УДК 004.773.6:316.472.4
DOI 10.51691/2541-8327_2022_10_3

РАЗРАБОТКА ЧАТ-БОТА «БЕСТИАРИЙ» В СОЦИАЛЬНОЙ СЕТИ ВКОНТАКТЕ

Бойко А. В.
студент,
Хакасский технический институт – филиал федерального государственного автономного образовательного учреждения высшего образования «Сибирский федеральный университет»,
Абакан, Россия

Аннотация

Современные компании с целью повышения эффективности работы и уровня коммуникации с потенциальными клиентами уже не ограничиваются информационными сайтами, представляя себя в сети Интернет и переходят на сервисы, которые позволяют организовать коммуникацию пользователя с чат-ботами. В статье рассматриваются процесс создания чат-бота в социальной сети ВКонтакте, как способ автоматизации процесса получения информации, на примере бестиария. В статье обоснован выбор средств разработки чат-бота, описан процесс настройки API и необходимых библиотек. В результате исследования получен практический результат – чат-бот «Бестиарий» в социальной сети ВКонтакте.

Ключевые слова: чат-бот, социальная сеть, бестиарий, Python, API, библиотека, база данных, тестирование.

DEVELOPMENT OF CHATBOT «BESTARIJ» IN SOCIAL NETWORK «VK»

Boyko A.V.
Student,
Khakas Technical Institute – the Branch of SFU,
Abakan, Russia

Abstract

Modern companies do not make do with the websites to introduce themselves on the Internet in order to to increase the efficiency of work and the level of communication with potential customers. They turn to services which enable them to organise the interaction between a user and a chatbot. The article dwells upon the process of developing a chatbot in a social network "VK" to automate the process of getting the data, illustrating it through the example of bestiary. The article justifies the choice of chatbot development tools, describes the process of API configuration and necessary libraries. The practical result of this study is a chatbot "Bestiarij" in the social network "VK".

Key words: chatbot, social network, bestiary, Python, API, library, database, testing

Чат-бот – автоматизированная система общения с пользователем, имитирующая поведение человека. Другими словами – это робот, отвечающий на сообщения человека по внутреннему алгоритму.Такие роботы за последние несколько лет достаточно плотно вошли в обиход человека и используются повсеместно. Сейчас любой сайт, социальная сеть или приложение используют автоматизацию в сфере коммуникации с клиентами, что часто включает наличие чат-бота. Основные функции, которые выполняют чат-боты:

– выполнение стандартных задач в интернет-магазине: собрать заказ, напомнить об акции, обеспечить рассылку, подтвердить доставку, оплату и прочее;

– развлекательная функция: рассказать анекдот, подобрать изображение, фильм, музыкальный плейлист, найти собеседника;

– замена администратора в сообществах, ответ на стандартные вопросы пользователя, например, в чатах поддержки на сайтах банков.

Если обобщить все функции, то, в целом, бот заменяет человека в каких-либо элементарных задачах, обеспечивая высокую скорость получения ответа пользователем, освобождение человека от некоторых задач и направление его на уже определенные программой задачи.

Поначалу считалось, что боты в социальных сетях не могут выполнять весь перечень необходимых задач и потому они служили в основном для небольшого количества ответов и предоставления какого-либо развлекательного контента, но сейчас их возможности расширись настолько, что внутри бота можно разместить товары с возможностью мгновенной оплаты с последующей рассылкой информации о заказе администраторам.

После рассмотрения этих моментов, становится очевидным, почему чат-боты настолько популярны и востребованы в данное время. Люди стремятся снять с себя рутинную работу и довести ее до автоматического выполнения.

Целью исследования является разработка чат-бота «Бестиарий» в социальной сети ВКонтакте. Для достижения поставленной цели необходимо выполнить следующие задачи: изучить методы создания чат-ботов в социальной сети ВКонтакте; выбрать программные средства разработки; провести сбор информации о мифических существах; разработать чат-бот, провести его тестирование и отладку.

Работа выполненав раках учебной практики по получению первичных профессиональных умений и навыков при реализации условий, направленных на получение компетенций в сфере саморазвития и самообразования, опыта профессиональной деятельности в достижении результата – программы, пригодной для практического применения [9].

