**Инструкция по созданию API в Django + deploy на heroku**

**Работа на локальном сервере**

Создаем виртуальное окружение:

virtualenv env

Активируем виртуальное окружение:

env/Scripts/activate

Устанавливаем django и djangorestframework:

pip install django, djangorestframework

Настраиваем линтер в VS Code:

pip install pylint-django

ctrl+shift+p > Preferences: Configure Language Specific Settings > Python

settings.json:

{

"python.linting.pylintArgs": [

"--load-plugins=pylint\_django"

],

…..

"[python]": {

}

}

Создаем новый django проект:

django-admin startproject djreact

Переименовываем папку djreact (снаружи djreact) в src и переходим в неё

Выполняем миграции:

python manage.py migrate

Запускаем сервер:

python manage.py runserver

Проверяем, что по адресу http://127.0.0.1:8000/ запускается стартовая заглушка django

Создаем новое приложение:

python manage.py startapp articles

**Django RESTful API**

В src/djreact/settings.py в INSTALLED\_APPS добавляем два приложения:

'rest\_framework',

'articles'

В src/djreact/urls.py:

from django.contrib import admin

from django.urls import path, include

urlpatterns = [

    path('api-auth/', include('rest\_framework.urls'))

    path('admin/', admin.site.urls),

]

В конец src/djreact/settings.py:

REST\_FRAMEWORK = {

    # Use Django's standard `django.contrib.auth` permissions,

    # or allow read-only access for unauthenticated users.

    'DEFAULT\_PERMISSION\_CLASSES': [

        'rest\_framework.permissions.DjangoModelPermissionsOrAnonReadOnly'

    ]

}

**Создание модели**

файл src/articles/models.py:

from django.db import models

class Article(models.Model):

    title = models.CharField(max\_length=120)

    content = models.TextField()

    def \_\_str\_\_(self):

        return self.title

Создаем и выполняем миграции:

python manage.py makemigrations

python manage.py migrate

**Настраиваем админку**

Файл src/articles/admin.py:

from django.contrib import admin

from .models import Article

admin.site.register(Article)

Создаем суперюзера:

python manage.py createsuperuser

Проверяем, что админка работает:

<http://127.0.0.1:8000/admin/>

Создаем в админке пару статей

**Создаем сериалайзер**

Сериалайзер нужен для конвертации между JSON-данными и данными в модели Django

Создаем каталог src/articles/api

В нем создаем файлы \_\_init\_\_.py и serializers.py

В файле serializers.py:

from rest\_framework import serializers

from articles.models import Article

class ArticleSerializer(serializers.ModelSerializer):

    class Meta:

        model = Article

        fields = ('title', 'content')

**Создаем представление**

Создаем файл src/articles/api/views.py:

from rest\_framework.generics import ListAPIView, RetrieveAPIView

from articles.models import Article

from .serializers import ArticleSerializer

class ArticleListView(ListAPIView):

    queryset = Article.objects.all()

    serializer\_class = ArticleSerializer

class ArticleDetailView(RetrieveAPIView):

    queryset = Article.objects.all()

    serializer\_class = ArticleSerializer

Этот вариант максимально абстрактный (использование дженериков). Можно и по-другому сделать.

**Создаем маршрутизацию**

Создаем файл src/articles/api/urls.py:

from django.urls import path

from .views import ArticleListView, ArticleDetailView

urlpatterns = (

    path('', ArticleListView.as\_view()),

    path('<pk>', ArticleDetailView.as\_view()),

)

Подключаем этот файл urls в базовый файл src/djreact/urls.py:

from django.contrib import admin

from django.urls import path, include

urlpatterns = [

    path('api-auth/', include('rest\_framework.urls')),

    path('admin/', admin.site.urls),

**path('api/', include('articles.api.urls'))**

]

Тестируем:

<http://127.0.0.1:8000/api/>

<http://127.0.0.1:8000/api/1>

Улучшение – если создаем действительно RESTful API (CRUD):

articles/api/views.py:

from rest\_framework import viewsets

from articles.models import Article

from .serializers import ArticleSerializer

class ArticleViewSet(viewsets.ModelViewSet):

    serializer\_class = ArticleSerializer

    queryset = Article.objects.all()

articles/api/urls.py:

from articles.api.views import ArticleViewSet

from rest\_framework.routers import DefaultRouter

router = DefaultRouter()

router.register(r'', ArticleViewSet, basename='articles')

urlpatterns = router.urls

**Деплой на heroku**

Настраиваем settings.py:

DEBUG = False

import os

SECRET\_KEY = os.environ.get('DJANGO\_SECRET\_KEY', 'ключ по умолчанию (который был раньше)')

Устанавливаем библиотеки:

pip install gunicorn

pip install django-heroku

Замораживаем библиотеки:

pip freeze > requirements.txt

Все из папки src переносим в корень

Создаем файл Procfile:

web: gunicorn myproject.wsgi

myproject – это имя проекта

В начало settings.py добавляем:

import django\_heroku

В конец settings.py добавляем:

django\_heroku.settings(locals())

Регистрируемся на сайте Heroku

Устанавливаем Heroku CLI

Входим:

heroku login

Создаем приложение:

heroku create

Выполняем деплой:

git push heroku master

Выполняем команду:

heroku ps:scale web=1

Задаем секретный ключ:

heroku config:set DJANGO\_SECRET\_KEY=’секретный ключ’

Проверяем, что деплой произошел:

heroku open

Создаем суперюзера на heroku:

heroku run python manage.py createsuperuser

Выполняем миграции на heroku:

heroku run python manage.py migrate

Чтобы зайти в консоль на стороне Heroku:

heroku run bash -a название\_приложения

Аутентификация:

с помощью токенов

Библиотека djoser: <https://djoser.readthedocs.io/>

Login:

POST <http://127.0.0.1:8000/auth/token/login/>

{

"username": "…",

"password": "…"

}

Logout:

POST <http://127.0.0.1:8000/auth/token/logout>

Во всех обращениях к API добавляем заголовок:

key = Authorization

Value = Token 103a00c0ffffb4b893dd7aa7f50ed9ece3e6987c