cat %}  
 <li>Имя - {{i.CatName}}</li>  
 <li>Год рождения - {{i.CatBirthday}}</li>  
 <hr>  
 {% empty %}  
 <li>Список пуст</li>  
 {% endfor %}  
</ul>  
{% endblock %}  
</body>  
</html>

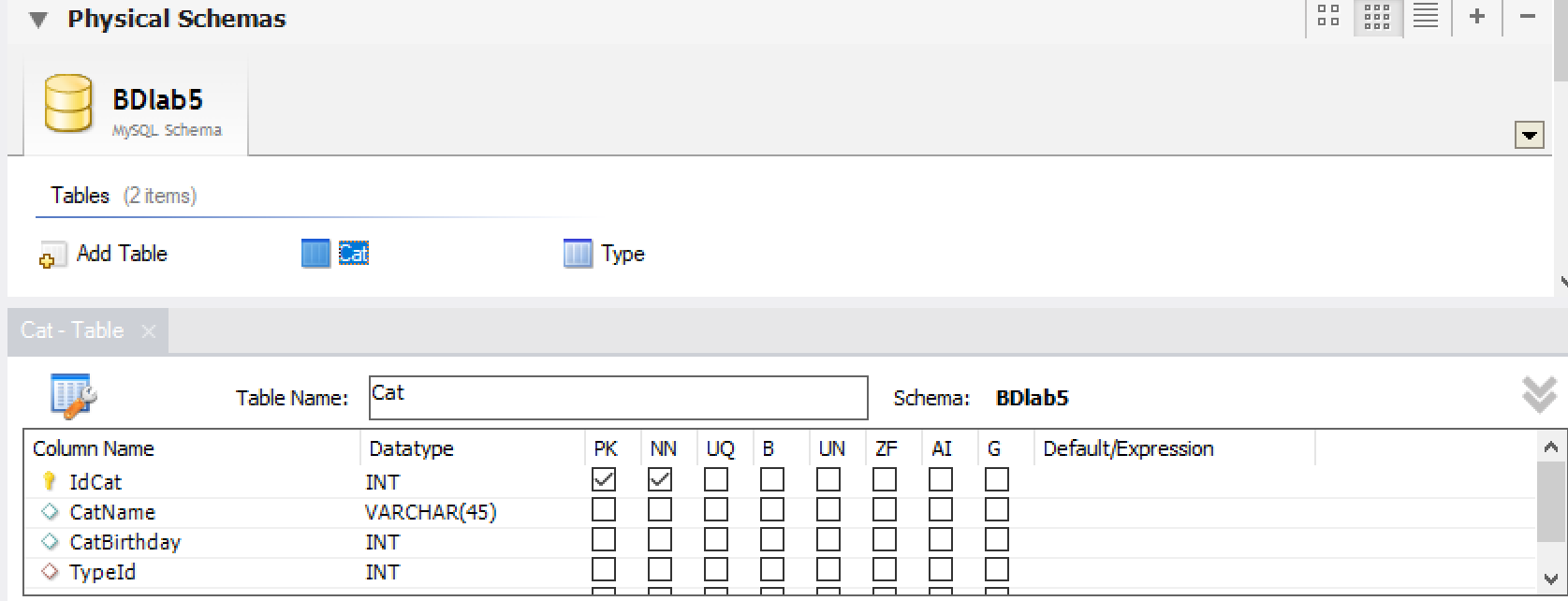
**index.html**

*"""lab5 URL Configuration  
  
The `urlpatterns` list routes URLs to views. For more information please see:  
 https://docs.djangoproject.com/en/3.2/topics/http/urls/  
Examples:  
Function views  
 1. Add an import: from my\_app import views  
 2. Add a URL to urlpatterns: path('', views.home, name='home')  
Class-based views  
 1. Add an import: from other\_app.views import Home  
 2. Add a URL to urlpatterns: path('', Home.as\_view(), name='home')  
Including another URLconf  
 1. Import the include() function: from django.urls import include, path  
 2. Add a URL to urlpatterns: path('blog/', include('blog.urls'))  
"""*from django.contrib import admin  
from django.urls import path  
from siteDiana import views  
  
urlpatterns = [  
 path('admin/', admin.site.urls),  
 path('', views.typeList),  
 path('<int:\_id>/', views.catInfo),  
]

