|  |  |  |
| --- | --- | --- |
|  |  |  |
| МИНОБРНАУКИ РОССИИ | | |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | | |

Институт Информационных технологий

Кафедра Инструментального и прикладного программного обеспечения

**ПРАКТИЧЕСКАЯ РАБОТА №5**

**Тема: “Создание и документирование API на основе спецификации OpenAPI”**

по дисциплине «Интерфейсы прикладного программирования»

**Студент группы**

Постнов И.Н.

(подпись студента)

Принял Зарипов Евгений Анатольевич

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись руководителя)

Работа представлена «\_\_» \_\_\_\_\_\_\_\_\_2022 г.

Допущен к работе «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2022 г.

**ОГЛАВЛЕНИЕ**

[Ход работы](#_gjdgxs) [3](#_gjdgxs)

[Выводы](#_3znysh7) [7](#_3znysh7)

[Ответы на вопросы к практической работе](#_2et92p0) [8](#_2et92p0)

[Список использованной литературы:](#_tyjcwt) 9

# 

# Цель работы

Целью данной практической работы является знакомство студентов с основными подходами к построению API и инструментарием, используемый для ускорения процесса разработки API, в части документирования интерфейса прикладного программирования.

# Ход работы

Создадим файл books.yaml (Листинг. 1)

Листинг кода 1. Файл books.yaml

openapi: 3.0.0

info:

version: 1.0.0

title: library

license:

name: MIT

servers:

- url: http://api.openapi.io/v1

paths:

/authors:

get:

operationId: find all authors

parameters:

- name: tags

in: query

description: tags to filter by

required: false

style: form

schema:

type: array

items:

type: string

- name: limit

in: query

description: maximum number of results to return

required: false

schema:

type: integer

format: int32

responses:

'200':

description: author response

content:

application/json:

schema:

type: array

items:

$ref: '#/components/schemas/Author'

default:

description: unexpected error

content:

Продолжение листинга 1.

application/json:

schema:

$ref: '#/components/schemas/Error'

post:

description: Add a new author in the lib. Duplicates are allowed

operationId: add author

requestBody:

description: author to add to the lib

required: true

content:

application/json:

schema:

$ref: '#/components/schemas/NewAuthor'

responses:

'200':

description: author response

content:

application/json:

schema:

$ref: '#/components/schemas/Author'

default:

description: unexpected error

content:

application/json:

schema:

$ref: '#/components/schemas/Error'

/authors/{id}:

get:

description: Returns a author by id

operationId: find author by id

parameters:

- name: id

in: path

description: ID of author to fetch

required: true

schema:

type: integer

format: int64

responses:

'200':

description: author response

content:

application/json:

schema:

$ref: '#/components/schemas/Author'

default:

description: unexpected error

content:

application/json:

schema:

$ref: '#/components/schemas/Error'

delete:

description: deletes a single author based on the ID supplied

operationId: delete author

parameters:

- name: id

in: path

description: ID of author to delete

required: true

Продолжение листинга 1

schema:

type: integer

format: int64

responses:

'204':

description: author deleted

default:

description: unexpected error

content:

application/json:

schema:

$ref: '#/components/schemas/Error'

/books:

get:

operationId: find all books

parameters:

- name: tags

in: query

description: tags to filter by

required: false

style: form

schema:

type: array

items:

type: string

- name: limit

in: query

description: maximum number of results to return

required: false

schema:

type: integer

format: int32

responses:

'200':

description: books response

content:

application/json:

schema:

type: array

items:

$ref: '#/components/schemas/Book'

default:

description: unexpected error

content:

application/json:

schema:

$ref: '#/components/schemas/Error'

post:

description: Add a new book in the store. Duplicates are allowed

operationId: add book

requestBody:

description: Book to add to the store

required: true

content:

application/json:

schema:

$ref: '#/components/schemas/NewBook'

responses:

'200':

description: book response

content:

application/json:

Продолжение листинга 1

schema:

