**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 №7

«Разработка пользовательского интерфейса для работы с REST API»

|  |  |  |
| --- | --- | --- |
| 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 создания пользовательского интерфейса в веб-приложениях.

# Описание задания

# На основе методических указаний разработайте пользовательский интерфейс для работы с REST API. Используйте REST API, разработанный Вами в предыдущей лабораторной работе.

# Текст программы

**admin.py**

from django.contrib import admin  
from .models import Cat  
  
  
# Register your models here.  
admin.site.register(Cat)

**apps.py**

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

**models.py**

from django.db import models  
  
class Cat(models.Model):  
 CatName = models.CharField(max\_length=100)  
 CatBirthday = models.IntegerField()  
 #TypeId = models.ForeignKey(Type, on\_delete=models.CASCADE)  
 pic = models.CharField(max\_length=500)  
 #def \_\_str\_\_(self):  
 #return self.CatName

**serializers.py**

from .models import Cat  
from rest\_framework import serializers  
  
class CatSerializer(serializers.ModelSerializer):  
 class Meta:  
 model = Cat  
 fields = ["pk", 'CatName','CatBirthday', 'pic']

**views.py**

from django.shortcuts import render  
  
from rest\_framework import viewsets  
from .serializers import CatSerializer  
from .models import Cat  
  
class CatViewSet(viewsets.ModelViewSet):  
 queryset = Cat.objects.all()  
 serializer\_class = CatSerializer

