Цифриум

Технологическая карта

Умный код на Python. Базовый уровень

Модуль 3 | Тема 4 | Урок 4

Интеграция с web-сервисами: обзор API, настройка взаимодействия

Уважаемый коллега, обратите внимание!

Учёт присутствия ребёнка на уроке ведётся только по цифровому следу. Каждый ученик из группы на уроке должен передать цифровой след на нашей платформе.

Вы отвечаете за передачу следа учеником на нашей платформе во время урока.

Передавать цифровой след нужно в личном кабинете ученика.

Цифровой след засчитывается по итогу выполнения учеником трёх задач на платформе в ходе урока.

Обращаем внимание — в задачах с отправкой ответа в виде файла наша платформа поддерживает следующие форматы: `.pdf, .doc, .png, .jpeg, .xlsx, .mp3, .py, .ipynb, .txt, .csv, .json, .xml, .sb3, .ino, .hex`.

Контролируйте время урока, чтобы не пропустить момент, когда ученики должны залогиниться в личном кабинете и перейти к решению задач на платформе.

Проследите, чтобы каждый ученик залогинился в личном кабинете и выполнил 3 задачи на платформе. В противном случае, Вам не будет засчитано, что вы провели урок, занятие не будет оплачено, а ученику не будет выставлена отметка о присутствии на уроке.

Просим вас в ходе 45 минут урока проконтролировать, чтобы:

1. Каждый из учеников залогинился в личном кабинете.
2. Каждый ученик в разделе **«Решить задания»** отправил решения по 3 задачам — «Практическое задание 1», «Практическое задание 2» и «Практическое задание 3».

О занятии

Краткое содержание занятия:

1. Узнаете, что такое REST и WebSocket и в чём разница между ними.
2. Поймёте, что такое API и как он помогает программам общаться.
3. Изучите, что такое Webhook и зачем он нужен.
4. Научитесь создавать Telegram-бота через BotFather.
5. Подключите бота к GPT через API и протестируете работу бота с реальными сообщениями.

Цель:

Научиться интегрировать GPT с внешними сервисами через REST API и WebSocket, а также создать простой Telegram-бот.

Задачи:

1. Понять различия между REST и WebSocket для обмена данными.
2. Изучить настройку и использование webhook для получения событий.
3. Настроить Telegram-бота: регистрация в BotFather, получение токена.
4. Реализовать взаимодействие бота с GPT через API и протестировать сценарии«запрос-ответ»..

Термины

REST (Representational State Transfer)
Архитектурный стиль взаимодействия компонентов через HTTP, где каждое состояние представлено ресурсом с уникальным URL. Аналогия: как почтовая служба — каждый запрос это отдельное письмо.

О занятии

WebSocket
Протокол для двунаправленной постоянной связи между клиентом и сервером, позволяющий отправлять данные в реальном времени.
Аналогия: как телефонный звонок — соединение держится всё время.
API (Application Programming Interface)
Набор методов и правил, позволяющих разным программным компонентам взаимодействовать друг с другом. Переводчик между программами.
Webhook
Механизм обратного вызова, при котором сервер отправляет HTTP-запрос на заранее настроенный URL при наступлении определённых событий.
Bot Token
Уникальный ключ доступа, выдаваемый сервисом (например, Telegram), позволяющий идентифицировать и авторизовать бота.
Endpoint
Конкретный URL или адрес API, по которому клиент может отправлять запросы для выполнения определённого метода или получения ресурса.
BotFather
Официальный бот Telegram для создания и управления другими ботами.
Polling
Способ получения обновлений, при котором бот регулярно опрашивает сервер на наличие новых сообщений.

Переменные окружения

Специальные переменные операционной системы для хранения конфиденциальной информации (например, API-ключей).

Необходимые материалы:

1. Установленный Python ≥ 3.10
2. Библиотеки python-telegram-bot, requests
3. Тестовый Telegram-бот
4. Интернет-доступ

Темы и время

ЭТАП	ВРЕМЯ
Приветствие	5 мин.
Теория	25 мин.
Итоги занятия	5 мин.
Получение фидбека. Рефлексия. Проверка знаний. Домашнее задание	5 мин.
Вопрос-ответ	5 мин.
Итого	45 мин.

