МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ   
  
ОРДЕНА ТРУДОВОГО КРАСНОГО ЗНАМЕНИ  
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ  
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«МОСКОВСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ СВЯЗИ И ИНФОРМАТИКИ»

**Кафедра Математической кибернетики и информационных технологий**

**Лабораторная работа №2**

по дисциплине: Введение в информационные технологии

Выполнил:  
Студент группы БВТ 2306  
Шошкин Илья

Дмитриевич  
  
Проверил:

Москва, 2023

ОГЛАВЛЕНИЕ

* 1.ЦЕЛЬ РАБОТЫ……………………………………………………3
* 2.ПОСТАНОВКА ЗАДАЧИ………………………………………3
* 3.ВЫПОЛНЕНИЕ РАБОТЫ…………………………………….4
* 4.ВЫВОДЫ……………………………………………………7
* 1.ЦЕЛЬ РАБОТЫ

Целью работы является изучение FastAPI, pyjokes, pydantic, wikipedia-api

* 2.ПОСТАНОВКА ЗАДАЧИ

⦁ Создать файл main.py

⦁ Установить FastAPI

⦁ Установить Pyjokes

⦁ Установить Pydantic

⦁ Установить Wikipedia-api

⦁ Выполнить домашнее задание представленное в лабораторной работе №2

* 3. ВЫПОЛНЕНИЕ РАБОТЫ

1.Устанавливаем зависимости через терминал внутри проекта

pip install fastapi

pip intsall Pyjokes

pip intsall Pydantic

pip intsall Wikipediaapi

2.Пишем код:

#Импортируем необходимые модули и создадим инстанс FastAPI

from fastapi import FastAPI, Body, HTTPException

from pydantic import BaseModel

import pyjokes

import wikipediaapi

# Установим язык для wikipedia на РУ

wiki\_wiki = wikipediaapi.Wikipedia('127.0.0.1', 'ru')

#Определим Pydantic модели для запросов и ответов:

class WikiQueryParams(BaseModel):

query: str

class WikiBodyParams(BaseModel):

query: str

class WikiResponse(BaseModel):

title: str

pageid: int

summary: str

#Создадим роуты в соответствии с заданием:

app = FastAPI()

#GET /path/{title} получает данные страницы Wikipedia по заголовку.

@app.get("/path/{title}", response\_model=WikiResponse)

def get\_wiki\_by\_path(title: str):

page = wiki\_wiki.page(title)

if not page.exists():

raise HTTPException(status\_code=404, detail="Page not found")

return {

"title": page.title,

"pageid": page.pageid,

"summary": page.summary

}

#GET /query/ использует query параметр для извлечения данных страницы Wikipedia.

@app.get("/query/", response\_model=WikiResponse)

def get\_wiki\_by\_query(query: str):

page = wiki\_wiki.page(query)

if not page.exists():

raise HTTPException(status\_code=404, detail="Content not found")

return {

"title": page.title,

"pageid": page.pageid,

"summary": page.summary

}

#POST /body/ получает данные страницы Wikipedia через тело запроса в формате JSON.

@app.post("/body/", response\_model=WikiResponse)

def get\_wiki\_by\_body(params: WikiBodyParams = Body(...)):

page = wiki\_wiki.page(params.query)

if not page.exists():

raise HTTPException(status\_code=404, detail="Content not found")