В ВКонтактепредлагает большой набор функций для выполнения различных задач, в том числе для создания ботов на различных языках программирования. Как и любой ресурс, социальная сеть ВКонтакте имеет свои особенности: бот общается с пользователем через сообщения сообщества. Необходимые функции для создания ботов находятся в открытом доступе и описаны в отдельном разделе ВКонтакте [4].

Для того чтобы написанная программа имела доступ к каким-либо разделам сообщества, необходимо создать ключ с указанием уровней доступа. Этот ключ является конфиденциальным, его утечка может повлечь за собой потерю контроля над сообществом и его кражу.

Социальная сеть ВКонтакте предоставляет широкий выбор языков, на которых можно написать бота. Это может быть Java, Python, C++ и другие. Выберем для создания бота язык Python, так какна нем представлено достаточное количество примеров созданных чат-ботов, что позволило в достаточной мере рассмотреть варианты написания кода и выбор загружаемой библиотеки.

Большинство функций чат-бота имеют реализацию в интернете и доступны в качестве внешнего API [8]. Для связи кода бота и социальной сети будет использоваться VK API. API ВКонтакте позволяет получать информацию из базы данных vk.com с помощью http-запросов к специальному серверу.

Для Python самыми популярными являются библиотеки vk и vk_api. Объективное их сравнение получается довольно проблематичным, так как окончательный комфорт их использования зависит от запросов и привычек программиста.

Основные минусы библиотеки vk:малое количество документации; имеющаяся документация предоставлена на английском языке. Это усложняет работу с библиотекой и требует большего времени для ее досконального изучения и свободного пользования.

В библиотеке vk_api представлена более широкая русскоязычная документация, что дает ей преимущество перед предыдущим вариантом [3, 5]. Таким образом, для написания бота была выбрана библиотека vk_api.

Для подключения бота и его работы требуется создать сообщество и«позволить» сообществу подключать ботов. Для этого нужно перейти в настройки в правой части страницы сообщества и выбрать их (рисунок 1).

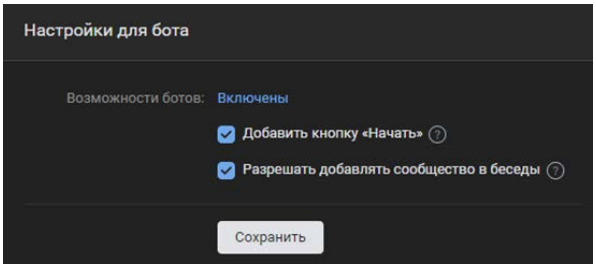

Рис. 1 – Скриншот настроек бота (выполнено автором)

Далее необходимо вернуться в настройки и перейти в раздел «Работа с API». Для того чтобы появилась возможность связать чат-бота с сетью, требуется включить LongPoll API.LongPoll API позволяет работать со всеми сообщениями в режиме реального времени. Это обеспечит максимально быстрый ответ на команды пользователя.

Для работы со всеми методами API необходимо передавать в запросе access_token– специальный ключ доступа. Он представляет собой строку из латинских букв и цифр и может соответствовать отдельному пользователю, сообществу или самому приложению.

Такой ключ позволяет работать с API от имени группы или публичной страницы. Например, с его помощью можно отвечать подписчикам сообщества на сообщения, поступившие в его адрес.

При создании ключа указывается, к каким разделам сообщества будет разрешен доступ (рисунок 2).

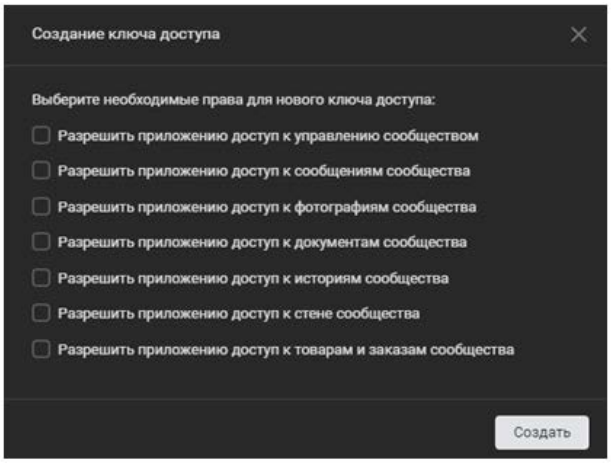

Рис. 2 – Скриншот списка разрешений (выполнено автором)

Для информационного бота достаточно разрешить доступ к сообщениям, фотографиям и документам сообщества. Для более многофункциональных чат-ботов или приложений можно будет создать новый ключ.Стоит обратить внимание, что изменить уже созданный ключ нельзя. Поэтому, если возникнет необходимость расширить или уменьшить доступ, старый ключ придется заменить.При этом большим преимуществом создания нескольких разных

ключей является возможность привязать к сообществу несколько разных ботов или приложений, которые будут стабильно функционировать параллельно друг с другом.