$ref: '#/components/schemas/Book'

default:

description: unexpected error

content:

application/json:

schema:

$ref: '#/components/schemas/Error'

/book/{id}:

get:

description: Returns a book by id

operationId: find book by id

parameters:

- name: id

in: path

description: ID of book to fetch

required: true

schema:

type: integer

format: int64

responses:

'200':

description: book response

content:

application/json:

schema:

$ref: '#/components/schemas/Book'

default:

description: unexpected error

content:

application/json:

schema:

$ref: '#/components/schemas/Error'

delete:

description: deletes a single book based on the ID supplied

operationId: delete book

parameters:

- name: id

in: path

description: ID of book to delete

required: true

schema:

type: integer

format: int64

responses:

'204':

description: book deleted

default:

description: unexpected error

content:

application/json:

schema:

$ref: '#/components/schemas/Error'

components:

schemas:

Author:

allOf:

- $ref: '#/components/schemas/NewAuthor'

- type: object

required:

- id

properties:

Продолжение листинга 1

id:

type: integer

format: int64

NewAuthor:

type: object

required:

- name

properties:

name:

type: string

surname:

type: string

dateOfBirth:

type: integer

dateOfDeath:

type: integer

Book:

allOf:

- $ref: '#/components/schemas/NewBook'

- type: object

required:

- id

properties:

id:

type: integer

format: int64

NewBook:

type: object

required:

- name

properties:

name:

type: string

authorId:

type: integer

Error:

type: object

required:

- code

- message

properties:

code:

type: integer

format: int32

message:

type: string

Создадим веб версию документации по API (рис. 1)

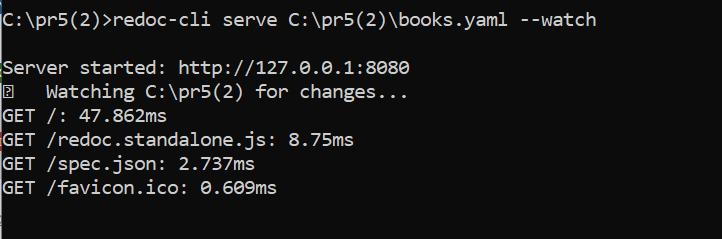


Рисунок 1. Создание веб версии документации

Запустим сервер со спецификацией (рис. 2)

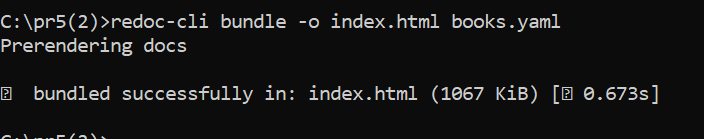


Рисунок 2. Создание веб версии документации

Сервис документации расположенный по адресу http://127.0.0.1:8080 (рис. 3)