return {

"title": page.title,

"pageid": page.pageid,

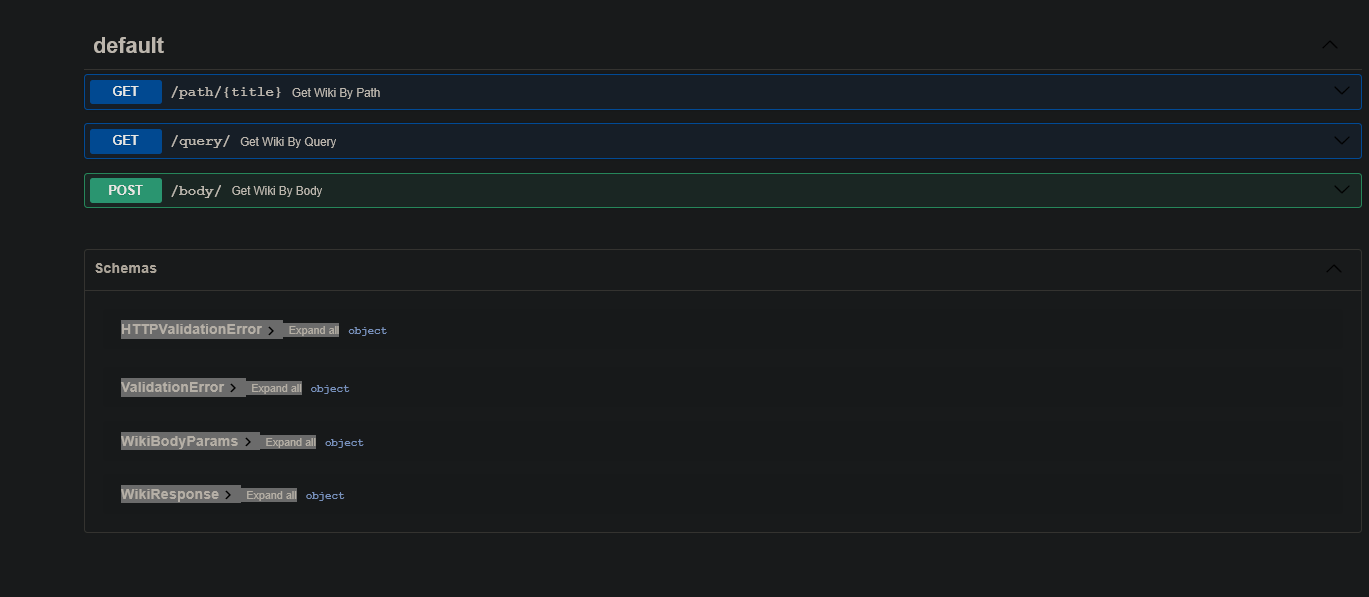
"summary": page.summary

}

3.Запускаем веб сервер следующей командой в терминале: uvicorn main:app --reload

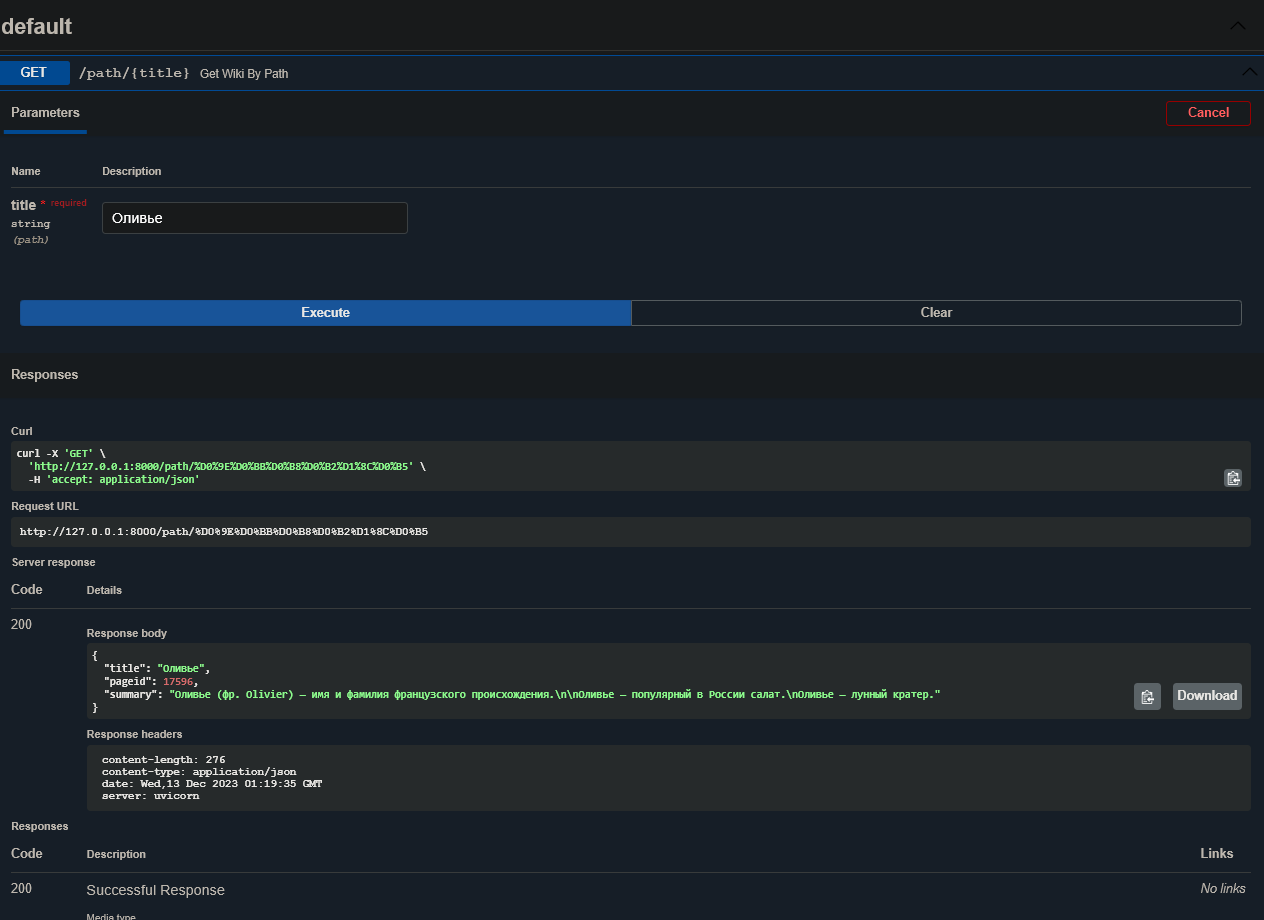
4.Переходим на Swagger UI: <http://127.0.0.1:8000/docs>