Ход занятия

Номер слайда	Пояснение к слайду
1	Приветствие учеников. Здравствуйте, ребята! Сегодня мы начинаем новую тему – интеграция с веб-сервисами. Мы научимся создавать Telegram-бота, который будет общаться с GPT. Представьте, что вы можете создать своего помощника, который всегда на связи в телефоне и отвечает на вопросы как умный друг. Сегодня мы сделаем первый шаг к этому. Не волнуйтесь, если слышите сложные слова — мы разберем всё на простых примерах из жизни.
2	Титульный слайд. Знакомство с темой урока: Интеграция с web-сервисами: обзор API, настройка взаимодействия. Звучит это сложно, но на самом деле это как конструктор. Мы просто соединим две детали: мессенджер и искусственный интеллект, чтобы они работали вместе. Представьте, что вы строите мост: с одной стороны ваш телефон, с другой — суперкомпьютер нейросети. API — это чертежи этого моста, и сегодня мы научимся строить такие мосты.
3	На прошлом занятии вы: Повторение материала предыдущего урока. Кто помнит, что мы отправляли нейросети на прошлом уроке? А что получали взамен? Отлично! Вы уже умеете общаться с мозгом GPT. Но раньше мы делали это прямо в коде или через специальную страницу, а это неудобно. Сегодня мы возьмем этот мозг и вставим его в удобную оболочку — Telegram-бота. Всё, что вы знали раньше, сейчас станет начинкой для вашего нового проекта.
4	Сегодня на занятии вы: Узнаете о REST, WebSocket, API и Webhook. Главная задача – создать своего Telegram-бота, который будет общаться с GPT. Урок будет насыщенным: сначала немного теории без скуки — разберемся, как данные летают по интернету. Потом самая вкусная часть — практика. Вы получите своего уникального бота. К концу урока вы напишите боту Привет, а он ответит вам как живой собеседник.
5	Цель и задачи урока. Наша глобальная цель — автоматизация, чтобы бот отвечал сам, без вашего участия.

Ход занятия

	Для этого нужно решить три задачи: первое — понять правила общения программ REST и WebSocket; второе — настроить так, чтобы бот сразу узнавал о новых сообщениях через Webhook; третье — написать код, который свяжет всё воедино. Все поняли задачу? Если что-то непонятно в процессе — сразу поднимайте руку, мы разберемся вместе.
6	Что такое REST. Объясняю на простой аналогии: REST – это как почта. Представьте, что вы пишете другу бумажное письмо. Вы кладете его в конверт, бросаете в ящик и ждете. Друг получает, пишет ответ, отправляет обратно. Пока письмо идет, связи между вами нет. В интернете REST работает так же: браузер просит у сервера показать сайт, сервер отдает сайт и говорит: Всё, я свободен. Чтобы спросить еще раз, нужно отправлять новый запрос. Постоянно устанавливать новое соединение для каждого сообщения — это долго и затратно. Но разработчики нашли способ заставить REST работать в таких ситуациях, и называется это поллинг (polling) или «опрос». Как думаете, удобно ли это для чата, где сообщения летят каждую секунду?
7	Что такое WebSocket. Аналогия: WebSocket – это телефонный звонок. Представьте, что вы позвонили другу. Вы соединились один раз. И теперь можете говорить минуту, час, хоть весь день. Не нужно каждый раз перезванивать для новой фразы. WebSocket — это такая трубка между программами, она открыта постоянно. Как только вы пишете сообщение в чате, оно мгновенно улетает по этому каналу. Поэтому в онлайн-играх вы видите движение других игроков без задержек. Для нашего бота мы будем использовать комбинацию подходов: WebSocket — для скорости, REST — для простых запросов.
8	Что такое API. API – это переводчик между программами. Представьте: один человек знает только русский, а робот знает только свой цифровой язык. Они не поймут друг друга. Нужен посредник. API — это набор правил, как меню в ресторане. Вы не идете на кухню готовить, вы смотрите в меню, выбираете блюдо, и официант приносит его. В нашем случае API Telegram принимает ваше сообщение, а API GPT генерирует ответ. Наш код будет тем самым официантом, который бегает между ними и обеспечивает понимание.