# Экранные формы с примерами выполнения программы

**База данных**

Таблица «Cat»



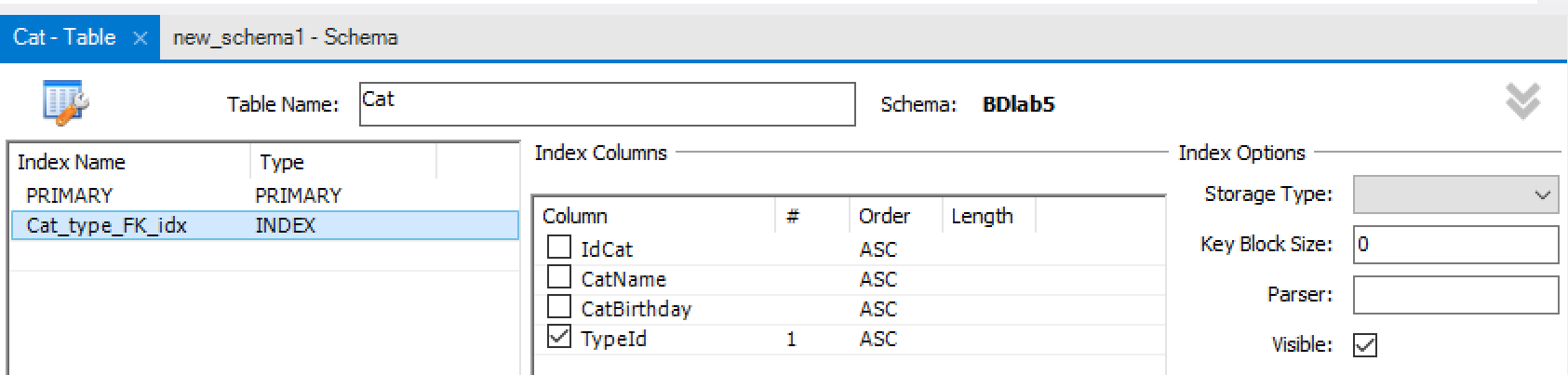
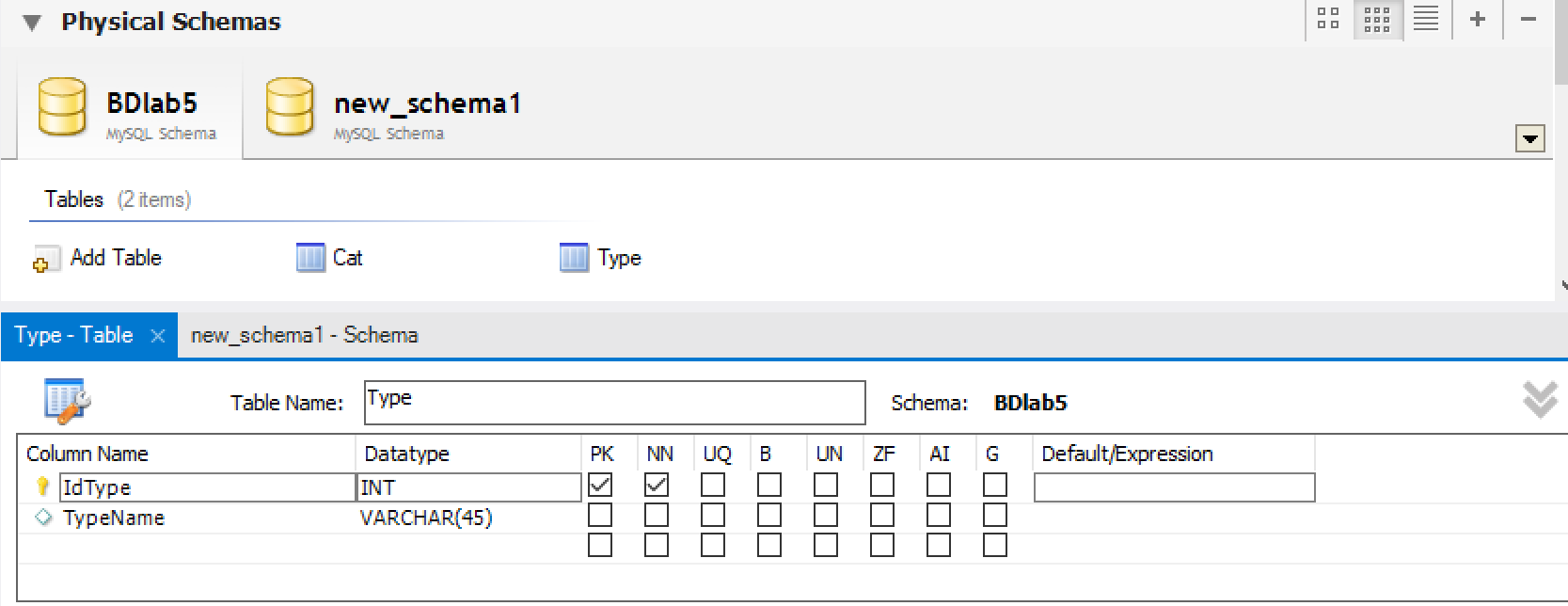
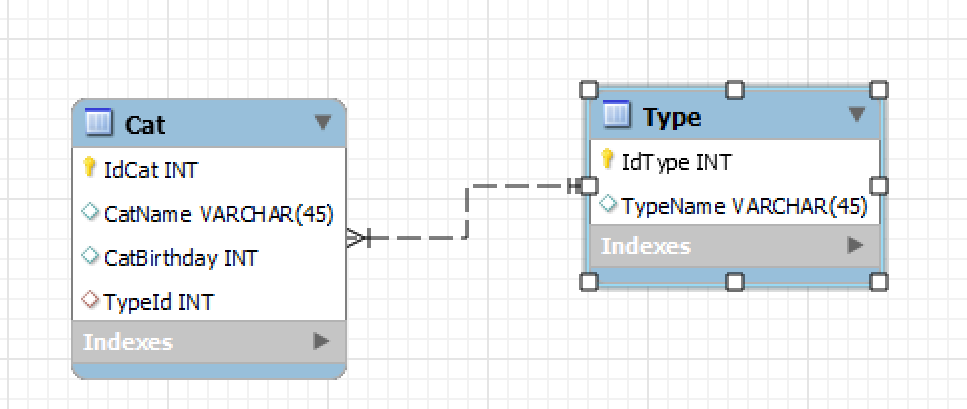


