Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

"Российский экономический университет имени Г. В. Плеханова"

**МОСКОВСКИЙ ПРИБОРОСТРОИТЕЛЬНЫЙ ТЕХНИКУМ**

Специальность 09.02.07 Информационные системы и программирование

Квалификация: Программист

ОТЧЕТ

КОД БУДУЩЕГО

Листов: 18

|  |  |
| --- | --- |
| Выполнила студент | Проверил преподаватель |
| Группы П50-5-20 | \_\_\_\_\_\_\_\_\_\_Д.В. Серяк |
| Завьялов М.А. | " " \_\_\_\_\_\_\_\_2023 года |

Москва 2023

# Практическая работа №5

## Тема «Использование API».

Цель работы: изучить работу http запросов.

Ход работы:

2. Создайте скрипт, который будет получать фотографии котов по запросу. Запрос может быть, как обычный, так и с дополнительными параметрами, для получения конкретных фотографий. В обработке и отображении картинок вам может помочь пакет Pillow.



Рисунок 1 – Создание gif объекта

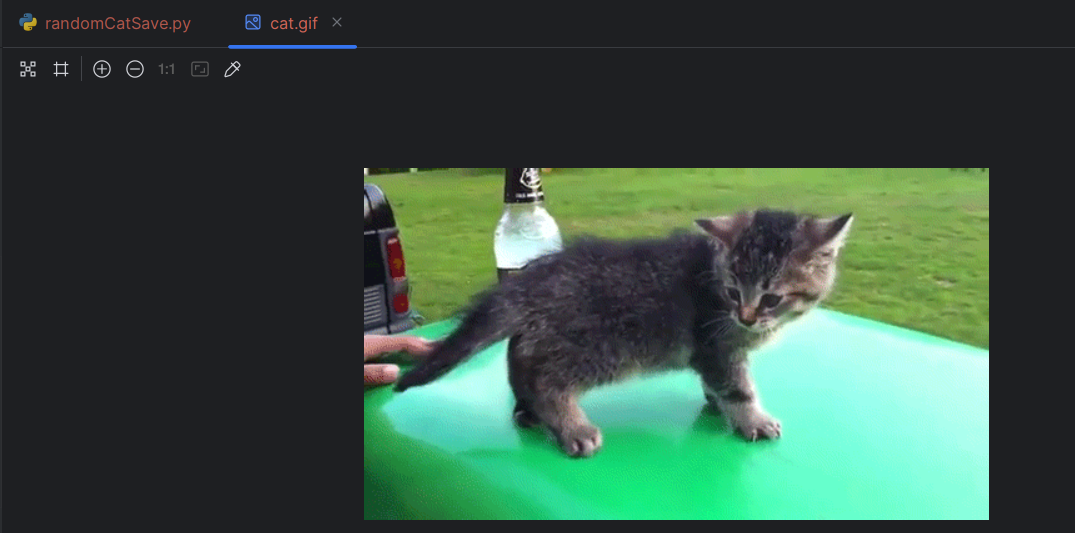


Рисунок 2 – Gif объект

randomCatSave.py

import requests

from PIL import Image

from io import BytesIO

response = requests.get('https://cataas.com/cat/gif')

img = Image.open(BytesIO(response.content))

img.save('cat.gif')

1. Создайте скрипт по обработке ответов праздников по странам. Позвольте пользователю выбирать страну п после выбора выведите все праздники и даты полученные от API ответа

