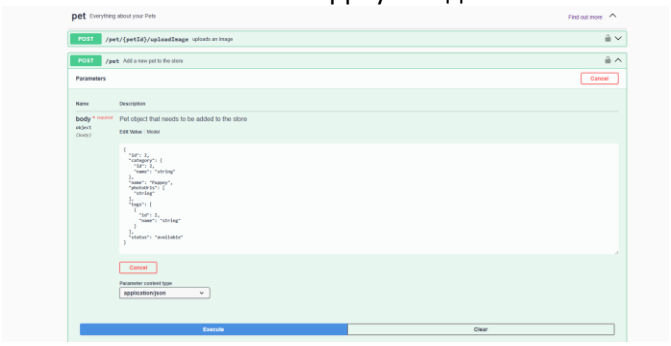
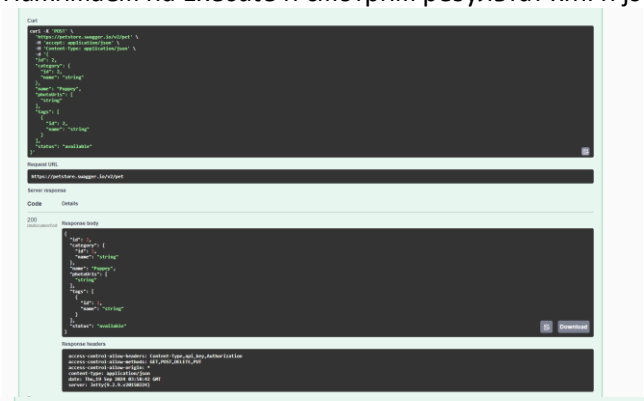
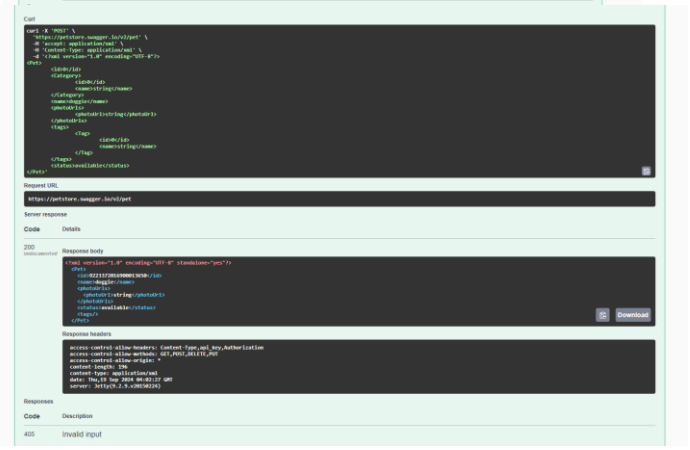
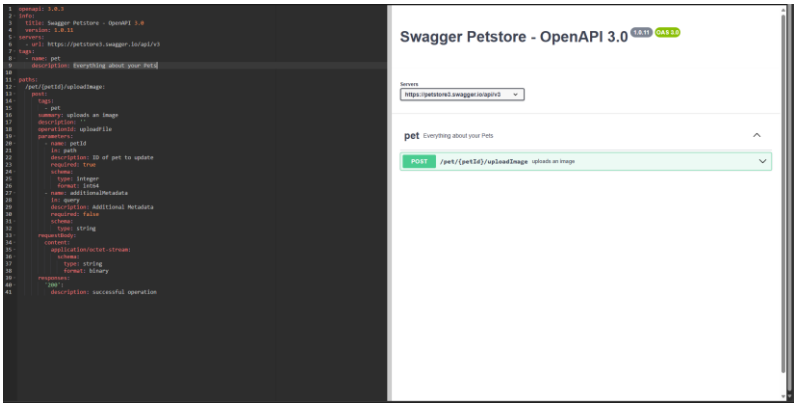
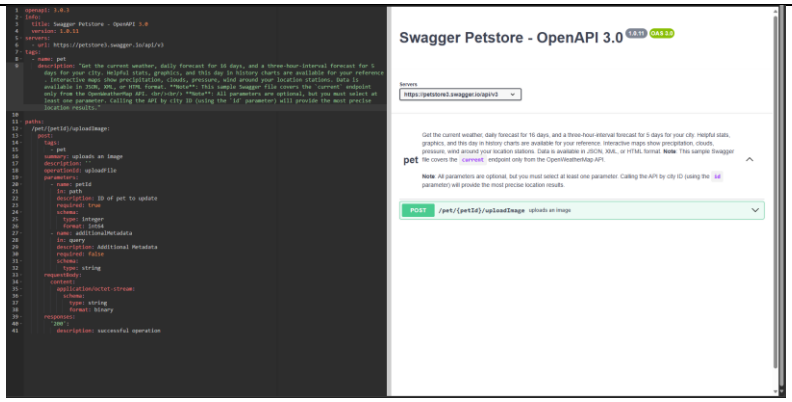
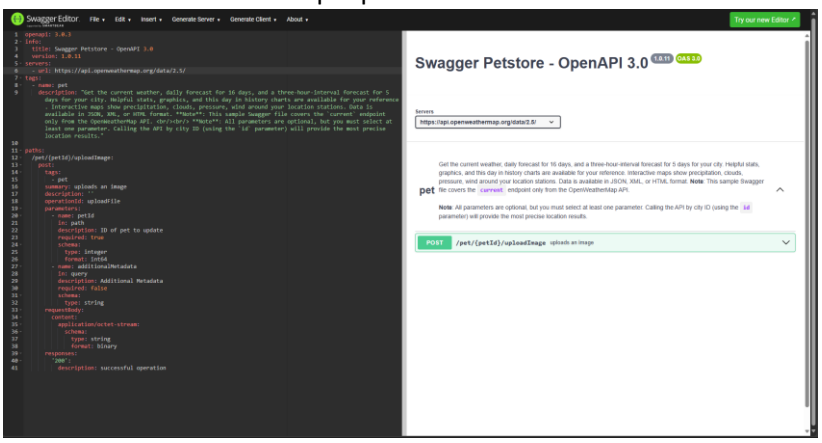
Тема лабораторной работы	lb_5.1
Выполняющий	Тысленко Егор
Помогающий	Запрегалов Николай
Ход работы	1. Меняем имя песика на Puppey и айди на 2  2. Нажимаем на Execute и смотрим результат xml и json   3. Вставляем код  4. Меняем дескрипшен