Таблица «Type»

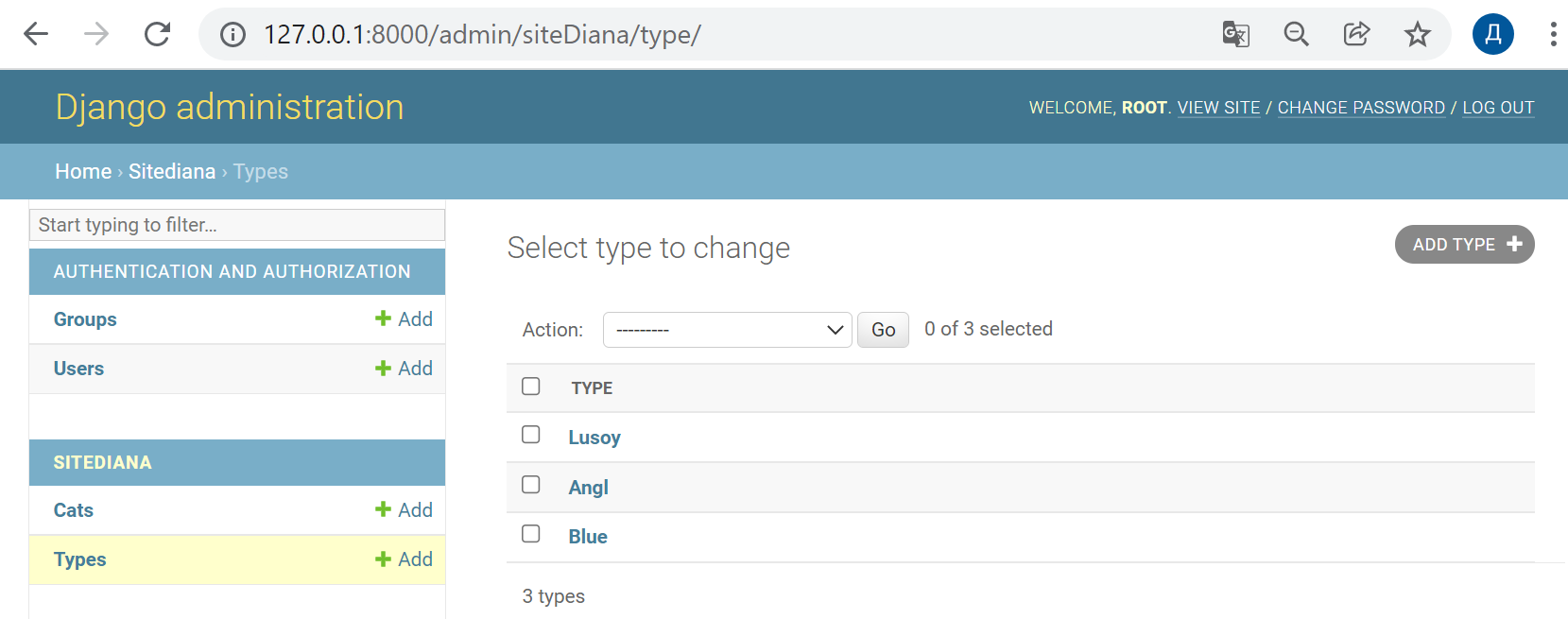