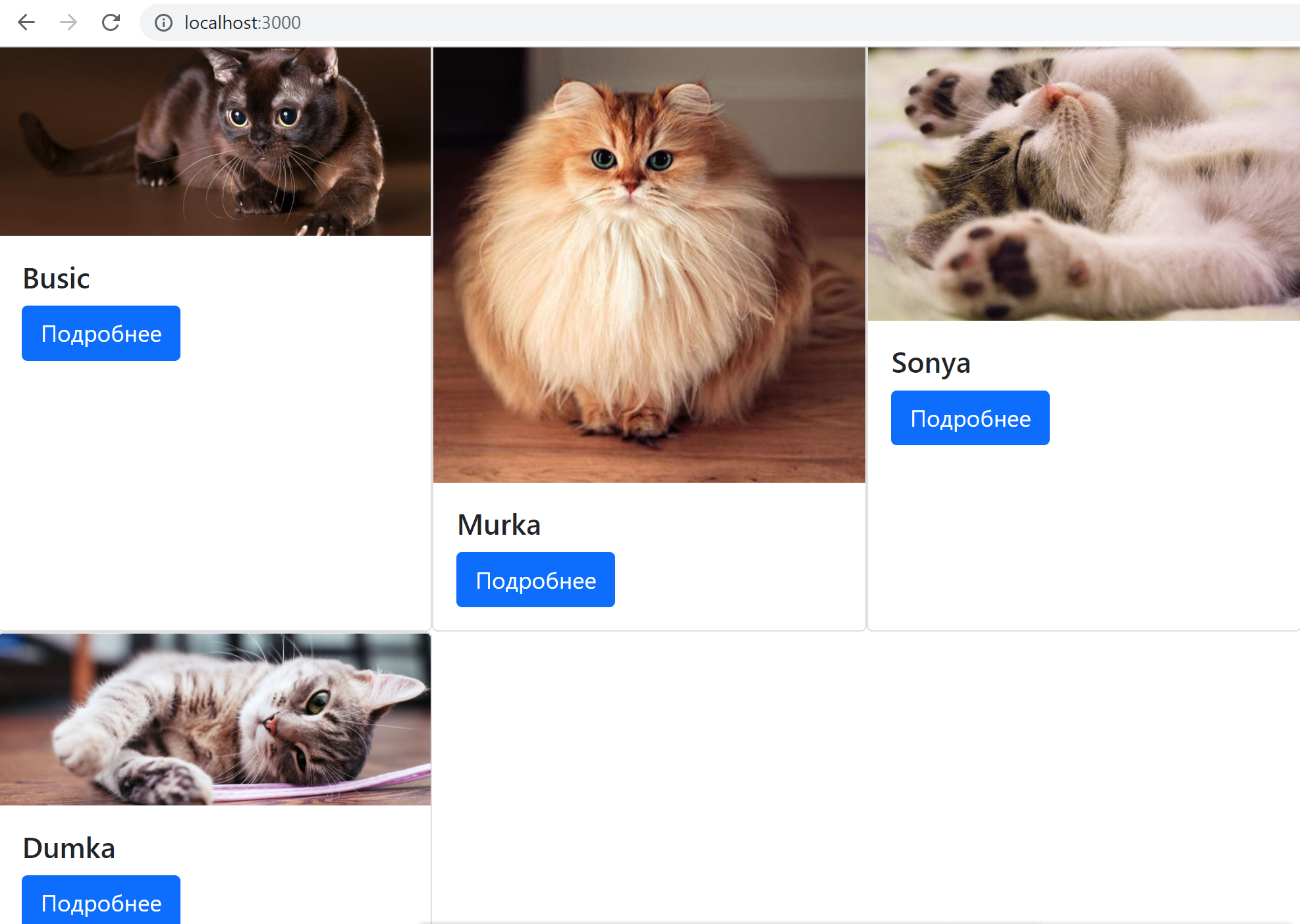
**settings.py**

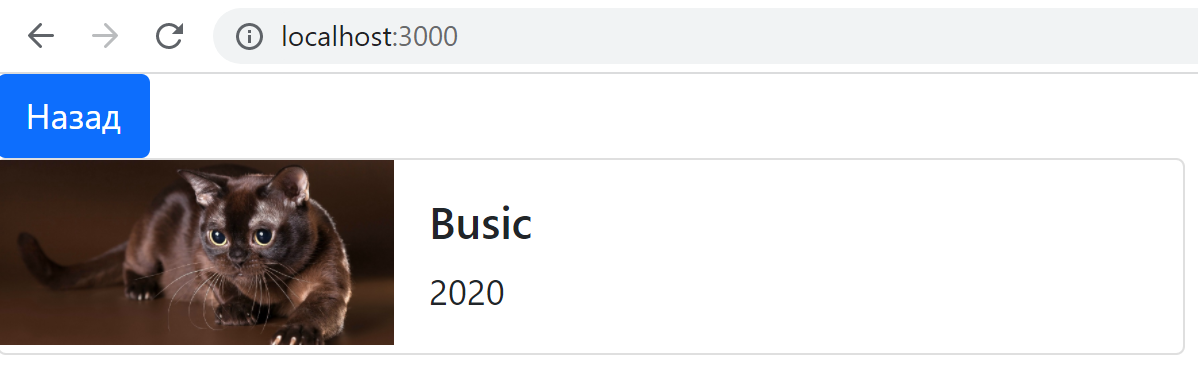
*"""  
Django settings for BookShop project.  
  
Generated by 'django-admin startproject' using Django 4.0.  
  
For more information on this file, see  
https://docs.djangoproject.com/en/4.0/topics/settings/  
  
For the full list of settings and their values, see  
https://docs.djangoproject.com/en/4.0/ref/settings/  
"""*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/4.0/howto/deployment/checklist/  
  
# SECURITY WARNING: keep the secret key used in production secret!  
SECRET\_KEY = 'django-insecure-4\*u+9-$e\_7m-d!c2^93ft-rp%n9m@f+4222h)a!p)d!xy5tzy6'  
  
# 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',  
 'bk\_app.apps.BkAppConfig',  
 'rest\_framework',  
 'corsheaders'  
]  
  
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',  
 "corsheaders.middleware.CorsMiddleware",  
 "django.middleware.common.CommonMiddleware",  
  
]  
  
ROOT\_URLCONF = Cats.urls'  
  
TEMPLATES = [  
 {  
 'BACKEND': 'django.template.backends.django.DjangoTemplates',  
 'DIRS': [],  
 '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 = Cats.wsgi.application'  
  
  
# Database  
# https://docs.djangoproject.com/en/4.0/ref/settings/#databases  
  
DATABASES = {  
 'default': {  
 'ENGINE': 'django.db.backends.sqlite3',  
 'NAME': BASE\_DIR / 'db.sqlite3',  
 }  
}  
  
  
# Password validation  
# https://docs.djangoproject.com/en/4.0/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/4.0/topics/i18n/  
  
LANGUAGE\_CODE = 'en-us'  
  
TIME\_ZONE = 'UTC'  
  
USE\_I18N = True  
  
USE\_TZ = True  
  
  
# Static files (CSS, JavaScript, Images)  
# https://docs.djangoproject.com/en/4.0/howto/static-files/  
  
STATIC\_URL = 'static/'  
  
# Default primary key field type  
# https://docs.djangoproject.com/en/4.0/ref/settings/#default-auto-field  
  
DEFAULT\_AUTO\_FIELD = 'django.db.models.BigAutoField'  
CORS\_ALLOWED\_ORIGINS = [  
 "http://localhost:3000",  
]