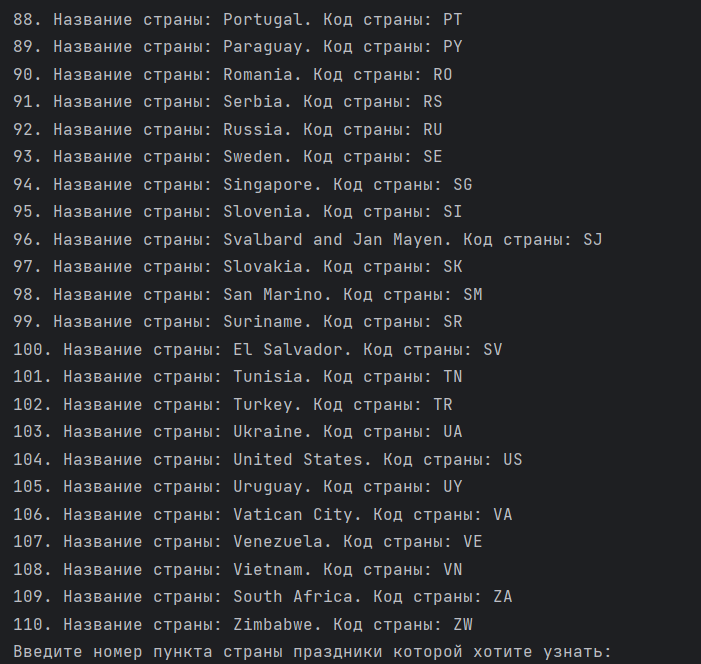


Рисунок 3 – Страны

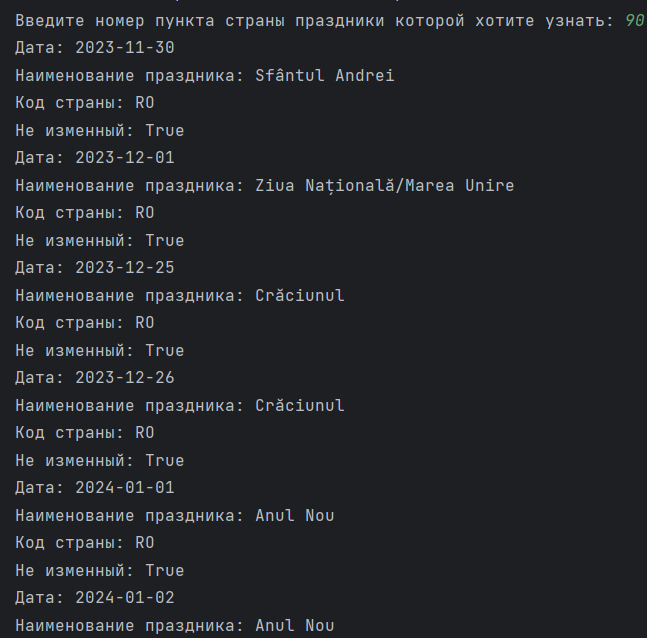


Рисунок 4 – Праздники

contryHoliday.py

import requests

import json

from types import SimpleNamespace

responseAvailableCountries = requests.get("https://date.nager.at/api/v3/AvailableCountries").content

AvailableCountries = json.loads(responseAvailableCountries, object\_hook=lambda d: SimpleNamespace(\*\*d))

for idx, country in enumerate(AvailableCountries):

print(f'{idx + 1}. Название страны: {country.name}. Код страны: {country.countryCode}')

id = int(input("Введите номер пункта страны праздники которой хотите узнать: "))

responseCountry = requests.get(f"https://date.nager.at/api/v3/NextPublicHolidays/{AvailableCountries[id - 1].countryCode}").content

AvailableCountry = json.loads(responseCountry, object\_hook=lambda d: SimpleNamespace(\*\*d))

for event in AvailableCountry:

print(f"Дата: {event.date}")

print(f"Наименование праздника: {event.localName}")

print(f"Код страны: {event.countryCode}")

print(f"Не изменный: {event.fixed}")

1. Напишите скрипт для получения и вывода данных с API в терминал. Позвольте пользователю выбирать или вписывать самостоятельно маршруты к API, тем самым дать выбор в использовании тех или иных функций. Документация и возможные конечные точки имеются на самих сайтах.