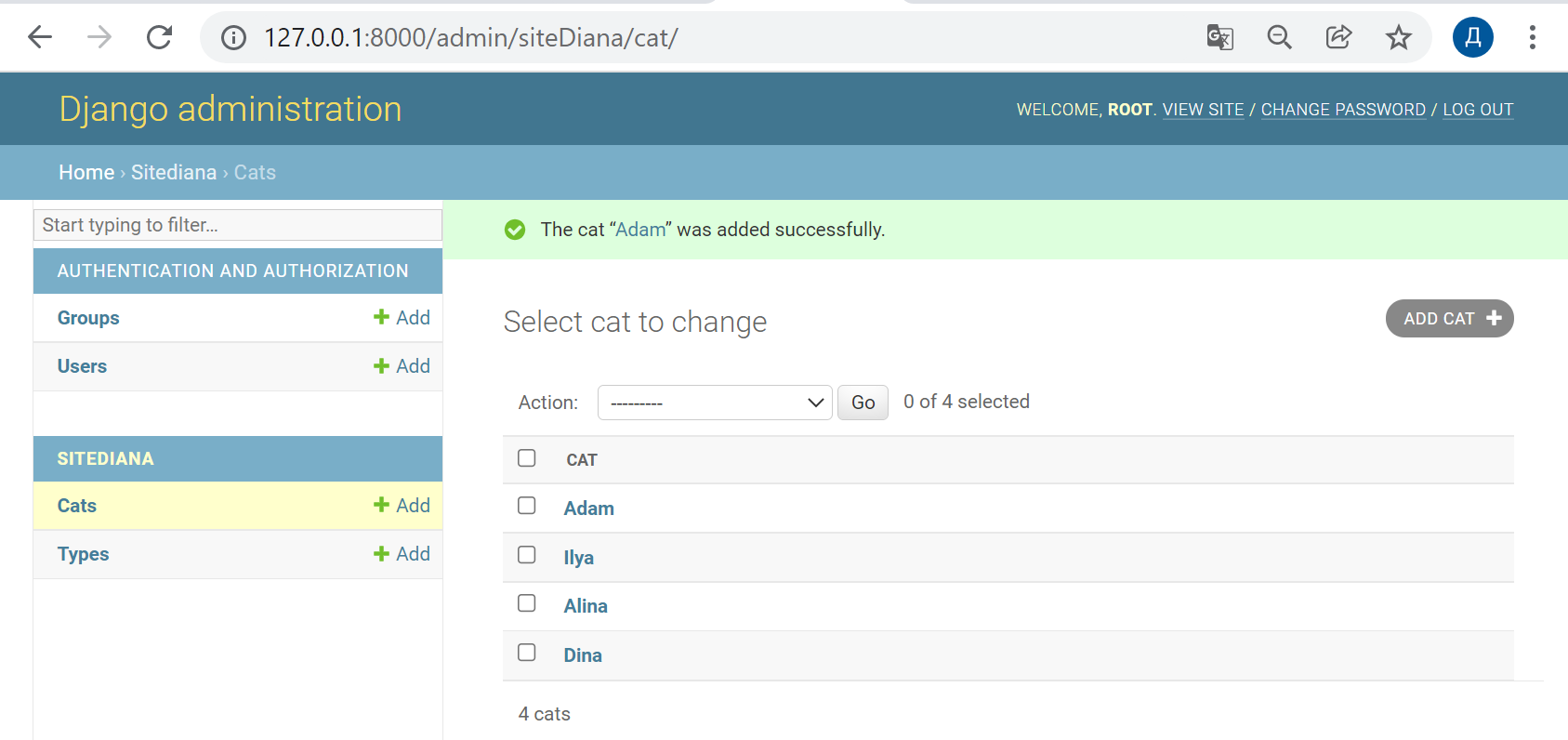