Для дальнейшего использования ключ требуется сохранить, так как для доступа к нему каждый раз будет требоваться подтверждение.

На этом моменте основная подготовка заканчивается, открывается возможность использовать программу для работы с ботом. Можно считать, что дальнейшая работа с социальной сетью напрямую будет осуществляться только при тестировании, а после, при использовании бота.

Бестиарий – средневековый сборник зоологических статей (с иллюстрациями), в которых подробно описывались различные животные в прозе и стихах, главным образом, с аллегорическими и нравоучительными целями. В современном употреблении термина, подразумевается любо ресурс (книга или сайт), в котором собрана информация о различных мифических существах с подробным их описанием и изображениями[2].

Бестиарии часто встречаются в различных играх. Там их пополняет сам игрок, находя в игровом мире новых животных и существ.

Проектируемый чат-бот представляет собой бестиарий мифических лошадей, разделенных на три классификации: обитающих в воздушной среде, водной и на земле.

Для проектирования масштабного бестиария, содержащего достаточно большое количество существ, несомненно, потребовалась бы база данных, но для бота небольших размеров существует другой вариант. Это вложение информации в самого бота или создание библиотек с данными.В чем кардинально отличие этих способов, и какой из них будет считаться наиболее верным?

Если вкладывать большие объемы текста и ссылки на изображения сразу в основной код, могут возникнуть проблемы с более долгой обработкой команд и увеличением времени ответа накоманды.Такая проблема обусловлена тем, что для создания структуры ответов бота используется условный оператор и

для воспроизведения всех возможных команд потребуется множество блоков elif, которые значительно утяжелят программу.Помимо этого, такой способ усложнит поиск ошибок и целостную читабельность кода.

Создание же отдельного файла, содержащего информацию о существах и их изображения, позволяет четко отделить друг от друга код и данные. Любую часть при этом можно будет изменить, не затрагивая вторую.Хранение информации в виде библиотеки ускорит обработку запросов, а с точки зрения эстетики код будеткрасивее.Здесь опять же встает выбор либо на размещение всех данных в одном файле, либо на разделение по предполагаемым классификациям.Этот выбор зависит только от программиста и обуславливается удобством, так как относительно данного момента не существует определенных правил.

В нашем случае приоритет был отдан созданию трех отдельных от кода файлов, названных по содержанию библиотек соответственно.

Файлы библиотек имеют разрешение json.

JSON (JavaScriptObjectNotation) – простой формат обмена данными, удобный для чтения и написания как человеком, так и компьютером. Он основан на подмножестве языка программирования JavaScript.

JSON – текстовый формат, полностью независимый от языка реализации. JSON основан на двух структурах данных: коллекция пар ключ/значение и упорядоченный список значений.Это универсальные структуры данных и современные языки программирования поддерживают их [1, 7].

В языке Python эта концепция реализована как словарь [6].Максимально комфортная для использования и читабельная структура словаря представлена на рисунке 3.

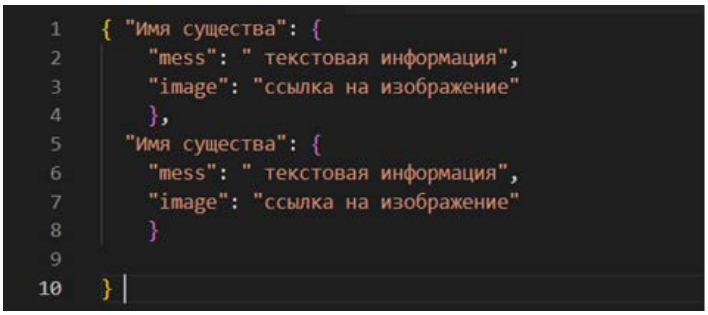

Рис. 3 – Скриншот структуры словаря (выполнено автором)

Первым в нем идет наименование существа, далее представляется его текстовое описание в значении ключа «mes», что буквально означает «сообщение». В значении ключа «image» лежит ссылка на изображение.