4.1 Меняем серверс



5. Разворачиваем GET и вставляем ключ

string (query) Copenhagen

id — string (query) — **City ID**. *Example:* `2172797` . You can call by city ID. The API responds with the exact result. The List of city IDs can be downloaded here. You can include multiple cities in this parameter — just separate them by commas. The limit of locations is 20. *Note: A single ID counts as a one API call. So, if you have city IDs, it's treated as 3 API calls.*

lat — string (query) — **Latitude**. *Example: 35.* The latitude coordinate of the location of your interest. Must use with `lon` .

lon — string (query) — **Longitude**. *Example: 139.* Longitude coordinate of the location of your interest. Must use with `lat` .

zip — string (query) — **Zip code**. Search by zip code. *Example: 95050,us.* Please note that if the country is not specified, the search uses USA as a default.

units — string (query) — **Units**. *Example: imperial.* Possible values: `standard` , `metric` , and `imperial` . When you do not use the `units` parameter, the format is `standard` by default.

imperial

lang — string (query) — **Language**. *Example: en.* You can use lang parameter to get the output in your language. We support the following languages that you can use with the corresponded lang values: Arabic - `ar` , Bulgarian - `bg` , Catalan - `ca` , Czech - `cz` , German - `de` , Greek - `el` , English - `en` , Persian (Farsi) - `fa` , Finnish - `fi` , French - `fr` , Galician - `gl` , Croatian - `hr` , Hungarian - `hu` , Italian - `it` , Japanese - `ja` , Korean - `kr` , Latvian - `la` , Lithuanian - `lt` , Macedonian - `mk` , Dutch - `nl` , Polish - `pl` , Portuguese - `pt` , Romanian - `ro` , Russian - `ru` , Swedish - `se` , Slovak - `sk` , Slovenian - `sl` , Spanish - `es` , Turkish - `tr` , Ukrainian - `ua` , Vietnamese - `vi` , Chinese Simplified - `zh_cn` , Chinese Traditional - `zh_tw` .