Ход занятия

9	Что такое Webhook. Webhook – это специальный URL, куда Telegram отправляет входящие сообщения. Представьте, вы ждете доставку пиццы. Вы можете каждую минуту выбегать на лестницу проверять курьера — устанете. Это работа без Webhook: бот бегает и спрашивает сервер: Пришло что-то? Нет? А сейчас? Webhook — это когда вы оставляете свой адрес курьеру, и он сам приезжает и звонит в дверь. Telegram сам отправляет сообщение на наш сервер, как только пользователь его напишет. Нам нужно будет указать Telegram этот адрес, чтобы он знал, куда стучаться.
10	Telegram-бот и GPT. Полная схема работы: давайте проследим путь одного сообщения. Вы пишете Привет. Telegram мгновенно будит нашу программу. Программа бежит к нейросети. Нейросеть думает. И ответ летит обратно вам в телефон. Обратите внимание: вы общаетесь с Telegram, но мозг находится у GPT. Наш код — это просто связной. Весь процесс занимает 1-3 секунды. Кто может своими словами повторить, куда идет сообщение после того, как я его отправил?
11	Практика: создаём своего Telegram-бота. Пошаговая инструкция: откройте поиск в Telegram, введите @BotFather. Это главный бот, который создает других ботов. Нажмите Запустить. Введите команду /newbot. Придумайте имя, например MySuperAIBot. Потом придумайте ссылку, обязательно чтобы в конце было слово bot, например ivanov_test_bot. BotFather выдаст длинный набор букв и цифр — это TOKEN, пароль от вашего бота. Скопируйте его и никогда не показывайте посторонним. Откройте файл с кодом, найдите строчку BOT_TOKEN и вставьте ключ внутрь кавычек. То же самое сделайте с ключом от GPT. Запустите программу. Если внизу написано Bot started, идите в Telegram и напишите своему боту любое сообщение. Я прохожу по классу, помогаю тем, у кого ошибка в коде.
12	Проверка: работает ли бот? Запускаем тест-драйв! Напишите боту Привет. Если он молчит — смотрим в черное окно консоли, там написана причина ошибки. Если ошибка Unauthorized — значит, неверно вставили токен бота. Если Connection Error — проверьте интернет. Теперь усложним: спросите у бота что-то творческое, например: Придумай название для команды программистов. Посмотрите, как

Ход занятия

	быстро он ответит. У кого бот ответил быстрее всех? У кого самый интересный ответ?
13	Сегодня на занятии вы: Подведение итогов урока. Посмотрите на себя! Еще час назад у вас не было своего бота, а теперь он есть. Вы научились интегрировать сложные сервисы. Вы поняли различия между REST и WebSocket, настроили Telegram-бота и реализовали взаимодействие с GPT. Давайте помечтаем: что еще может делать такой бот? Может, помогать с домашкой? Напоминать о расписании? Искать информацию? Эти навыки — работа с API, ботами — очень востребованы. Вы сделали шаг в профессию разработчика.
14	На следующем занятии вы: Анонс следующего урока. Сегодня мы работали в Telegram. Но представьте, если вы сможете вставить такого умного собеседника на свой личный сайт-визитку? Чтобы друзья заходили и общались с вашим ИИ. На следующем уроке мы коснемся веб-разработки. Изучим HTML — скелет сайта, и JavaScript — мышцы сайта. Мы сделаем так, чтобы нейросеть работала прямо в браузере. Домашнее задание: попробуйте изменить системное сообщение в коде, чтобы бот отвечал в стиле пирата или робота. Всем спасибо, урок окончен!

Практика

Вопросы для самопроверки

- Чем REST похож на почтовые письма?
- Почему WebSocket сравнивают с телефонным звонком?
- Зачем Telegram-боту нужен webhook?
- Как GPT «узнаёт» текст из Telegram?
- Что будет, если опубликовать свой API-ключ в интернете?

Задание 1

Создайте Telegram-бота, который отвечает на сообщения пользователей через GPT.

Шаги выполнения:

1. Получите Bot Token через BotFather и запишите его (не показывайте другим).

2. Установите библиотеки: pip install python-telegram-bot requests

3. Создайте файл bot.py и вставьте шаблон кода.

4. Замените your-api-key и your-telegram-token на свои значения.

5. Запустите скрипт.

6. Напишите боту сообщение «Привет!» и запишите ответ.

Примеры выполнения заданий

Проверь себя по следующим критериям:

REST похож на почтовые письма, потому что клиент отправляет запрос (письмо), сервер отвечает — и связь разрывается. Каждый запрос независим, как отдельное письмо.