На этом этапе определились со способом хранения и использования информации самого бестиария, получена и закреплена опытным путем информация об отдельных файлах, хранящих какую-либо программную единицу.

После завершения библиотеки полностью готовы импортированию и последующему использованию.

Метод в объектно-ориентированном программировании – это функция или процедура, принадлежащая какому-то классу или объекту. Как и процедура в процедурном программировании, метод состоит из некоторого количества операторов для выполнения какого-то действия и имеет набор входных аргументов (параметров).

Для отправки сообщений от бота используется метод messages.send, включающий в себя следующие параметры:

1. user_id– идентификатор пользователя, которому отправляется сообщение.

2. random_id– уникальный (в привязке к API_ID и ID отправителя) идентификатор, предназначенный для предотвращения повторной отправки одинакового сообщения. Сохраняется вместе с сообщением и доступен в истории сообщений. Переданный в запросе random_id используется для проверки уникальности, проверяя в заданном диалоге сообщения за последний час (но не более 100 последнихсообщений).

3. message– Текст личного сообщения. Обязательный параметр, если не задан параметр attachment.

4. attachment–медиавложения к личному сообщению, перечисленные через запятую. Каждое прикрепление представлено вформате:

<type><owner_id>_<media_id>.

Где<type>–тип медиавложения: photo – фотография;video–видеозапись;audio–аудиозапись;doc–документ;wall– записьнастене; market –товар; poll –опрос. <owner_id>– идентификатор владельца медиавложения (если объект находится в сообществе, этот параметр должен быть отрицательным). <media_id>– идентификатор медиавложения. Пример: photo-212647391_457239036.

У метода есть множество других параметров, например lat и long, в которых указывается широта и долгота, или sticker_id, описывающий идентификатор стикера. Но для разрабатываемого бота достаточно тех, что были описаны выше.

Метод messages.send – основной для чат-бота, так как коммуникация с пользователем осуществляется только через сообщения.

Следует отдельно уделить внимание параметру attachment, отвечающему за медиавложения. Конечно, хранить изображения можно где-то на компьютере, но этот параметр позволяет значительно упростить и ускорить загрузку изображения всообщение.

Для использования прикрепления в соответствующем параметру виде, необходимо загрузить необходимые изображения в любой удобный альбом (для программируемого бота изображения были загружены в основной альбом сообщества), откуда можно взять ссылку для прикрепления, например из адресной строки. Так можно использовать любые медиафайлы, находящиеся в социальной сети. Этот этап позволил конкретно отделить от других понятие метода ивозможность его использования, из большинства возможных аргументов выделены необходимые в предстоящем написании кода, сформирована структура медиавложения. На данном этапе все приготовления к написанию самого кода окончены.Перед написанием кода для бота необходимо

установить модуль. Для этого выполняется переход в командную строку, где вводится команда pipinstallvk_api. Если установка модуля прошла успешно, увидим надпись, представленную на рисунке 4.

Рис. 4 – Результат установки модуля vk_api (выполнено автором)

Теперь подготовку к написанию и запуску бота можно считать законченной.

Для того чтобы наши команды были понятны программе, необходимо импортировать библиотеку и подключить модуль json, для получения возможности работать с этими файлами, а также вытащить из импортированной библиотеки longpoll (рисунок 5).

```
import json
import vk_api
from vk_api.longpoll import VkLongPoll, VkEventType
```

Рис. 5 – Импортирование библиотеки и модулей (выполнено автором)

В Python ключевое слово import применяется для того, чтобы сделать код в одном модуле доступным для работы в другом. Импорт в Python важен для эффективного структурирования кода. Импортируем в программу необходимые для нас модули и делаем их доступными для дальнейшего использования.

Теперь нужно обучить программу читать файлы с библиотеками существ, для этого используется команда write, в параметрах которой нужно указать на файл, режим доступа, а также кодировку.

Режим доступа – часть, в которой мы указываем для чего открывается файл: для чтения, записи, добавления информации, и т.д. Например, "w". По умолчанию файл открывается для чтения – "r".