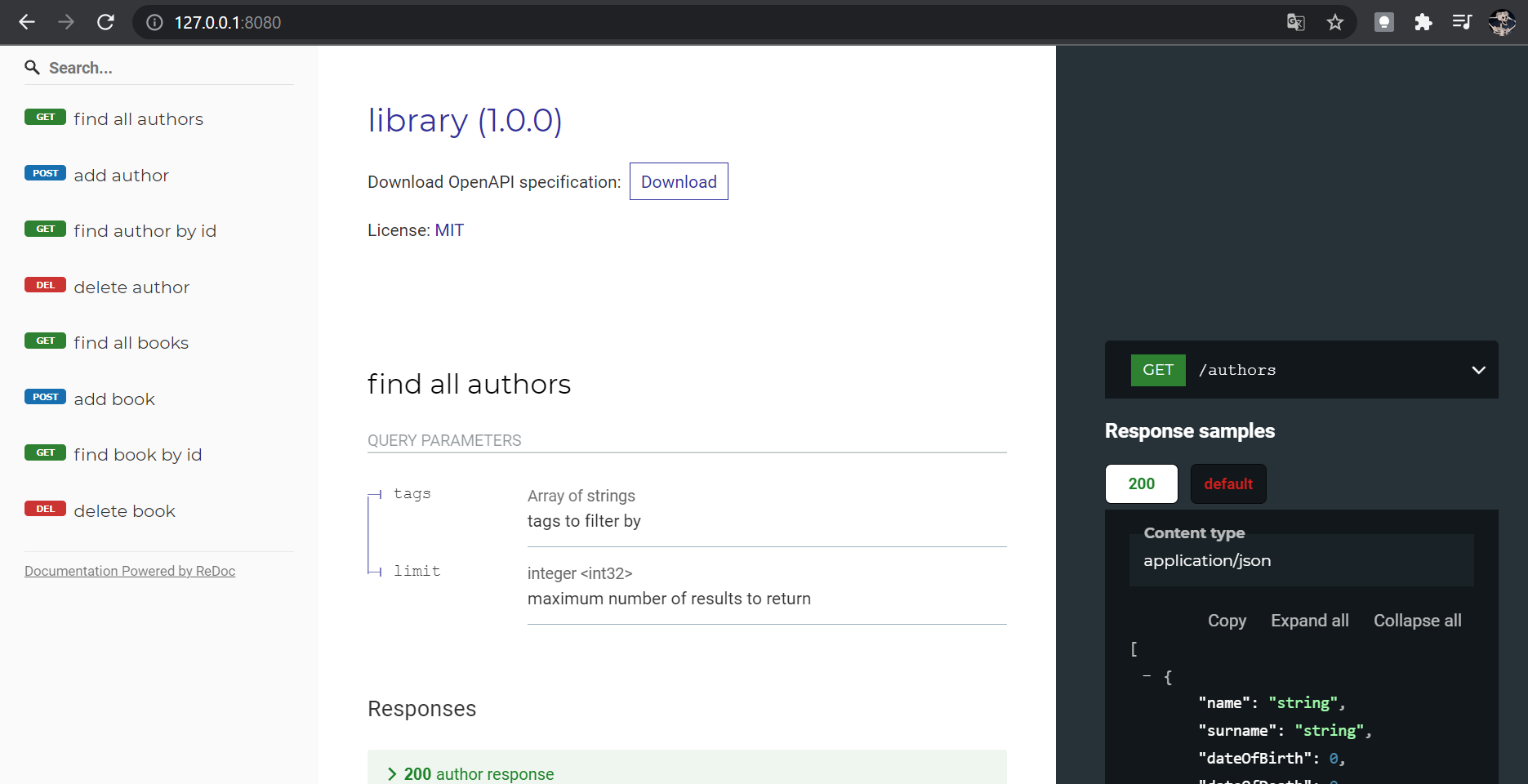


Рисунок 3.Сервис документации

Проверим правильность работы. Отправим соответствующий запрос в Postman и его с ReDoc в правой части html-страницы. (рис. 4, рис 5)