WebSocket сравнивают с телефонным звонком, потому что устанавливается постоянное двустороннее соединение: обе стороны могут обмениваться данными в реальном времени, пока «звонок» не завершён.

Telegram-боту нужен webhook, чтобы получать обновления (сообщения) мгновенно — сервер Telegram «звонит» боту при новом событии, вместо того чтобы бот постоянно опрашивал сервер (long polling).

GPT «узнаёт» текст из Telegram, потому что бот-сервер пересылает содержимое сообщения пользователя в API GPT как текстовый запрос — модель ничего не знает о Telegram, она просто получает текст.

Если опубликовать API-ключ — им воспользуются злоумышленники: потратят ваш лимит, украдут данные или взломают интеграции. Аккаунт могут заблокировать.

Задание 1

```
import requests

import os

from dotenv import load_dotenv

from telegram import Update

from telegram.ext import Application, MessageHandler,
filters, ContextTypes

load_dotenv()



API_KEY = os.getenv("OPENROUTER_API_KEY")
```

Примеры выполнения заданий

```
MODEL = os.getenv("OPENROUTER_MODEL",
"qwen/qwen3-vl-30b-a3b-thinking")

TG_TOKEN = os.getenv("TELEGRAM_BOT_TOKEN")

if not API_KEY or not TG_TOKEN:

    print("❌ Укажите OPENROUTER_API_KEY и
TELEGRAM_BOT_TOKEN в .env")

    exit(1)

print(f"✓ Бот запущен (модель: {MODEL})")

messages = {}  # Хранение истории по user_id

async def handle_message(update: Update, context:
ContextTypes.DEFAULT_TYPE):

    user_id = update.effective_user.id

    text = update.message.text



    if user_id not in messages:

        messages[user_id] = []

    messages[user_id].append({"role": "user",
"content": "отвечай на русском" + text})

    await update.message.reply_text("🤖 Печатает...")

    response = requests.post(
```

Примеры выполнения заданий

```
    "https://openrouter.ai/api/v1/chat/completions",
        headers={
            "Authorization": f"Bearer {API_KEY}",
            "Content-Type": "application/json"
        },
        json={
            "model": MODEL,
            "messages": messages[user_id]
        },
        timeout=120
    )
    assistant_message =
response.json()["choices"][0]["message"]["content"]
    messages[user_id].append({"role": "assistant",
"content": assistant_message})
    await
update.message.reply_text(assistant_message)

app = Application.builder().token(TG_TOKEN).build()

app.add_handler(MessageHandler(filters.TEXT &
~filters.COMMAND, handle_message))
```

Примеры выполнения заданий

```
app.run_polling()
```

Сегодня на занятии вы:

- Научились интегрировать GPT с внешними сервисами через REST API и WebSocket.
- Поняли различия между REST и WebSocket для обмена данными.
- Изучили настройку и использование webhook для получения событий.
- Настроили Telegram-бота: зарегистрировались в BotFather, получили токен.
- Реализовали взаимодействие бота с GPT через API и протестировали сценарии запрос-ответ.

На следующем занятии вы:

- Познакомитесь с основами HTML и JavaScript для создания веб-интерфейса.
- Узнаете, как работают теги <input>, <button> и другие элементы форм.
- Создадите веб-страницу с формой «Задай вопрос» и подключённым script.js.
- Подключите GPT API к веб-интерфейсу и протестируете работу.

Получение фидбека. Рефлексия. Проверка знаний

Попросите ребят решить интерактивные задачи по теме урока.

Ответы к интерактивным задачам

Практическое задание 1

Как называется протокол для двунаправленной постоянной связи между клиентом и сервером в реальном времени?

WebSocket

Практическое задание 2

Что такое Webhook в контексте работы с Telegram-ботами?

- Способ хранения данных в базе
- **Механизм, при котором Telegram отправляет HTTP-запрос на указанный URL при новом сообщении**
- Библиотека для создания веб-сайтов
- Тип шифрования данных

Практическое задание 3

Что необходимо для создания Telegram-бота, который работает с GPT? (выберите все правильные варианты)

- **Bot Token от BotFather**
- **API-ключ модели**
- Собственный сервер с операционной системой Linux
- **Библиотека python-telegram-bot**

Ответы к интерактивным задачам

Вопрос-ответ

Если во время подготовки или проведения урока возникают вопросы, их необходимо адресовать в методический чат.