en

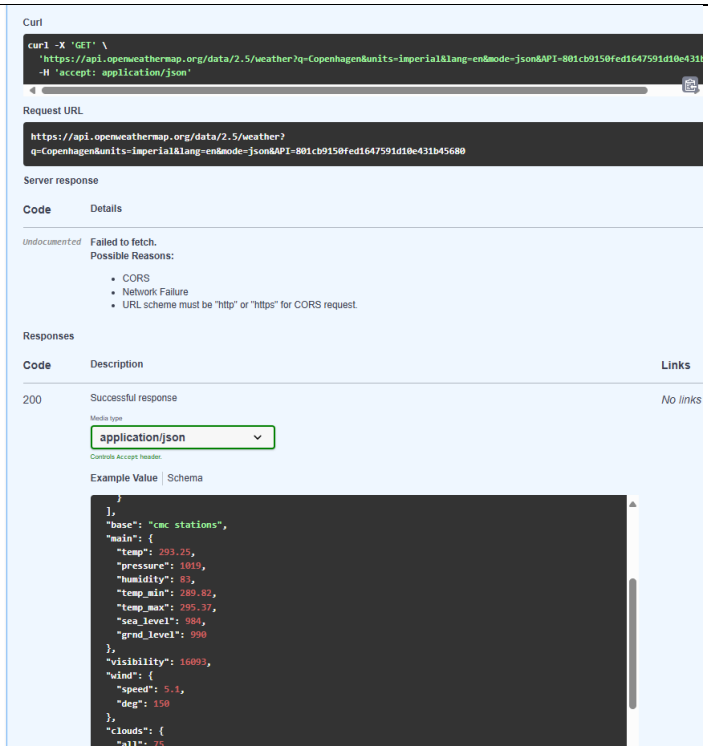
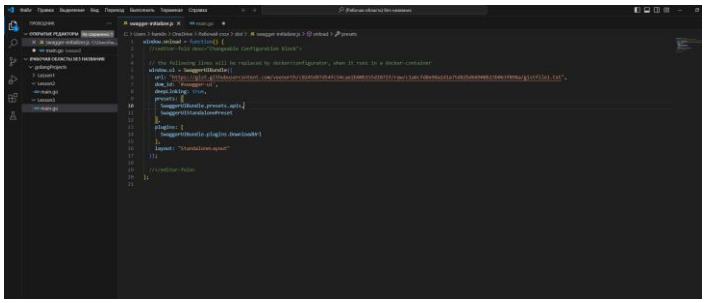
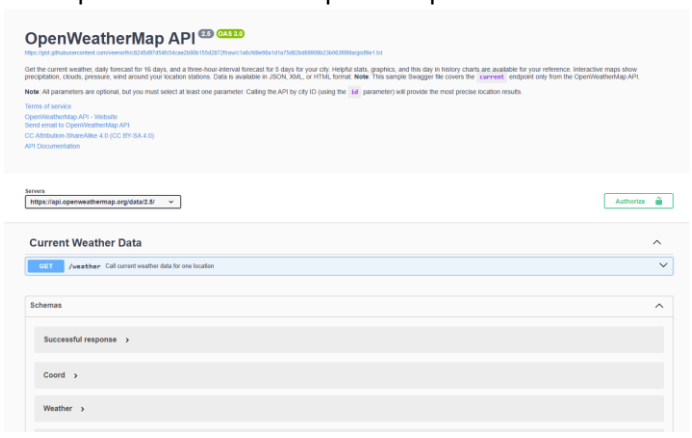
mode — string (query) — **Mode**. *Example: html.* Determines the format of the response. Possible values are `xml` and `html` . If the mode parameter is empty, the format is `json` by default.

json

API — string (query) — Write there your `API key` from OpenWeatherMap

801cb9150fed1647591d10e431b45680

Execute | Clear

	 6. Сайт не выдает ответ так что двигаемся дальше 7. Вставляем нашу Raw ссылку и скачиваем папку с гитзаба  8. Открываем сайт и смотрим что работает 
Результат	Мы научились работать в редакторе свегера, так же закрепили работу с гитхабом. Результат работы отобразился на сайте

Листинг:

```
openapi: "3.0.2"
info:
  title: "OpenWeatherMap API"
  description: "Get the current weather, daily forecast for 16 days, and a three-hour-interval forecast for 5 days for your city. Helpful stats, graphics, and this day in history charts are available for your reference. Interactive maps show precipitation, clouds, pressure, wind around your location stations. Data is available in JSON, XML, or HTML format. **Note**: This sample Swagger file covers the `current` endpoint only from the OpenWeatherMap API. <br/><br/> **Note**: All parameters are optional, but you must select at least one parameter. Calling the API by city ID (using the `id` parameter) will provide the most precise location results."
  version: "2.5"
  termsOfService: "https://openweathermap.org/terms"
  contact:
    name: "OpenWeatherMap API"
    url: "https://openweathermap.org/api"
    email: "notsayed@gmail.com"
  license:
    name: "CC Attribution-ShareAlike 4.0 (CC BY-SA 4.0)"
    url: "https://openweathermap.org/price"
servers:
  - url: "https://api.openweathermap.org/data/2.5/"
externalDocs:
  description: API Documentation
  url: https://openweathermap.org/api


paths:
  /weather:
    get:
      tags:
      - Current Weather Data
```

summary: "Call current weather data for one location"

description: "Access current weather data for any location on Earth including over 200,000 cities! Current weather is frequently updated based on global models and data from more than 40,000 weather stations."

