Департамент образования и науки города Москвы

Государственное автономное образовательное учреждение

высшего образования города Москвы

«Московский городской педагогический университет»

Институт цифрового образования

Департамент информатики управления и технологий

Кузьмина Дарья Юрьевна БД-241м

Программные средства сбора, консолидации и аналитики данных

**Практическая работа 1. Сбор и анализ данных с использованием API**

**Вариант 11**

Направление подготовки/специальность

38.04.05 - Бизнес-информатика

Бизнес-аналитика и большие данные

(очная форма обучения)

Руководитель дисциплины:

Босенко Т.М., доцент департамента

информатики, управления и технологий,

доктор экономических наук

Москва

2025

Содержание

[Введение 2](#_Toc211635823)

[Основная часть 3](#_Toc211635824)

[Заключение 7](#_Toc211635825)

## Введение

**Цель**

Освоить практические навыки взаимодействия с веб-источниками данных с помощью API, включая получение, обработку и анализ информации для решения прикладных аналитических и бизнес-задач. В ходе выполнения работы формируется понимание принципов аутентификации, построения запросов к API и интерпретации полученных данных в контексте задач анализа больших данных, технологий и рыночных тенденций.

**Задачи**

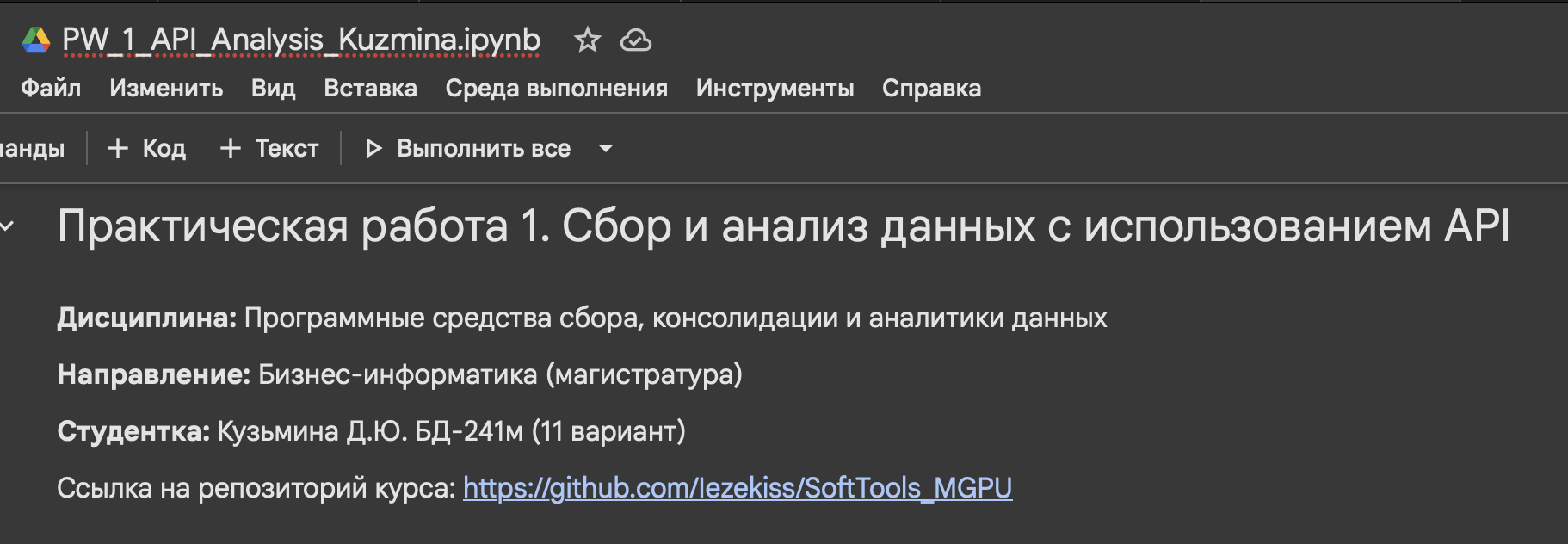
**Задачи работы**

1. Настроить рабочее окружение Python и установить необходимые библиотеки для работы с API (Requests, Pandas, Matplotlib, Seaborn, Kaggle, PyGithub).
2. Зарегистрироваться на платформах **Kaggle** и **GitHub**, получить и настроить персональные токены доступа (API Keys) для выполнения авторизованных запросов.
3. На основе индивидуального варианта (№ 11) выполнить три прикладных задания:
   * **Kaggle API:** осуществить поиск датасетов по теме *analytics* и выявить датасеты, содержащие файлы формата .parquet;
   * **GitHub API:** найти топ-10 пользователей, в профиле которых указано *Data Scientist*, и провести сравнительный анализ по числу подписчиков;
   * **API hh.ru:** собрать 100 вакансий по запросу *Project Manager* и проанализировать распределение вакансий по типу графика работы (полный, сменный, гибкий).
4. Сформировать структурированные данные, выполнить их обработку и визуализировать результаты анализа с помощью инструментов Python (Pandas, Matplotlib, Seaborn).
5. Подготовить отчёт о проделанной работе и разместить исходный код в публичном Git-репозитории.

## ССЫЛКА НА GIT: <https://github.com/Iezekiss/SoftTools_MGPU>

## Основная часть

Настроим рабочее окружение и познакомимся с предложенным для практики блокнотом. Оформим его в соответствии со своими данными.