Режимов доступа существует множество и каждый из них выполняет своюфункцию:

1. r – открывает файл только длячтения;

2. rb– открывает файл для чтения в двоичномформате;
3. r+ – открывает файл для чтения изаписи;
4. rb+ – открывает файл для чтения и записи в двоичномформате;
5. w – открывает файл только для записи, создает файл с именем имя_файла, если такового не существует;
6. wb– открывает файл для записи в двоичном формате, создает файл с именем имя_файла, если такового не существует;
7. w+ – открывает файл для чтения и записи, создает файл с именем имя_файла, если такового не существует;
8. wb+ – открывает файл для чтения и записи в двоичном формате, создает файл с именем имя_файла, если такового не существует;
9. a – открывает файл для добавления информации в файл, создает файл с именем имя_файла, если такового не существует;
10. ab– открывает файл для добавления в двоичном формате, создает файл с именем имя_файла, если такового не существует;
11. a+ – открывает файл для добавления и чтения, создает файл с именем имя_файла, если такового не существует;
12. ab+ – открывает файл для добавления и чтения в двоичном формате, создает файл с именем имя_файла, если такового не существует.

Для проектируемого бота достаточно открыть файл только для чтения. Все, что получается в результате работы библиотеки, передаем в какую-нибудь переменную, здесь переменные названы по именам открываемых файлов.

В проектируемом боте используем кодировку utf-8, чтобы программа могла работать с кириллицей. Код операции представлен на рисунке 6.

```
with open('terra.json', 'r', encoding='utf-8') as file:
    terra = json.load(file)

with open('water.json', 'r', encoding='utf-8') as file:
    water = json.load(file)

with open('air.json', 'r', encoding='utf-8') as file:
    air = json.load(file)
```

Рис. 6 – Чтение библиотек, присваивание результата переменной (выполнено автором)

Чтобы при каждом использовании ключа доступа не дублировать весь еготекст, токен присваивается одноименной переменной token. Уже через переменную выполняется ее подключение, здесь же подключается longpool (рисунок 7).

```
token = "ключ доступа к сообществу"
bh = vk_api.VkApi(token=token)
longpoll = VkLongPoll(bh)
```

Рис. 7 – Подключение ключа доступа и longpool (выполнено автором)

Для того чтобы не использовать большие объемы кода при прописывании команд для бота использованы функции.

Функция – именованный блок кода, который вызывается в нужных местах программы по имени. Другими словами, функция представляет собойподпрограмму, которую можно вызвать из основной программы, причем неоднократно. Повторяющийся блок программного кода обычно обозначают некоторым уникальным именем, чтобы потом при необходимости обратиться к нему по этому имени.

В языке программирования Python функции определяются с помощью оператора def.

Первая функция send_message() будет использоваться для ответа на сообщения сообщества, она получает id пользователя (user_id), которому оно отправит сообщение и собственно само сообщение и какой-либо файл. Значение для аргумента файла по умолчанию указано None, что позволяет

использовать данную функцию даже тогда, когда к сообщению не предполагается прикреплять медиафайл, программа не выдаст ошибки.

Python NONE – это объект идентичный значению NULL в других языках программирования, таких как java или php. Объект NONE относится к типу данных «NoneType» и, следовательно, не может рассматриваться как значение некоторых примитивных типов данных или логических значений.Можно считать, что переменной присваивается нулевоезначение,«ничего».

Вторая функция get_creatures() создается с целью формирования строки из всех ключей (наименований существ) указанного словаря, где каждый ключ начинается с большой буквы и отделен от другого символом новойстроки.Функции send_message и get_creaturesпредставлены на рисунке 8.

Рис. 8 – Функцииsend_messageиget_creatures (выполнено автором)

Чтобы бот умел «слушать» сообщения, используем цикл, в котором по кругу проверяем наличие новых событий. Внутрь цикла вкладываем условный оператор, выполняющийся, если новым событием сообщества является сообщение.

Второй условный оператор проверяет, имеет ли сообщениеметкудля бота. Если его не использовать, бот может отвечать насобственныесообщения. Далее полученное сообщение и id пользователя присваивается переменным, и работа продолжается уже с ними. Причем указываем для бота чтение сообщений с маленькой буквы, что обеспечит одинаковый ответ на сообщение как из прописных, так и из строчных букв. Этот блок кода представлен на рисунке 9.