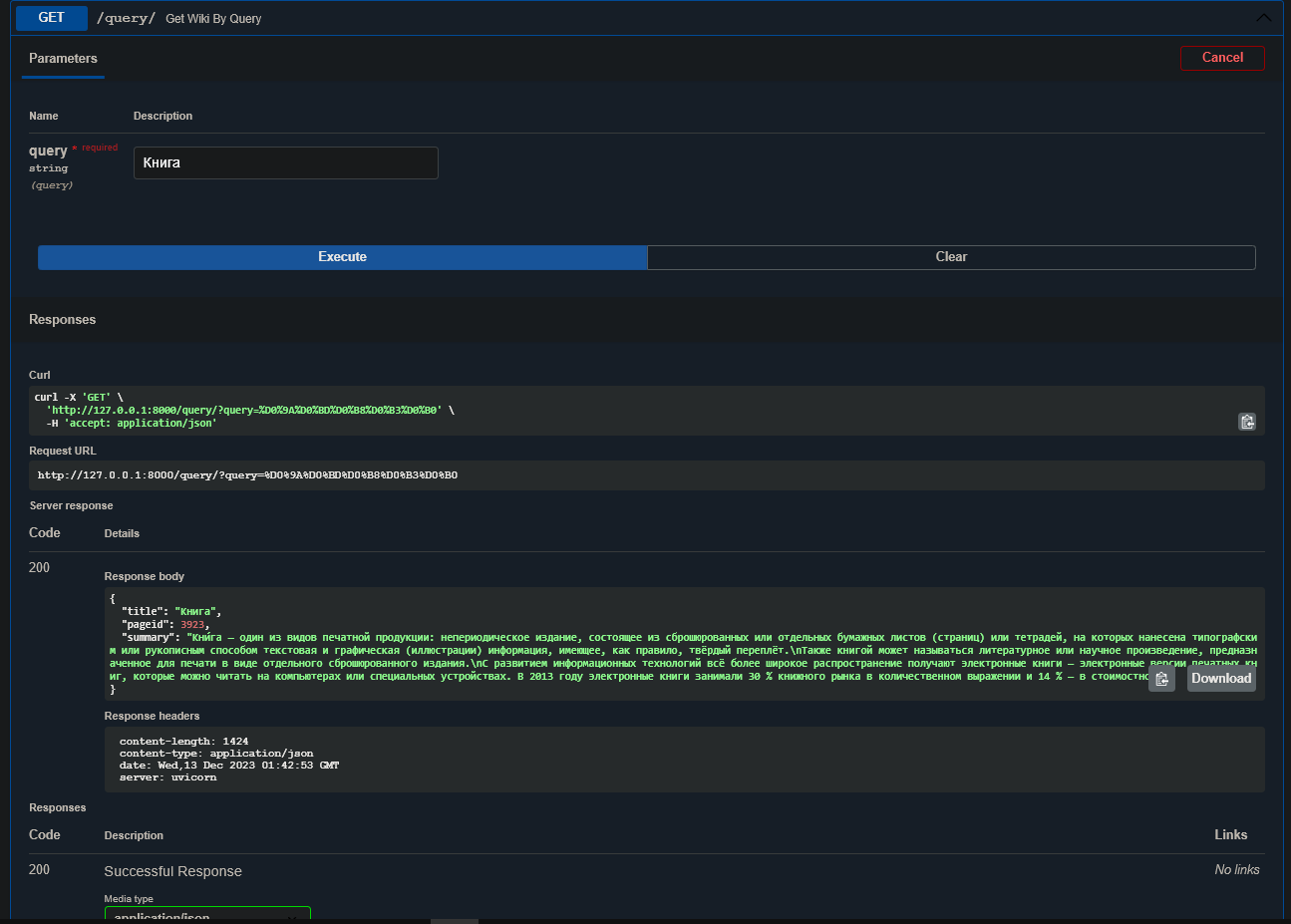
5.Проверяем работу:



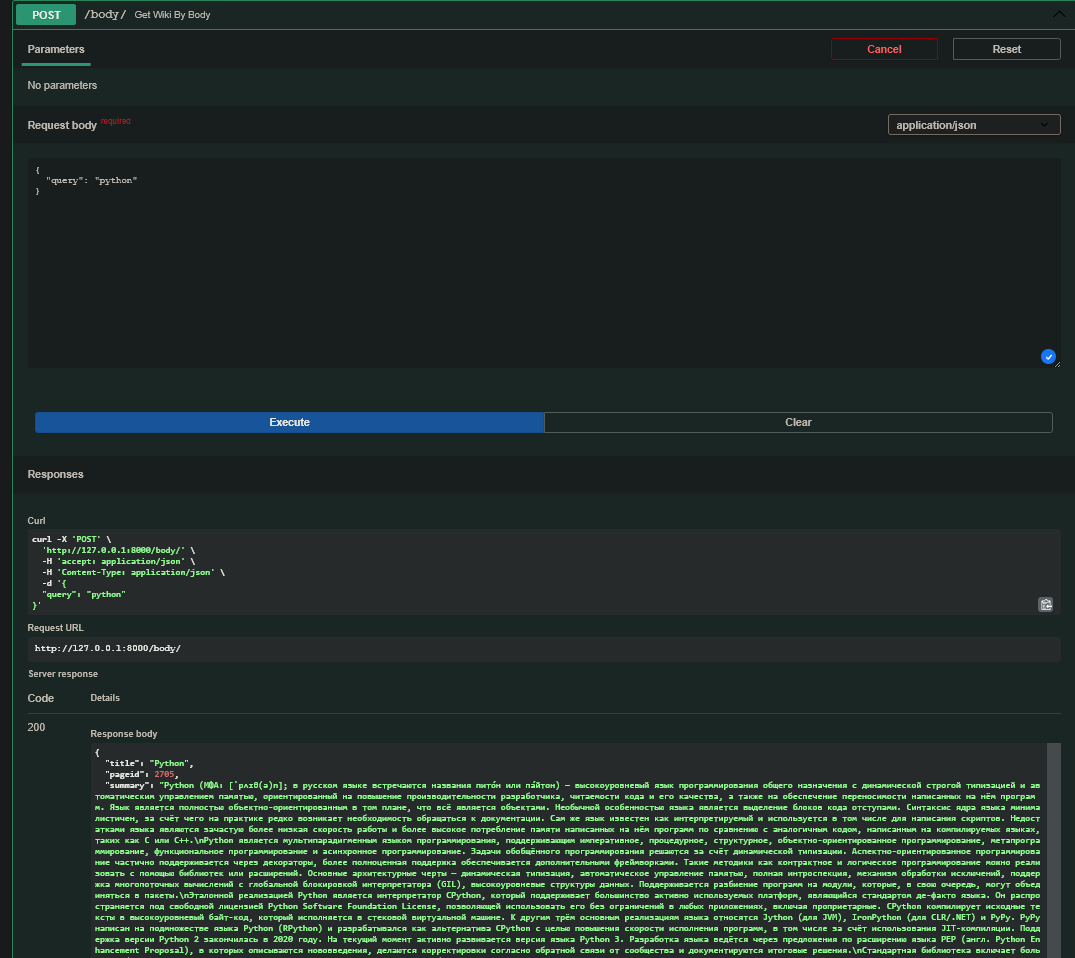
Результаты запросов:

GET Запрос по /path/{title}



GET Запрос по /query/  


POST Запрос по /body/



4.ВЫВОДЫ

В ходе данной лабораторной работы было выполнено ознакомление с работой FastAPI, pyjokes, pydantic, wikipedia-api. Мы научились, запускать веб-сервер, использовать Swagger UI, создавать POST и GET запросы с помощью этих инструментов.