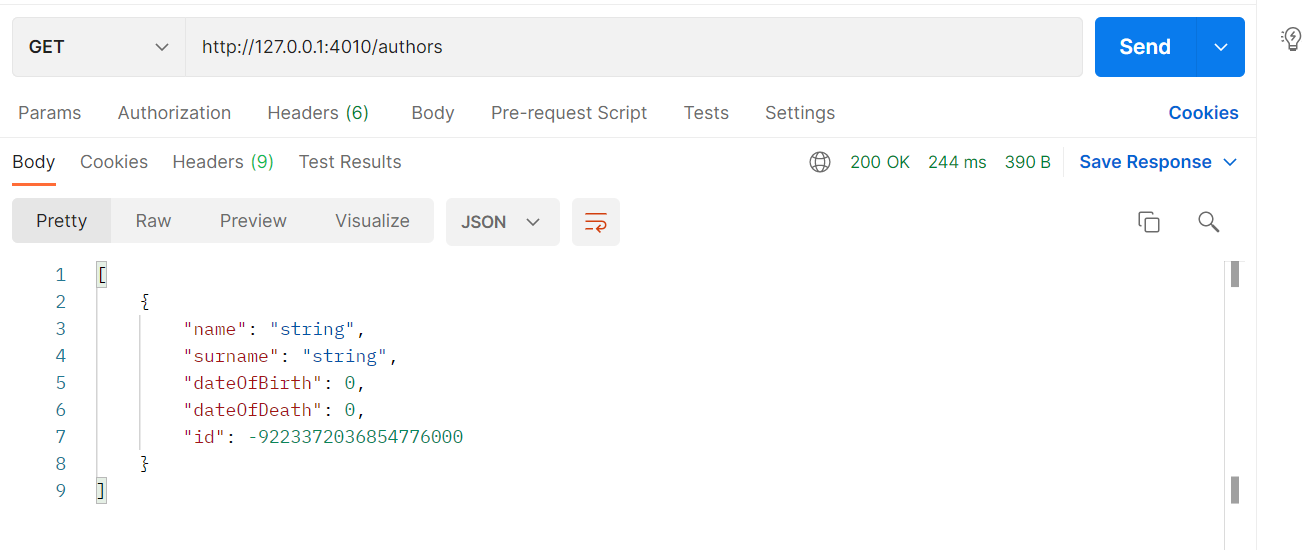


Рисунок 4. Запрос в Postman

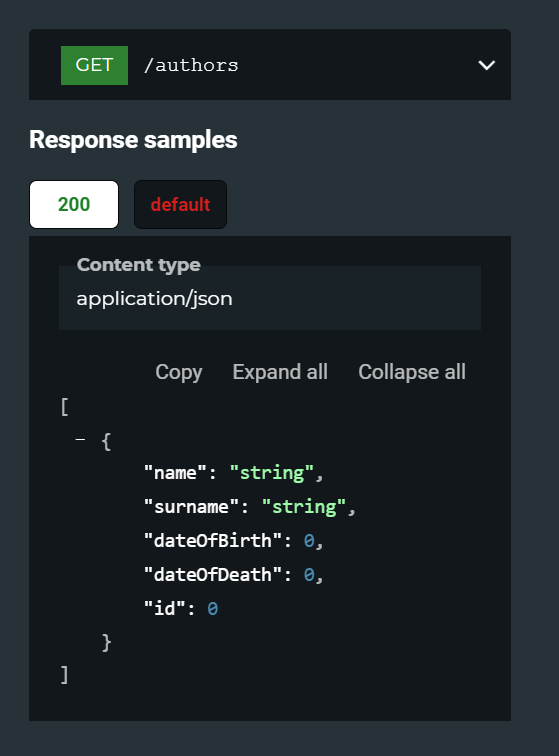


Рисунок 5. Запрос в ReDoc

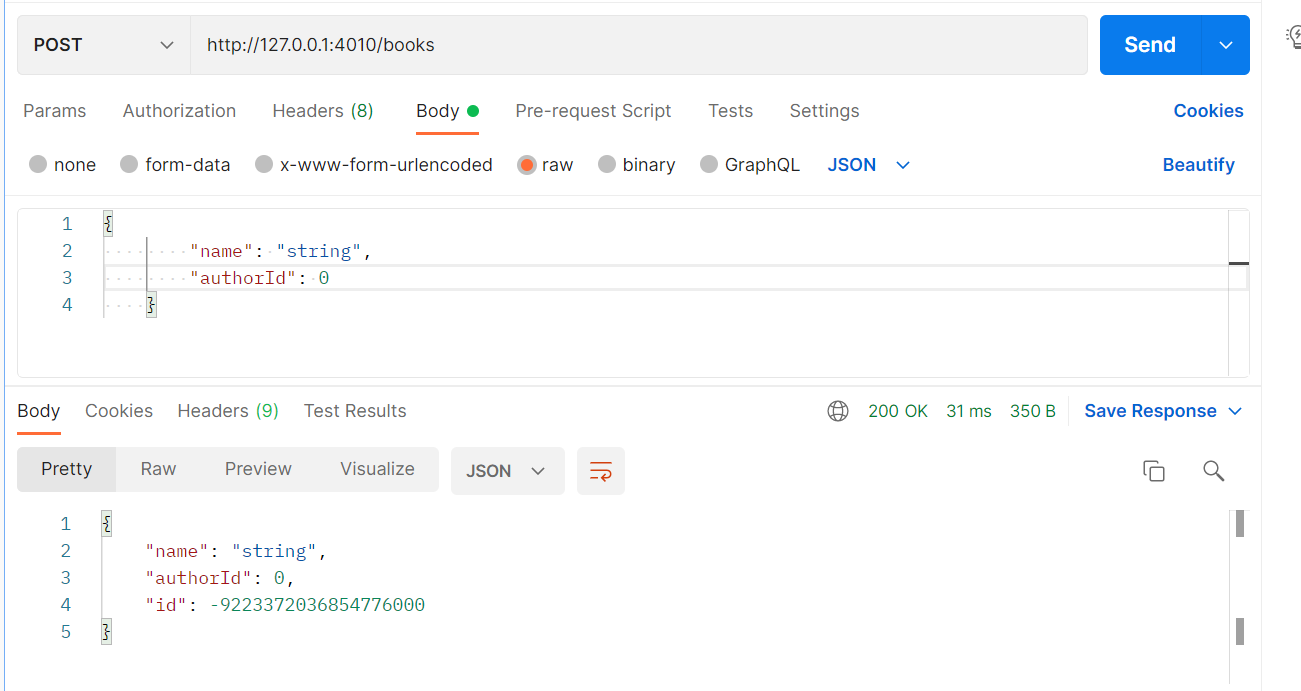


Рисунок 6. Запрос в Postman

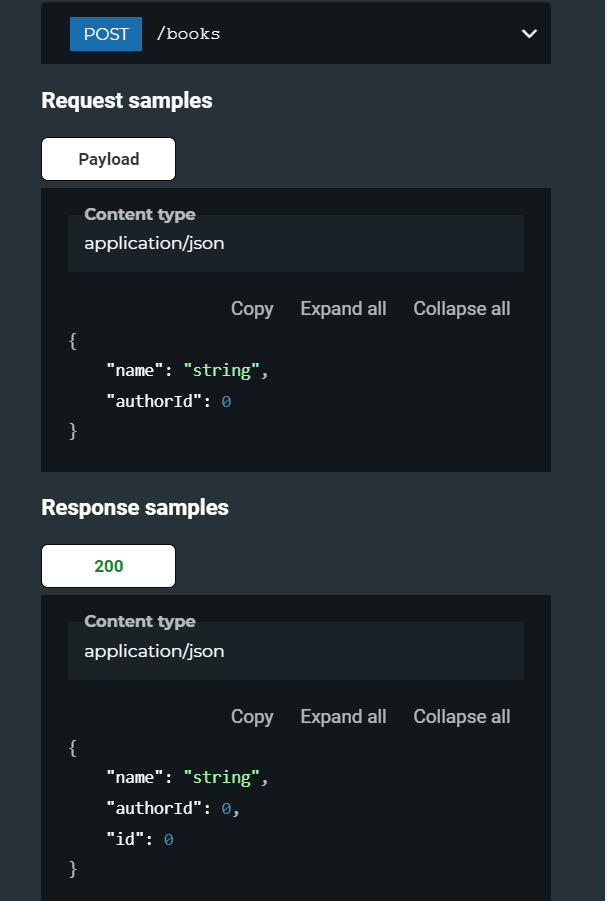


Рисунок 7. Запрос в ReDoc

# Вывод

В этой работе я ознакомился с основными подходами к построению API и инструментарием, используемый для ускорения процесса разработки API, в части документирования интерфейса прикладного программирования.