Изображение выглядит как текст, снимок экрана, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.

**1.1 Настройка доступа.**  
Был сгенерирован и подключён персональный токен kaggle.json, размещён в каталоге ~/.kaggle/.  
Работа велась через CLI-интерфейс Kaggle для стабильного получения списков файлов.

!kaggle datasets list -s analytics --csv > /content/kaggle\_analytics.csv

df\_all = pd.read\_csv("/content/kaggle\_analytics.csv")

Выполним подготовку API для Kaggle

Изображение выглядит как текст, снимок экрана, Шрифт, число

Содержимое, созданное искусственным интеллектом, может быть неверным.Изображение выглядит как текст, Шрифт, снимок экрана

Содержимое, созданное искусственным интеллектом, может быть неверным.

Изображение выглядит как текст, снимок экрана, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.

**1.2 Первичный поиск.**  
Выполнен запрос по ключевому слову analytics, получен перечень ≈ 100 датасетов с базовыми метаданными (название, владелец, загрузки, голоса).

**1.3 Фильтрация по формату .parquet.**  
Для каждого датасета был запрошен список файлов, реализована проверка наличия .parquet.

import subprocess, shlex

def has\_parquet(ref):

out = subprocess.run(f'kaggle datasets files {shlex.quote(ref)} --csv',

shell=True, capture\_output=True, text=True)

return '.parquet' in out.stdout.lower()

df\_all['has\_parquet'] = df\_all['ref'].apply(has\_parquet)

**1.4 Расширенный поиск.**  
Так как базовый запрос не дал совпадений, был выполнен дополнительный — analytics parquet.  
Найдены отдельные датасеты, использующие формат .parquet в задачах Big Data.

**1.5 Визуализация.**  
Построена круговая диаграмма доли .parquet-датасетов от общего числа найденных, оформлена таблица-витрина (название, владелец, скачивания, голоса, ссылка).

Изображение выглядит как текст, снимок экрана, Шрифт, диаграмма

Содержимое, созданное искусственным интеллектом, может быть неверным.

## 2. GitHub API — поиск экспертов с «Data Scientist» в био

**2.1 Подготовка доступа.**  
Создан токен GitHub Personal Access Token, подключён к среде как переменная GITHUB\_TOKEN.

**2.2 Поиск пользователей.**  
Реализован запрос к эндпоинту /search/users с параметрами q="Data Scientist in:bio type:user".  
Получено ~200 профилей-кандидатов.

params = {"q": "Data Scientist in:bio type:user", "per\_page": 100, "page": 1}

data = requests.get("https://api.github.com/search/users", headers=HEADERS, params=params).json()

logins = [u['login'] for u in data['items']]

**2.3 Детализация профилей.**  
Для каждого логина выполнен запрос /users/{login} для получения followers, company, location, bio, html\_url.