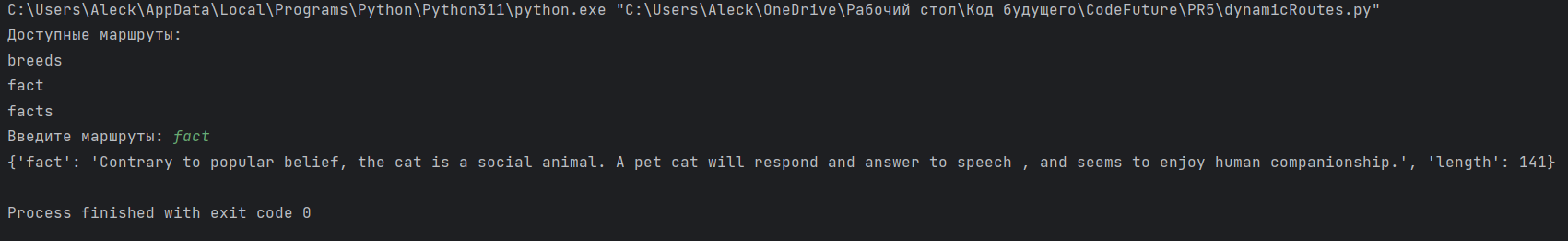


Рисунок 5 – Удачно введённый маршрут

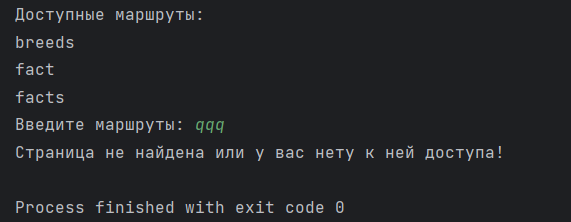


Рисунок 6 – Неудачно введённый маршрут

dynamicRoutes.py

import requests

import json

responseAvailableDocumentation = requests.get("https://catfact.ninja/docs/api-docs.json").content

AvailableDocumentation = json.loads(responseAvailableDocumentation)

print("Доступные маршруты: ")

for route in dict.keys(AvailableDocumentation['paths']):

print(route.replace('/', ''))

path = str(input("Введите маршруты: "))

response = requests.get(f"https://catfact.ninja/{path}")

if 200 <= response.status\_code <= 299:

AvailableObject = json.loads(response.content)

print(AvailableObject)

elif 400 <= response.status\_code <= 499:

print("Страница не найдена или у вас нету к ней доступа!")

elif 500 <= response.status\_code <= 599:

print("Сервер не может обработать ваш запрос в текущий момент!")

2. Создайте скрипт для получения различных данных из игры Valorant. Вы можете вывести данные по персонажам, ивентам, картам и предметам на ваше усмотрение. Минимум необходимо использовать 4 различных конечных точек.

Документация по API находится по ссылке: <https://dash.valorant-api.com/>

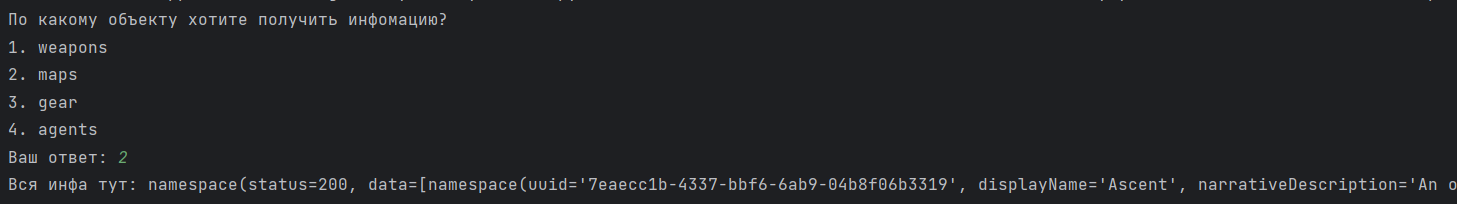


Рисунок 7 – Получение данных с API игрового проекта

valorant\_api.py

import requests

import json

from types import SimpleNamespace

themes = {

1:"weapons",

2:"maps",

3:"gear",

4:"agents"

}

print("По какому объекту хотите получить инфомацию?")

for key, theme in themes.items():

print(f"{key}. {theme}")

id = int(input("Ваш ответ: "))

responseAvailable = requests.get(f"https://valorant-api.com/v1/{themes.get(id)}/").content

info = json.loads(responseAvailable, object\_hook=lambda d: SimpleNamespace(\*\*d))

print("Вся инфа тут: " + info)