**Связь таблиц**

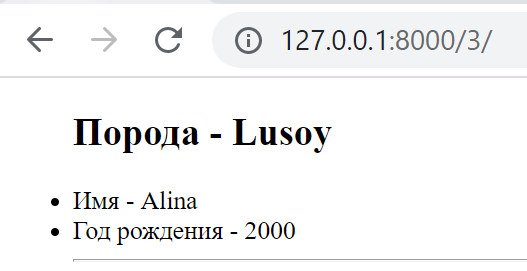
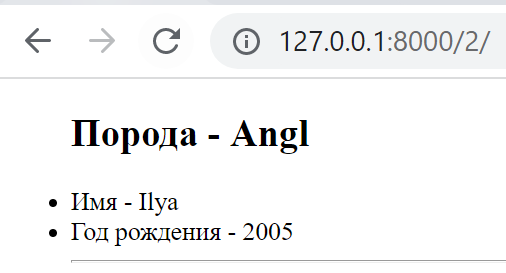
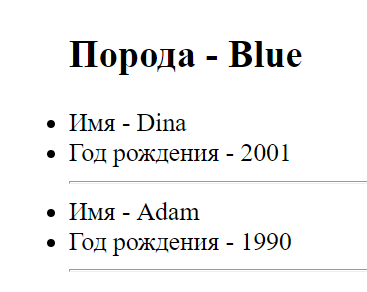
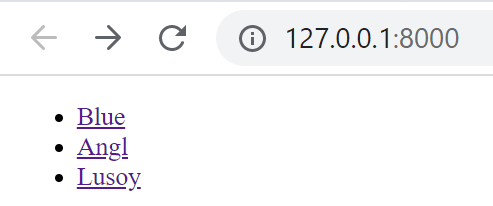


