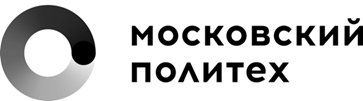
Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

«МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

(МОСКОВСКИЙ ПОЛИТЕХ)



ЛАБОРАТОРНАЯ РАБОТА 8

**ПРОЕКТИРОВАНИЕ API. ПРОЕКТИРОВАНИЕ REST API.**

По курсу

**Проектирование Веб-сервисов**

ВыполнилКилеев С.И**.**

**студент группы 211-321**

Проверил

Пардаев А.А.

Москва, 2024

**Лабораторная работа 8**

Цель работы: Получить навыки проектирования REST API.

Задачи:

КОМАНДНАЯ РАБОТА

1. Создать базу данных по модели из лабораторной работы 4 (указать название базы данных, домен/IP адрес, порт (если имеются)).

ИНДИВИДУАЛЬНАЯ РАБОТА

1. Указать выбранную функцию для реализации из лабораторной работы 4.
2. Спроектировать API для созданной базы данных:

a. составить перечень операций для реализации выбранной функции;

b. разработать структуру маршрутов (endpoints);

1. Составить спецификацию REST API, определив URI для каждого ресурса и описать доступные методы и ожидаемые ответы для каждой операции.

**Отчёт по выполнению:**

openapi: 3.0.3

info:

title: Forum Service - OpenAPI 3.0

version: 1.0.11

servers:

- url: https://itforum.swagger.io/api/v3

tags:

- name: user

description: Operations about user

paths:

/user:

commentary:

tags:

- user

summary: Create user

description: This can only be done by the logged in user.

operationId: createUser

requestBody:

description: Created user object

content:

application/json:

schema:

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

application/xml:

schema:

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

application/x-www-form-urlencoded:

schema:

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

responses:

default:

description: successful operation

content:

application/json:

schema:

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

application/xml:

schema:

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

/user/createWithList:

post:

tags:

- user

summary: Creates list of users with given input array

description: Creates list of users with given input array

operationId: createUsersWithListInput

requestBody:

content:

application/json:

schema:

type: array

items:

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

responses:

'200':

description: Successful operation

content:

application/json:

schema:

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

application/xml:

schema:

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

default:

description: successful operation

/user/login:

get:

tags:

- user

summary: Logs user into the system

description: ''

operationId: loginUser

parameters:

- name: username

in: query

description: The user name for login

required: false

schema:

type: string

- name: password

in: query

description: The password for login in clear text

required: false

schema:

type: string

responses:

'200':

description: successful operation

headers:

X-Rate-Limit:

description: calls per hour allowed by the user

schema:

type: integer

format: int32

X-Expires-After:

description: date in UTC when token expires

schema:

type: string

format: date-time

content:

application/xml:

schema:

type: string

application/json:

schema:

type: string

'400':

description: Invalid username/password supplied

/user/logout:

get:

tags:

- user

summary: Logs out current logged in user session

description: ''

operationId: logoutUser

parameters: []

responses:

default:

description: successful operation

/user/{username}:

get:

tags:

- user

summary: Get user by user name

description: ''

operationId: getUserByName

parameters:

- name: username

in: path

description: 'The name that needs to be fetched. Use user1 for testing. '

required: true

schema:

type: string

responses:

'200':

description: successful operation

content:

application/json:

schema:

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

application/xml:

schema:

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

'400':

description: Invalid username supplied

'404':

description: User not found

put:

tags:

- user

summary: Update user

description: This can only be done by the logged in user.

operationId: updateUser

parameters:

- name: username

in: path

description: name that need to be deleted

required: true

schema:

type: string

requestBody:

description: Update an existent user in the forum

content:

application/json:

schema:

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

application/xml:

schema:

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

application/x-www-form-urlencoded:

schema:

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

responses:

default:

description: successful operation

delete:

tags:

- user

summary: Delete user

description: This can only be done by the logged in user.

operationId: deleteUser

parameters:

- name: username

in: path

description: The name that needs to be deleted

required: true

schema:

type: string

responses:

'400':

description: Invalid username supplied

'404':

description: User not found

components:

schemas:

User:

type: object

properties:

id:

type: integer

format: int64

example: 10

name:

type: string

example: “TEST\_USER\_NAME”

email:

type: string

example: “TEST@MAIL.RU”

password:

type: string

example: '123457'

oauth\_code:

type: string

example: 'SlAV32hkKG'

Status:

type: integer

description: User Status

format: int32

example: 1

Role:

type: integer

description: User Status

format: int32

example: 0

Avatar:

type: string

description: User Avatar

example: “TEST\_USER\_AVATAR”

securitySchemes:

petstore\_auth:

type: oauth2

flows:

implicit:

authorizationUrl: https://itforum.swagger.io/oauth/authorize

scopes:

write:users: modify user in your account

read:users: read your users

api\_key:

type: apiKey

name: api\_key

in: header