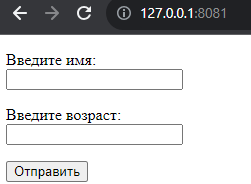


Рисунок 8 – Страница

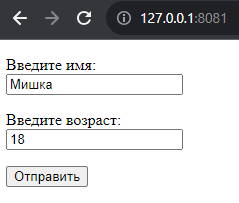


Рисунок 9 – Добавление

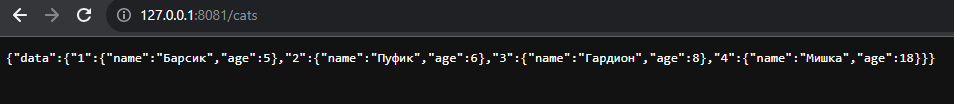


Рисунок 10 – Просмотр

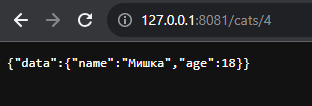


Рисунок 11 – Поиск

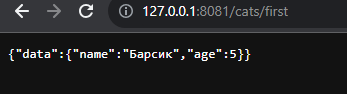


Рисунок 12 – Минимальный

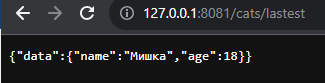


Рисунок 13 – Максимальный

CRUD.py

from pydantic import BaseModel

from typing import Optional

import uvicorn

from fastapi import FastAPI, Response, status, HTTPException

from fastapi.responses import FileResponse

def information() -> dict:

all\_cats = dict()

all\_cats[1] = (Cat(name="Барсик", age=5))

all\_cats[2] = (Cat(name="Пуфик", age=6))

all\_cats[3] = (Cat(name="Гардион", age=8))

return all\_cats

app = FastAPI(debug=True)

class Cat(BaseModel):

name: str

age: Optional[int] = 0

MEMORY\_DB = information()

@app.get("/")

def root():

return FileResponse("index.html")

@app.get("/cats")

async def index():

return {"data": MEMORY\_DB}

@app.get("/cats/first")

async def first():

return {"data": MEMORY\_DB[min(MEMORY\_DB.keys())]}

@app.get("/cats/lastest")

async def lastest():

return {"data": MEMORY\_DB[max(MEMORY\_DB.keys())]}

@app.get("/cats/{id}")

async def show(id: int, response: Response):

cat = MEMORY\_DB.get(id)

if not cat:

raise HTTPException(status\_code=status.HTTP\_404\_NOT\_FOUND, detail="Post with id not exists!")

return {"data": cat}

@app.post("/cats")

async def store(newCat:Cat):

MEMORY\_DB[len(MEMORY\_DB) + 1] = newCat.model\_dump()

return {"data": "post created", "new cat": newCat}

if \_\_name\_\_ == '\_\_main\_\_':

uvicorn.run(app, port=8081)

index.html

<!DOCTYPE html>

<html>

<head>

<title>METANIT.COM</title>

<meta charset="utf-8" />

</head>

<body>

<div id="message"></div>

<p>

Введите имя: <br />

<input name="name" id="name"/>

</p>

<p>

Введите возраст: <br />

<input name="age" id="age" type="number"/>

</p>

<button onclick="send()">Отправить</button>

<script>

async function send(){

// получаем введеное в поле имя и возраст

const name = document.getElementById("name").value;

const age = document.getElementById("age").value;

// отправляем запрос

const response = await fetch("/cats", {

method: "POST",

headers: { "Accept": "application/json", "Content-Type": "application/json" },

body: JSON.stringify({

name: name,

age: age

})

});

if (response.ok) {

const data = await response.json();

document.getElementById("message").textContent = data.message;

}

else

console.log(response);

}

</script>

</body>

</html>

Вывод: изучил работу http запросов

# Список иллюстраций

[Рисунок 1 – Создание gif объекта 2](#_Toc150650395)

[Рисунок 2 – Gif объект 2](#_Toc150650396)

[Рисунок 3 – Страны 3](#_Toc150650397)

[Рисунок 4 – Праздники 4](#_Toc150650398)

[Рисунок 5 – Удачно введённый маршрут 5](#_Toc150650399)

[Рисунок 6 – Неудачно введённый маршрут 5](#_Toc150650400)

[Рисунок 7 – Получение данных с API игрового проекта 6](#_Toc150650401)

[Рисунок 8 – Страница 7](#_Toc150650402)

[Рисунок 9 – Добавление 7](#_Toc150650403)

[Рисунок 10 – Просмотр 7](#_Toc150650404)

[Рисунок 11 – Поиск 7](#_Toc150650405)

[Рисунок 12 – Минимальный 7](#_Toc150650406)

[Рисунок 13 – Максимальный 8](#_Toc150650407)