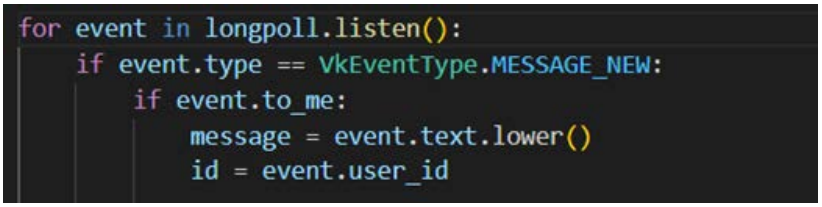

Рис. 9 – Основной цикл (выполнено автором)

Остальной код программы состоит из оператора if – elif – else, в котором прописаны стандартные команды для бота и соответствующие им ответы. Соответственно частей с elif в такой конструкции может быть сколько угодно много, но для уменьшения их количества и упрощения кода использовались отдельные файлы с библиотеками.

Предполагая дельнейшее увеличение объемов информации, в чат-бот добавили команду, используя которую, пользователь может отправить администрации любое сообщение о некорректности предоставленных данных или уведомить об ошибке.Часть этого блока на рисунке 10.

```
if message == 'начать':
    send_message(id, 'Приветствую! Я могу рассказать тебе о многих мифических лошадях!\n'
    'Если хочешь увидеть список водяных лошадок, напиши "вода".\n'
    'Если же тех, кто передвигается по воздуху, напиши "воздух".\n'
    'Чтобы посмотреть наземных существ, напиши "земля".\n'
    'Если ты знаешь информацию, которой недостает в описаниях осуществах, напиши команду "Добавить данные"')

...

elif message == 'земля':
    send_message(id, 'Если захочешь узнать о ком-то из списка поподобнее, просто напиши его имя. '
    'Извини, но я еще не умею исправлять опечатки, поэтому постарайся не ошибиться ^^ \n'
    + get_creatures(terra))

elif message in air.keys():
    send_message(id, air[message]['mes'], air[message]['image'])

...

else:
    send_message(id, 'Я не знаю такой команды. Чтобы вернуться в начало, напиши мне "начать".'
    'Если ты отправлял дополнительную информацию, она будет передана администратору и добавлена при подтверждени подлинности.')
```

Рис. 10 – Блок команд и ответов (выполнено автором)

К концу этого пункта бот полностью сформирован и готов к тестированию, загружены и приведены в необходимый вид библиотеки, написаны блоки команд, необходимых для полноценной работы программы. Следующим и завершающим этапом является тестирование.

Для проверки всех функций чат-бота запускается программа и дальнейшие наблюдения производятся через сообщения сообщества. При возникновении ошибки программа отправит уведомление, а пользователь не получит ответа на сообщение.Так на этапе тестирования выявляются основные ошибки и неполадки в коде.

При нажатии кнопки «начать» или ручном написании, в ответ приходит приветственное сообщение с перечнем доступных команд. На команды бот так

же выдает прописанные в коде ответы. На рисунках 11–14 представлены скриншоты выполнения команд чат-ботом (примеры автора).