operationId: CurrentWeatherData

parameters:

- $ref: '#/components/parameters/q'
- $ref: '#/components/parameters/id'
- $ref: '#/components/parameters/lat'
- $ref: '#/components/parameters/lon'
- $ref: '#/components/parameters/zip'
- $ref: '#/components/parameters/units'
- $ref: '#/components/parameters/lang'
- $ref: '#/components/parameters/mode'
- $ref: '#/components/parameters/appid'

responses:

200:

description: Successful response

content:

application/json:

schema:

$ref: '#/components/schemas/200'

404:

description: Not found response

content:

text/plain:

schema:

title: Weather not found

type: string

example: Not found

components:

parameters:

q:

name: q

in: query

description: "**City name**. *Example: London*. You can call by city name, or by city name and country code. The API responds with a list of results that match a searching word. For the query value, type the city name and optionally the country code divided by a comma; use ISO 3166 country codes."

schema:

type: string

id:

name: id

in: query

description: "**City ID**. *Example: `2172797`*. You can call by city ID. The API responds with the exact result. The List of city IDs can be downloaded [here](http://bulk.openweathermap.org/sample/). You can include multiple cities in this parameter — just separate them by commas. The limit of locations is 20. *Note: A single ID counts as a one API call. So, if you have city IDs, it's treated as 3 API calls.*"

schema:

type: string

lat:

name: lat

in: query

description: "**Latitude**. *Example: 35*. The latitude coordinate of the location of your interest. Must use with `lon`."

schema:

type: string

lon:

name: lon

in: query

description: "**Longitude**. *Example: 139*. Longitude coordinate of the location of your interest. Must use with `lat`."

schema:

type: string

zip:

name: zip

in: query

description: "**Zip code**. Search by zip code. *Example: 95050,us*. Please note that if the country is not specified, the search uses USA as a default."

schema:

type: string

units:

name: units

in: query

description: '**Units**. *Example: imperial*. Possible values: `standard`, `metric`, and `imperial`. When you do not use the `units` parameter, the format is `standard` by default.'

schema:

type: string

enum: [standard, metric, imperial]

default: "imperial"

lang:

name: lang

in: query

description: '**Language**. *Example: en*. You can use lang parameter to get the output in your language. We support the following languages that you can use with the corresponded lang values: Arabic - `ar`, Bulgarian - `bg`, Catalan - `ca`, Czech - `cz`, German - `de`, Greek - `el`, English - `en`, Persian (Farsi) - `fa`, Finnish - `fi`, French - `fr`, Galician - `gl`, Croatian - `hr`, Hungarian - `hu`, Italian - `it`, Japanese - `ja`, Korean - `kr`, Latvian - `la`, Lithuanian - `lt`, Macedonian - `mk`, Dutch - `nl`, Polish - `pl`, Portuguese - `pt`, Romanian - `ro`, Russian - `ru`, Swedish - `se`, Slovak - `sk`, Slovenian - `sl`, Spanish - `es`, Turkish - `tr`, Ukrainian - `ua`, Vietnamese - `vi`, Chinese Simplified - `zh_cn`, Chinese Traditional - `zh_tw`.'

schema:

type: string

enum: [ar, bg, ca, cz, de, el, en, fa, fi, fr, gl, hr, hu, it, ja, kr, la, lt, mk, nl, pl, pt, ro, ru, se, sk, sl, es, tr, ua, vi, zh_cn, zh_tw]

default: "en"

mode:

name: mode

in: query

description: "**Mode**. *Example: html*. Determines the format of the response. Possible values are `xml` and `html`. If the mode parameter is empty, the format is `json` by default."