**2.4 Фильтрация и сортировка.**  
Оставлены профили, где в описании встречается «Data Scientist».  
Данные отсортированы по числу подписчиков, сформирован топ-10 экспертов.

df\_users = pd.DataFrame(rows)

df\_users = df\_users[df\_users["bio"].str.contains("data scientist", case=False, na=False)]

top10 = df\_users.sort\_values("followers", ascending=False).head(10)

**2.5 Визуализация.**  
Построена горизонтальная столбчатая диаграмма подписчиков по логинам.  
Отдельно оформлена таблица top-10 с именами, компаниями и ссылками на профили.

Изображение выглядит как текст, снимок экрана, Шрифт, программное обеспечение

Содержимое, созданное искусственным интеллектом, может быть неверным.

## 3. hh.ru API — анализ графика работы для вакансий «Project Manager»

**3.1 Получение данных.**  
Сформирован запрос к https://api.hh.ru/vacancies с параметрами text="Project Manager", per\_page=100.  
Получено 100 актуальных вакансий.

params = {"text": "Project Manager", "per\_page": 100}

data = requests.get("https://api.hh.ru/vacancies", params=params).json()

items = data["items"]

**3.2 Извлечение ключевых полей.**  
Из каждого объекта выделены id, name, employer, schedule.id, schedule.name.

**3.3 Классификация графиков.**  
Создана карта соответствий:  
fullDay → Полный день, shift → Сменный, flexible → Гибкий, остальные → «Другое».

map\_core = {"fullDay":"Полный день","shift":"Сменный","flexible":"Гибкий"}

df\_hh["schedule\_group"] = df\_hh["schedule\_id"].map(map\_core).fillna("Другое")

**3.4 Распределение и доли.**  
Подсчитано количество вакансий по каждому типу графика и их доля (в %).

**3.5 Визуализация.**  
Построена столбчатая диаграмма: доминируют вакансии с полным днём; гибкий и сменный форматы встречаются реже.

Изображение выглядит как текст, снимок экрана, диаграмма, линия

Содержимое, созданное искусственным интеллектом, может быть неверным.

# Заключение

**Вывод**:

В ходе выполнения практической работы были получены и закреплены навыки взаимодействия с тремя прикладными API-сервисами — **Kaggle**, **GitHub** и **hh.ru**, предназначенными для решения аналитических задач различного уровня. Работа охватывала полный цикл обработки данных: от подключения и авторизации до анализа и визуализации полученных результатов.

На первом этапе была выполнена интеграция с Kaggle API. Проведён поиск датасетов по теме «analytics» и реализована фильтрация по формату файлов .parquet. В ходе эксперимента установлено, что при базовом поисковом запросе подобные датасеты встречаются редко, однако расширенный запрос «analytics parquet»позволил выявить отдельные примеры, подтверждающие использование формата .parquet в задачах обработки больших данных.

На втором этапе с использованием GitHub API был проведён поиск пользователей, указавших в описании профиля «Data Scientist». Сформирован топ-10 экспертов по количеству подписчиков, выполнена визуализация распределения и сделаны выводы о высокой концентрации специалистов с открытыми профилями и активной аудиторией в данной области.

На заключительном этапе применён API hh.ru для анализа 100 вакансий «Project Manager». Проведено распределение вакансий по типу графика работы — полный, сменный и гибкий. Анализ показал доминирование вакансий с полным днём, что отражает текущие тенденции российского рынка труда в сфере управления проектами.

Практическая работа позволила на практике освоить подходы к интеграции внешних источников данных через API, формированию выборок, их первичному анализу и представлению результатов в визуальной форме. Полученные навыки являются базовыми для дальнейших задач по консолидации и аналитике данных в рамках курса.