

**МИНОБРНАУКИ РОССИИ**

**федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«Московский государственный технологический университет «СТАНКИН»**

**(ФГБОУ ВО «МГТУ «СТАНКИН»)**

|  |  |
| --- | --- |
| **Институт**  **информационных**  **технологий** | **Кафедра**  **информационных технологий и вычислительных систем** |

ОТЧЕТ О ВЫПОЛНЕНИИ  
ИНДИВИДУАЛЬНОГО ЗАДАНИЯ ПО ДИСЦИПЛИНЕ

«Разработка web-приложений»

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| СТУДЕНТА | *1* | КУРСА | | *магистратуры* | ГРУППЫ | *ИДМ-21-08* |
|  | | | *(уровень профессионального образования)* | |  | |

|  |
| --- |
| **КАЛЬНИЦКОЙ КРИСТИНЫ ГЕННАДЬЕВНЫ** |
| *(ФИО)* |

НА ТЕМУ

|  |
| --- |
| Спецификации OpenApi (swagger) |

|  |  |
| --- | --- |
| Направление: | 09.04.04 Программная инженерия |
| Профиль подготовки: | Разработка и сопровождение программного обеспечения |

|  |  |  |  |
| --- | --- | --- | --- |
| Отчет сдан «\_\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_\_г. | | | |
|  |  |  |  |
| Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | | | |
|  | | | |
| Преподаватель | Коленкин В.А. |  |  |
|  | *(Ф.И.О., должность, степень, звание.)* |  | *(подпись)* |

МОСКВА 2022

В системе домашней библиотеки реализованы пять запросов, которые позволяют заносить книги в базу данных и отмечать их статус (прочитано, непрочитано), а также искать книги по статусу и ID.

Для того, чтобы избежать дублирования кода, yaml-файл включает в себя 4 компонента: **Book**, **Books**, **Error**, **Genres**. Компонент **Books** это массив объектов **Book**. Компонент **Book** можно охарактеризовать четырьмя полями, два из которых обязательны. Обязательные параметры: **id**, **name** (наименование). Дополнительные параметры: **genres** (жанр книги) и **status** (два возможных значения status: **read**, **unread**). Компонент **Genres** содержит поля **id** и **name** (наименование жанра). **Error** включат в себя два параметра: **code** (код ошибки) и **messege** (сообщение ошибки).

В данной спецификации OpenApi для пользователя предусмотрена возможность выполнить 5 действий:

1. Добавить книгу;

2. Обновить книгу;

3. Получить список книг;

4. Получить книгу по ID;

5. Получить список книг с фильтром по статусу.

Добавление книги реализовано методом POST. Обновление книги реализовано методом PUT.

В запросах PUT(обновление) и POST(добавление) основного пути **/book** в теле запроса содержится экземпляр класса **book**.

Описаны два QUERY запроса (**findByStatus**, и GET **/book** возвращающих массив из book) и один PATH GET запрос по пути GET **/book/{bookId}**.

Примеры запросов:

POST /book

**{**

**"id": 105,**

**"name": "Harry Potter",**

**"genres": {**

**"id": 1,**

**"name": "fantasy"**

**},**

**"status": "read"**

**}**

PUT /book

**{**

**"id": 105,**

**"name": "Harry Potter",**

**"genres": {**

**"id": 1,**

**"name": "fantasy"**

**},**

**"status": "read"**

**}**

GET/book/{bookIs}

Запрос на получение книги по определенному ID, указанному в пути запроса.

**{**

**"id": 105,**

**"name": "Harry Potter",**

**"genres": {**

**"id": 1,**

**"name": "fantasy"**

**},**

**"status": "read"**

**}**

Получение массива из BOOK – BOOKS.

GET /book

Запрос на получение списка книг выбранной длины (limit) с определенного ID (startId).

**curl -X 'GET' \**

**'http://myhomebookproject.io/v1/book?limit=50&startId=1' \**

**-H 'accept: application/json'**

GET /book/findByStatus

Запрос на получение прочитанных книг (status: read).

**curl -X 'GET' \**

**'http://myhomebookproject.io/v1/book/findByStatus?status=read' \**

**-H 'accept: application/json'**

Полный yaml-конфиг в формате OpenAPI 3.0.\*:

openapi: "3.0.0"

info:

version: 1.0.0

title: Books Info System

contact:

name: Kristina

url: http://www.example.com/support

email: support@example.com

servers:

- url: http://myHomeBookProject.io/v1

paths:

/book:

post:

tags:

- book

description: Add a new book to the system

summary: Add a new book

operationId: createBook

requestBody:

description: Book object that needs to be added to the system

required: true

content:

application/json:

schema:

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

parameters:

- in: query

name: book

description: Book object that needs to be added to the system

required: true

schema:

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

responses:

'200':

description: Successful Book Creation

content:

text/plain:

schema:

type: string

example: Success Create

'201':

description: Null response

'405':

description: Invalid input

default:

description: Unexpected error

content:

application/json:

schema:

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

put:

tags:

- book

description: Update an existing book in the system

summary: Update an existing book

operationId: updateBook

requestBody:

description: Book object that needs to be added to the system

required: true

content:

application/json:

schema:

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

responses:

'200':

description: Successful Book Update

content:

text/plain:

schema:

type: string

example: Success Update

'400':

description: Invalid ID supplied

'404':

description: Book not found

'405':

description: Validation exception

get:

tags:

- books

description: A JSON array of books

summary: Returns a list of books.

operationId: listBooks

parameters:

- name: limit

in: query

description: How many books to return at one time

required: false

schema:

type: integer

format: int32

default: 1

maximum: 100

example: 50

- name: startId

in: query

description: Starting from which id we look at books

required: false

schema:

type: integer

format: int64

default: 1

responses:

200:

description: Success

content:

application/json:

schema:

$ref: "#/components/schemas/Books"

default:

description: Unexpected error

content:

application/json:

schema:

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

/book/{bookId}:

get:

summary: Info for a specific book

operationId: showBookById

tags:

- book

parameters:

- name: bookId

in: path

description: The id of the book to retrieve

required: true

schema:

type: integer

format: int64

minimum: 1

responses:

'200':

description: Expected response to a valid request

content:

application/json:

schema:

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

default:

description: Unexpected error

content:

application/json:

schema:

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

/book/findByStatus:

get:

tags:

- books

summary: Finds Books by status

description: ""

operationId: findBooksByStatus

parameters:

- name: status

in: query

description: Status values that need to be considered for filter

required: true

schema:

type: string

enum:

- unread

- read

default: unread

responses:

'200':

description: Successful operation

content:

application/json:

schema:

$ref: "#/components/schemas/Books"

'400':

description: Invalid status value

components:

schemas:

Book:

type: object

required:

- id

- name

properties:

id:

type: integer

format: int64

name:

type: string

genres:

$ref: "#/components/schemas/Genres"

status:

type: string

description: Book status in the system

enum:

- unread

- read

example:

id: 105

name: Harry Potter

genres:

id: 1

name: fantasy

status: read

Books:

type: array

items:

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

Genres:

type: object

properties:

id:

type: integer

format: int64

name:

type: string

Error:

type: object

required:

- code

- message

properties:

code:

type: integer

format: int32

message:

type: string

example:

code: 100

message: Error