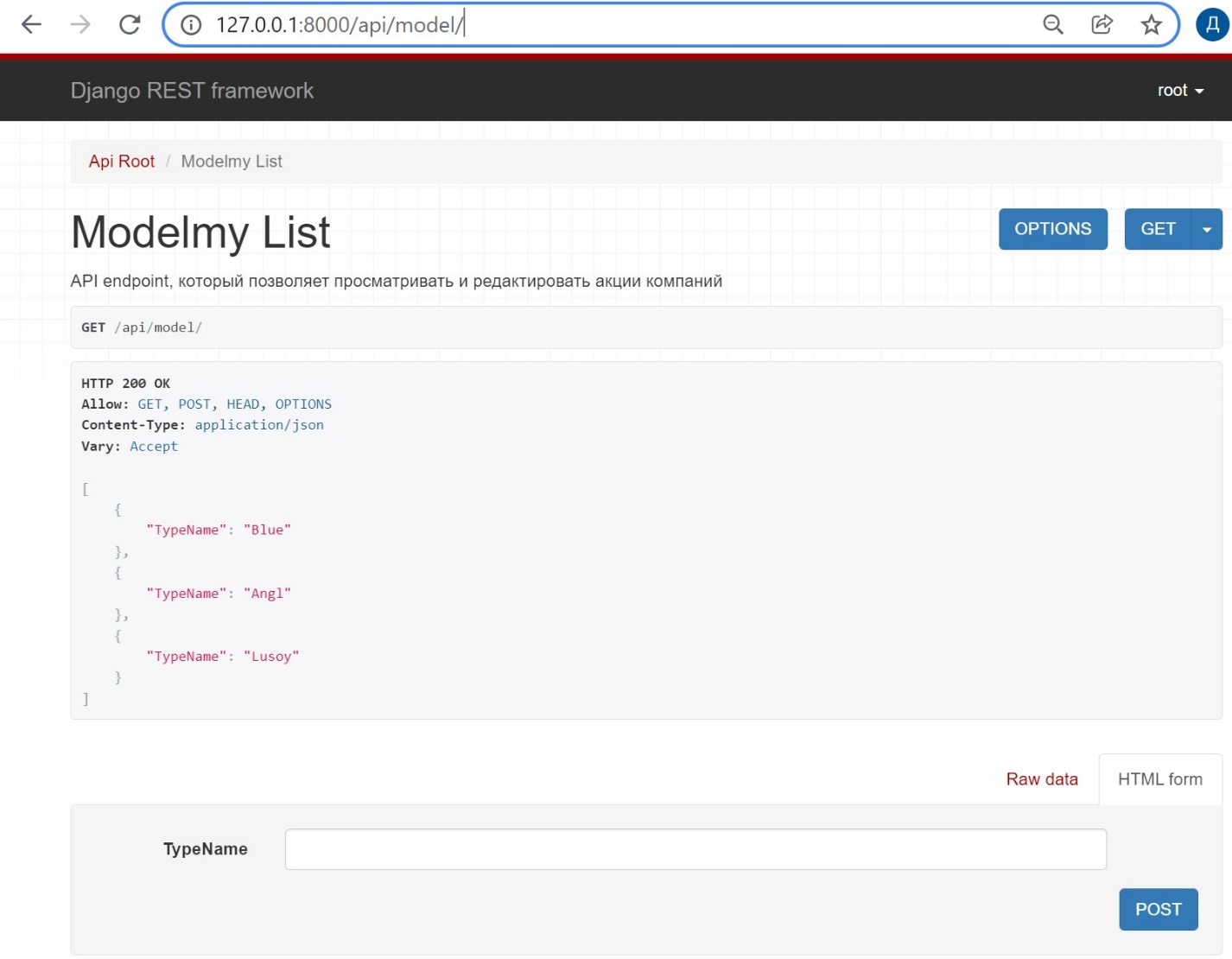
**Веб-приложение**

Страница администратора:



Страница 127.0.0.1:8000





# Вывод

В ходе выполнения данной лабораторной работы были повторены основные конструкции языка Python. Также были изучены возможности СУБД MySQL для разработки веб-приложений и обработки данных с использованием Django ORM. Были изучены возможности разработки REST API с использованием Django REST Framework.