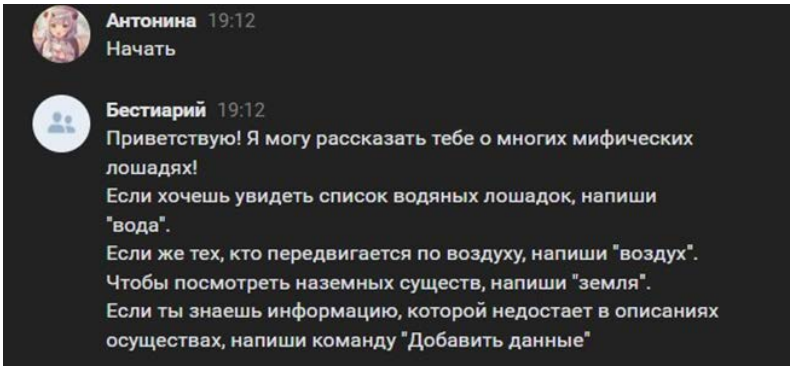

Рис. 11 – Скриншот приветственного сообщения чат-бота

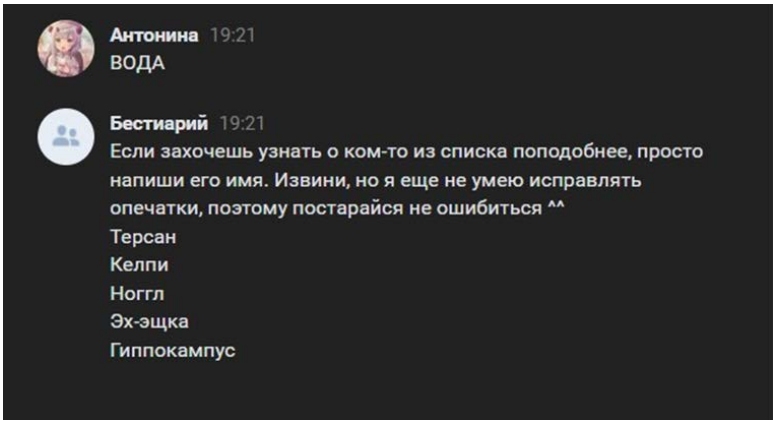

Рис. 12 – Скриншот выдачи списка водных существ в чат-боте

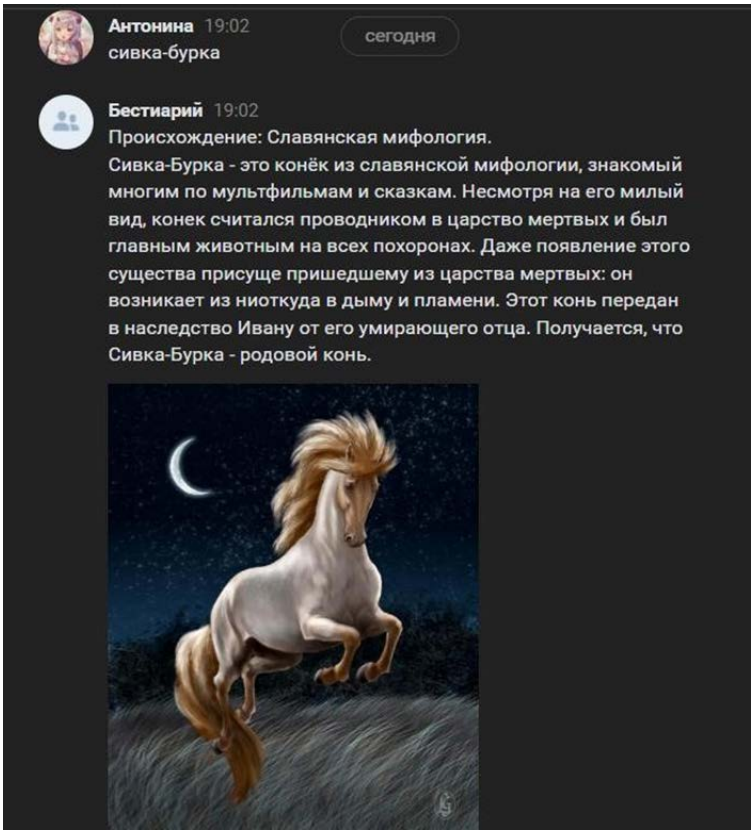

Рис. 13 – Ответ на команду «Сивка-Бурка» (выполнено автором)

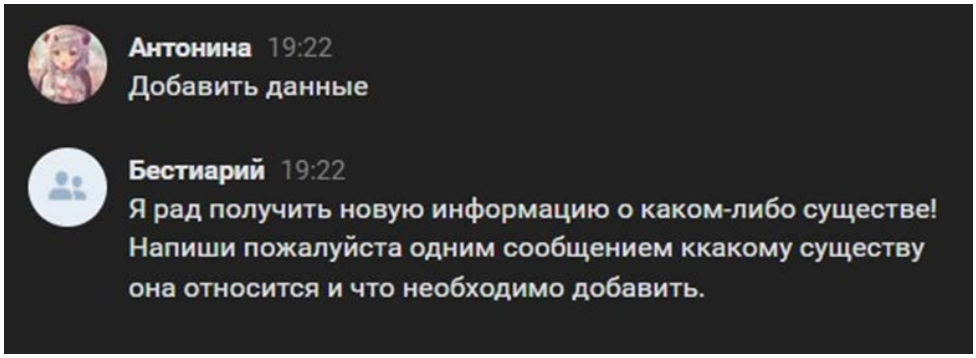

Рис. 14 – Команда на добавление данных

При выдаче перечня наименований существ бот использует функцию get_creatures(), которая позволила не прописывать имя существа вручную.