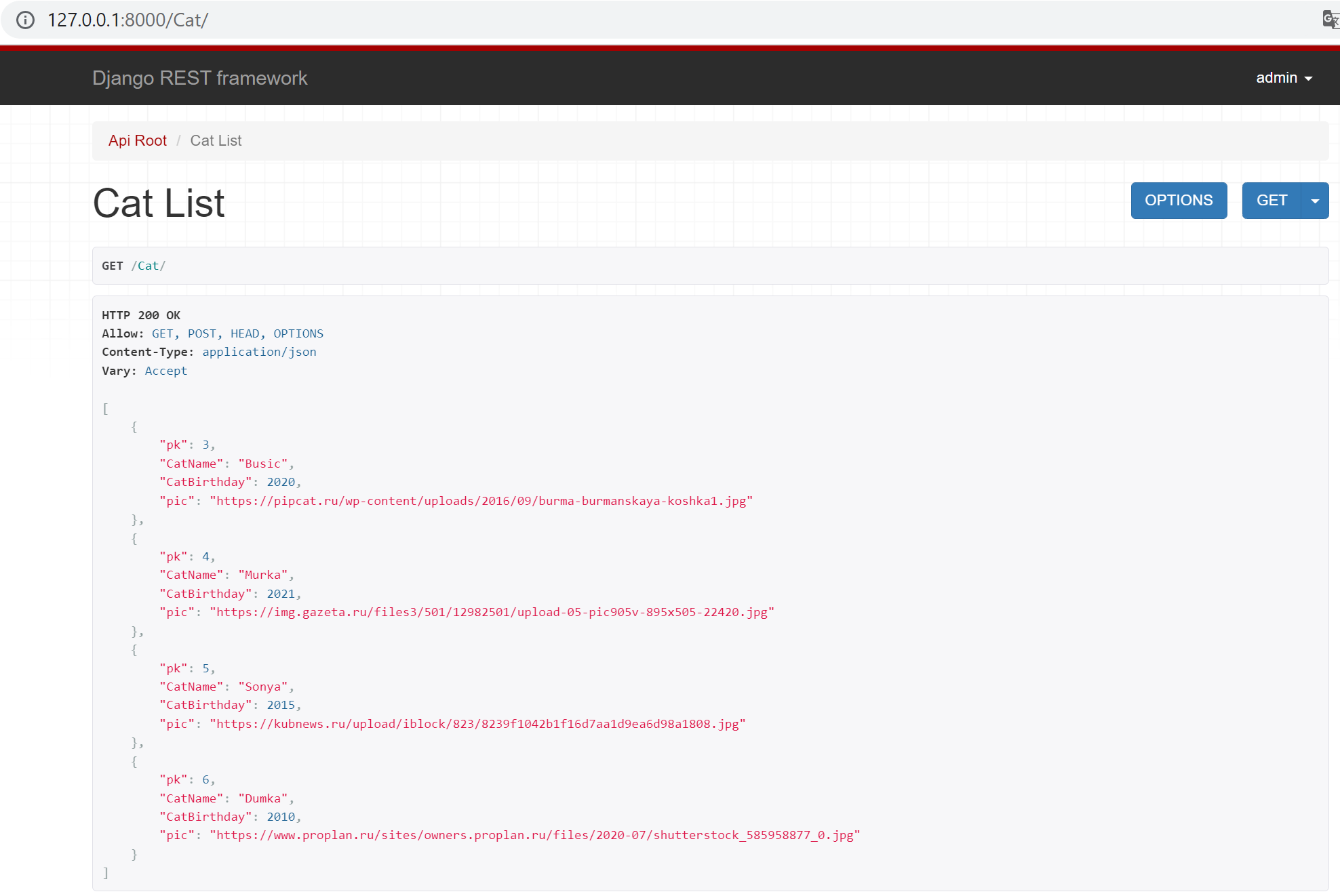
**urls.py**

from django.contrib import admin  
from django.urls import path  
from django.urls.conf import include  
from rest\_framework import routers  
from bk\_app import views as Cat\_views  
  
router = routers.DefaultRouter()  
router.register(r'Cat', Cat\_views.CatViewSet)  
  
urlpatterns = [  
 path('admin/', admin.site.urls),  
 path('', include(router.urls)),  
 path('api-auth/', include('rest\_framework.urls', namespace='rest\_framework')),  
]

# Экранные формы с примерами выполнения программы







# Вывод

В ходе выполнения данной лабораторной работы были повторены основные конструкции языка Python. Также были изучены возможности создания пользовательского интерфейса в веб-приложениях.