Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ РАДИОТЕХНИЧЕСКИЙ

КАФЕДРА СИСТЕМЫ ОБРАБОТКИ ИНФОРМАЦИИ И УПРАВЛЕНИЯ

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №6

Студент Столярова Ольга Денисовна
фамилия, имя, отчество

Группа РТ5-51Б

Название предприятия МГТУ им. Н. Э. Баумана

Студент Столярова О.Д.

Преподаватель Гапанюк Ю.Е.

2021 г.

Цель работы

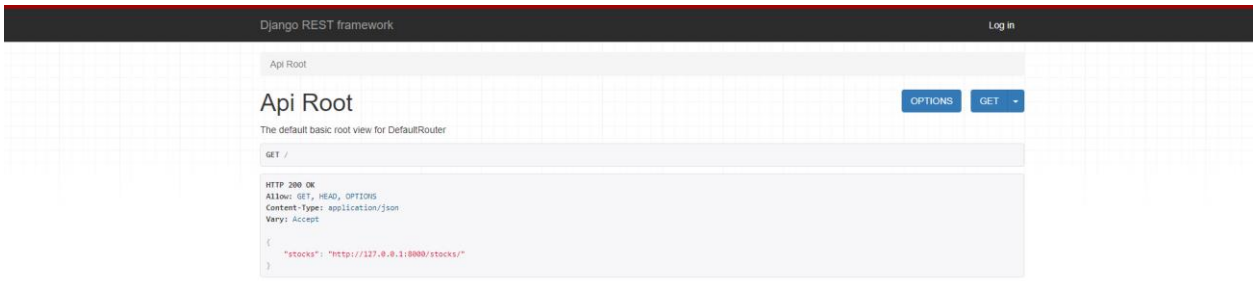
Цель лабораторной работы: изучение возможностей разработки REST API с использованием Django REST Framework.

Задание

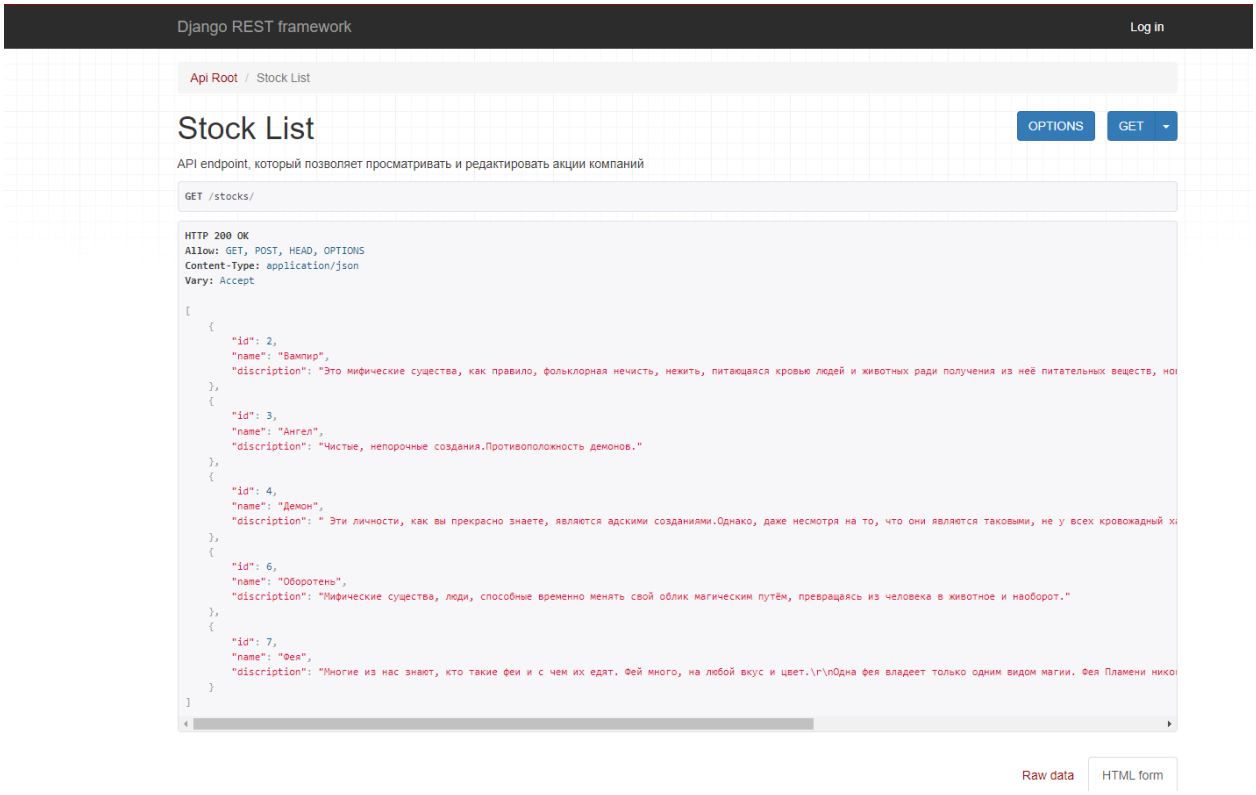
С использованием Django REST Framework разработайте REST API для одной модели (одной таблицы базы данных).

Пример выполнения

http://127.0.0.1:8000/



http://127.0.0.1:8000/stocks/



Код программы

Lr6/urls.py

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

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', include('main.urls')),
]
```

lr6/settings.py

```
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-
p(z8&hr)=ax2xxz4@##*%&3c3=46)v0_sunb#*v=!qp2_=tx5t'

# 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',
    'main'
]

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 = 'lr5.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 = 'lr5.wsgi.application'


# Database
# https://docs.djangoproject.com/en/4.0/ref/settings/#databases

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': 'creat',
        'USER': 'root',
        'PASSWORD': '20012011',
        'HOST': 'localhost',
        'PORT': '3306',
    }
}
# 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'
```

main/models.py

```
from django.db import models


class Rases(models.Model):
    name = models.CharField(max_length=30)
    discription = models.CharField(max_length=255)

    class Meta:
        managed = False
        db_table = 'rases'


class Cans(models.Model):
    idrases = models.IntegerField()
    cans = models.CharField(max_length=100)

    class Meta:
        managed = False
        db_table = 'cans'
```

main/urls.py

```
from django.urls import path, include
from . import views
from django.conf.urls.static import static
from django.conf import settings
from . import views as stock_views
from rest_framework import routers


router = routers.DefaultRouter()
router.register(r'stocks', stock_views.StockViewSet)


urlpatterns = [
    path('', include(router.urls)),
    path('api-auth/', include('rest_framework.urls',
namespace='rest_framework')),
    path('', views.master, name='master'),
    path('<int:cr_id>/', views.detail, name='detail'),
    path('<int:cr_id>/cans/', views.can, name='cann')

]
```

main/serializers.py

```
from .models import Rases
from rest_framework import serializers


class StockSerializer(serializers.ModelSerializer):
    class Meta:
        # Модель, которую мы сериализуем
        model = Rases
        # Поля, которые мы сериализуем
        fields = ["id","name", "discription"]
```

Ссылка на GitHub
https://github.com/OlyaSto/Olyabmstu