Если пользователь напишет сообщение, на которое при проверке в цикле не будет соответствовать ни одной стандартной команде, бот выдаст сообщение из части else (рисунок 15).

Даже если у пользователя будет нажата клавиша CapsLock, чат-бот считает его сообщение без ошибки и даст соответствующий прописанным командам ответ.

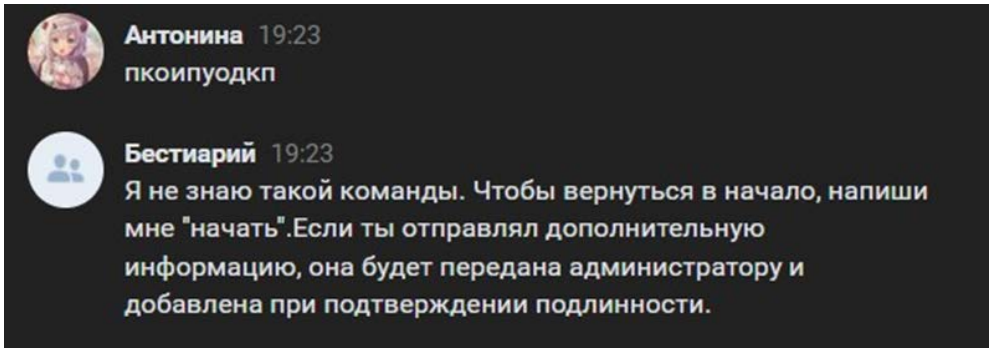

Рис. 15 – Реакция на бессвязный текст (выполнено автором)

Таким образом, бот успешно прошел тестирование. Команды прошли проверку и работают стабильно, что свидетельствует об отсутствии ошибок в коде библиотек и основном коде программы.

В процессе разработки чат-бота проведен поиск, сравнение, структурирование информации, а также сбор данных, на основе которых далее разрабатывалась программа, создавались дополнительные файлы библиотек, изучены дополнительные функции, методы и библиотеки языка Python, что

послужило получению важного для будущей профессиональной деятельностиопыта.

Библиографический список:

1. JSON в Python 3 // Образовательный портал Python 3. – URL: https://python-scripts.com/json (дата обращения:18.05.2022)

2. Бестиарий мифических существ // Русскоязычный информационно-развлекательный форум picabu.– URL: https://pikabu.ru/story/bestiariy_mificheskikh_sushchestv_chast_ii_loshadi_4769258 (дата обращения: 18.05.2022)

3. Библиотека vk для работы с VK API на Python // Система тематических блогов «Хабр». – URL: https://habr.com/ru/post/319178/ (дата обращения: 18.05.2022)

4. ВКонтакте // Социальная сеть. – URL: https://vk.com (дата обращения: 14.09.2022)

5. Документация vk_api // Документация vk_api. – URL: https://vk-api.readthedocs.io/en/docs/ (дата обращения:18.05.2022)

6. Обучение Python // Образовательный портал Pythonicway. – URL: http://pythonicway.com/ (дата обращения:18.05.2022)

7. Файлы JSON // Алексей Лавриненко | OleksiyLavrynenko – PR, SMM, WEB. – URL: https://lavrynenko.com/kak-chitat-json-fajl-v-python/ (дата обращения: 18.05.2022)

8. Шовин В.А. Программа chatbot - чат-бот или виртуальный собеседник // МСиМ. 2016. №4 (40). URL: https://cyberleninka.ru/article/n/programma-shatbot-chat-bot-ili-virtualnyy-sobesednik (дата обращения: 20.10.2022).

9. Янченко, И. В. Формирование карьерной компетентности студентов в профессиональном образовании : диссертация ... кандидата педагогических наук : 13.00.08 / Янченко Инна Валериевна; [Место защиты: Краснояр. гос. пед. ун-т им. В.П. Астафьева]. - Красноярск, 2013. – 255 с.

Оригинальность 75%