schema:

type: string

enum: [json, xml, html]

default: "json"

appid:

name: API

in: query

```
    description: "Write there your `API key` from OpenWeatherMap"
    schema:
      type: string

schemas:
  200:
    title: Successful response
    type: object
    properties:
      coord:
        $ref: '#/components/schemas/Coord'
      weather:
        type: array
        items:
          $ref: '#/components/schemas/Weather'
        description: (more info Weather condition codes)
      base:
        type: string
        description: Internal parameter
        example: cmc stations
      main:
        $ref: '#/components/schemas/Main'
      visibility:
        type: integer
        description: Visibility, meter
        example: 16093
      wind:
        $ref: '#/components/schemas/Wind'
      clouds:
```

```
      $ref: '#/components/schemas/Clouds'
    rain:
      $ref: '#/components/schemas/Rain'
    snow:
      $ref: '#/components/schemas/Snow'
    dt:
      type: integer
      description: Time of data calculation, unix, UTC
      format: int32
      example: 1435658272
    sys:
      $ref: '#/components/schemas/Sys'
    id:
      type: integer
      description: City ID
      format: int32
      example: 2172797
    name:
      type: string
      example: Cairns
    cod:
      type: integer
      description: Internal parameter
      format: int32
      example: 200
Coord:
  title: Coord
  type: object
  properties:
```

```
    lon:
      type: number
      description: City geo location, longitude
      example: 145.77000000000001
    lat:
      type: number
      description: City geo location, latitude
      example: -16.920000000000002
Weather:
  title: Weather
  type: object
  properties:
    id:
      type: integer
      description: Weather condition id
      format: int32
      example: 803
    main:
      type: string
      description: Group of weather parameters (Rain, Snow, Extreme etc.)
      example: Clouds
    description:
      type: string
      description: Weather condition within the group
      example: broken clouds
    icon:
      type: string
      description: Weather icon id
      example: 04n
```

Main:

title: Main

type: object

properties:

temp:

type: number

description: 'Temperature. Unit Default: Kelvin, Metric: Celsius, Imperial: Fahrenheit.'

example: 293.25

pressure:

type: integer

description: Atmospheric pressure (on the sea level, if there is no sea_level or grnd_level data), hPa

format: int32

example: 1019

humidity:

type: integer

description: Humidity, %

format: int32

example: 83

temp_min:

type: number

description: 'Minimum temperature at the moment. This is deviation from current temp that is possible for large cities and megalopolises geographically expanded (use these parameter optionally). Unit Default: Kelvin, Metric: Celsius, Imperial: Fahrenheit.'

example: 289.81999999999999

temp_max:

type: number

description: 'Maximum temperature at the moment. This is deviation from current temp that is possible for large cities and megalopolises geographically expanded (use these parameter optionally). Unit Default: Kelvin, Metric: Celsius, Imperial: Fahrenheit.'

example: 295.37

sea_level:

type: number

description: Atmospheric pressure on the sea level, hPa

example: 984

grnd_level:

type: number

description: Atmospheric pressure on the ground level, hPa

example: 990

Wind:

title: Wind

type: object

properties:

speed:

type: number

description: 'Wind speed. Unit Default: meter/sec, Metric: meter/sec, Imperial: miles/hour.'

example: 5.0999999999999996

deg:

type: integer

description: Wind direction, degrees (meteorological)

format: int32

example: 150

Clouds:

title: Clouds

type: object

```
  properties:
    all:
      type: integer
      description: Cloudiness, %
      format: int32
      example: 75
Rain:
  title: Rain
  type: object
  properties:
    3h:
      type: integer
      description: Rain volume for the last 3 hours
      format: int32
      example: 3
Snow:
  title: Snow
  type: object
  properties:
    3h:
      type: number
      description: Snow volume for the last 3 hours
      example: 6
Sys:
  title: Sys
  type: object
  properties:
    type:
      type: integer
```

```
      description: Internal parameter
      format: int32
      example: 1
    id:
      type: integer
      description: Internal parameter
      format: int32
      example: 8166
    message:
      type: number
      description: Internal parameter
      example: 0.0166
    country:
      type: string
      description: Country code (GB, JP etc.)
      example: AU
    sunrise:
      type: integer
      description: Sunrise time, unix, UTC
      format: int32
      example: 1435610796
    sunset:
      type: integer
      description: Sunset time, unix, UTC
      format: int32
      example: 1435650870

securitySchemes:
  app_id:
```

type: apiKey

description: API key to authorize requests. If you don't have an OpenWeatherMap API key, use `fd4698c940c6d1da602a70ac34f0b147`.

name: appid

in: query