ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ
ФГАОУ ВО НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

Факультет компьютерных наук
Образовательная программа «Прикладная математика и информатика»

Отчет о программном проекте

на тему Автоматизация модерирования телеграм-канала

Выполнила:
студентка группы БПМИ216 А.И.Ульянова
И.О. Фамилия Подпись

23.05.2023
Дата

Принял:
руководитель проекта Терлыч Никита Андреевич
Имя, Отчество, Фамилия
Штатный преподаватель
Должность
НИУ ВШЭ, Базовая кафедра фирмы 1С
Место работы

Дата 23.05.2023 10 _____
Оценка (по 10-тибалльной шкале) Подпись

Москва 2023

Аннотация

Цель моего проекта - создание телеграм бота для автоматизации модерирования телеграм-канала. Бот должен самостоятельно парсить информацию из источников, выбранных пользователем, согласовывать полученный контент с пользователем, после чего публиковать его в телеграм канале.

В процессе работы была изучена библиотека aiogram для написания ботов на Python, API для работы с различными платформами, такими как YouTube, ВКонтакте, telegram, особенности работы с базой данных SQLite3 и несколько библиотек для обработки естественного языка (NLP).

Результатом работы является директория, содержащая исходный код программы, а также телеграм-бот, работающий благодаря размещению кода на хостинге.

Содержание

Основные термины и определения

В настоящем отчете применяют следующие термины с соответствующими определениями:

Бот – это программа, выполняющая автоматические заранее настроенные повторяющиеся задачи.

Контент – это любая информация, выраженная речью, текстом и любыми другими способами передачи.

Мем – созданная в комичных целях текстовая или визуальная информация, распространяющаяся в интернете.

Парсинг – автоматизированный сбор и систематизация данных.

Репост – вторичная публикация сообщения, размещенного другим пользователем в социальной сети или блоге, со ссылкой на источник.

Введение

В настоящее время в социальных сетях растет популярность мемов: возникла определённая мода на их распространение, создание, тиражирование среди интернет-пользователей. Искать оригинальный контент самому становится практически невозможно из-за его бесконечного многообразия, поэтому особым спросом пользуются каналы, которые репостят контент со многих других площадок и каналов, чтобы пользователи могли зайти в одно место и насладиться всем многообразием шуток. Однако и в таких каналах часть контента окажется отмодерирована, часть отсеется, часть будет неинтересна. Это делает ленту пользователя не релевантной.

Цель проекта - создание своего телеграм-бота, который будет в автоматическом режиме собирать контент из наиболее интересных источников и предлагать лучшее на выбор. Таким образом можно автоматизировать процесс сбора контента и отсеивания лучшего, а заодно делиться избранным со своей аудиторией в личном телеграм-канале.

Задачи проекта:

- Изучить уже имеющиеся аналоги
- Спроектировать и разработать подсистемы сбора данных со сторонних сайтов и модерации контента
- Научить бота строить предположение относительно тематики/настроения собранного контента
- Настроить корректное взаимодействие между ботом и личным телеграм-каналом

1. Обзор и сравнительный анализ источников и аналогов

Среди многочисленных ботов в telegram уже существуют готовые решения для модерации личного telegram-канала. К одним из самых популярных можно отнести ControllerBot и TelepostBot. Их функционал по большой части совпадает и направлен на форматирование и изменение контента, вводимого пользователем собственноручно. Например, эти боты позволяют создать пост, отформатировать его, задать время отправки и время удаления, добавить к посту кнопки с различным функционалом, фотографию или ссылку. Также с помощью данных ботов можно работать с уже опубликованными постами или смотреть статистику канала. Описание функционала ControllerBot от разработчиков представлено ниже на рисунке 1.

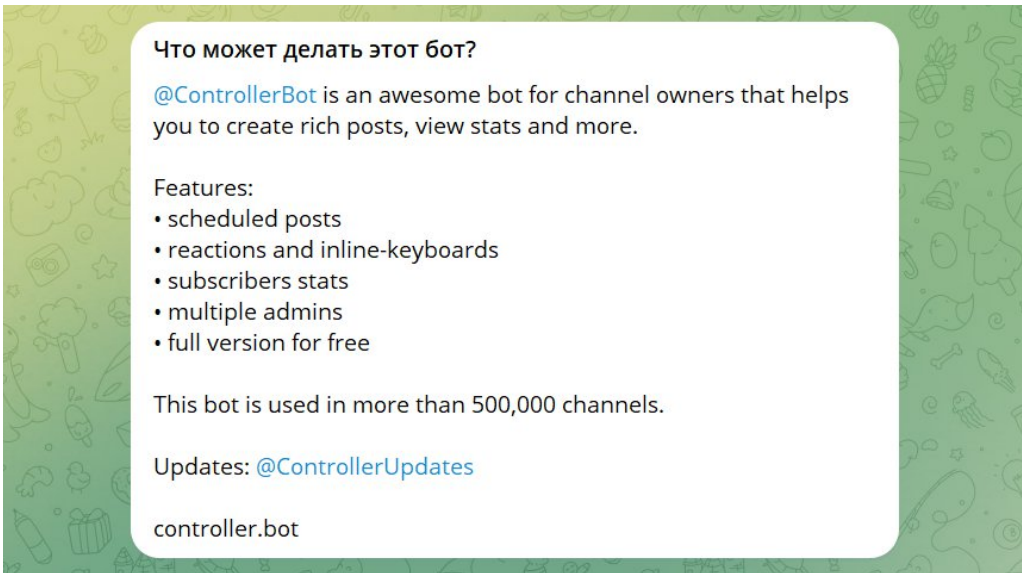

Рисунок 1 - Функционал ControllerBot

Стоит также обратить внимание на некоторое неудобство модерации канала через ControllerBot. Потребуется предварительно создать отдельного бота-посредника, после чего взаимодействовать именно с ним, а не с самим ControllerBot. Это несколько запутано и может быть нетривиальной задачей для пользователей, имеющих небольшой опыт работы в телеграме. Подробнее алгоритм создания бота-посредника представлен ниже на рисунке -2.

Рисунок 2 – Пример добавления модерируемого канала в ControllerBot

Еще один популярный бот для модерации канала – FleepBot. Это бот-сервис, который на выбор позволяет создавать ботов для отложенного постинга или для обратной связи (общения через бота). Выбрав вариант для постинга становится возможным откладывать сообщения, редактировать отправленные, добавлять к постам кнопки, реакции, закреп.

Ботов подобного функционала огромное количество, каждый из них имеет лишь свои небольшие отличительные черты. Более подробно вместе со всеми особенностями и тонкостями эти боты и программы описаны в статье Петра Ивлева “16 сервисов по ведению чатов и каналов в Telegram” на сайте интернет-издания vc.ru [8].

Есть также категория ботов, занимающаяся парсингом telegram-каналов, чатов и их участников, например, TgParsingBot, parsetgbot. Они могут найти все каналы, содержащие в названии заданное ключевое слово, составить базу данных из всех участников чата, или посмотреть на последние n сообщений в чате и отобрать некоторые из них по заданному пользователем критерию. Однако их функционал достаточно ограничен и распространяется исключительно на данный мессенджер. Ниже на рисунке 3 представлен функционал работы parsertgbot.

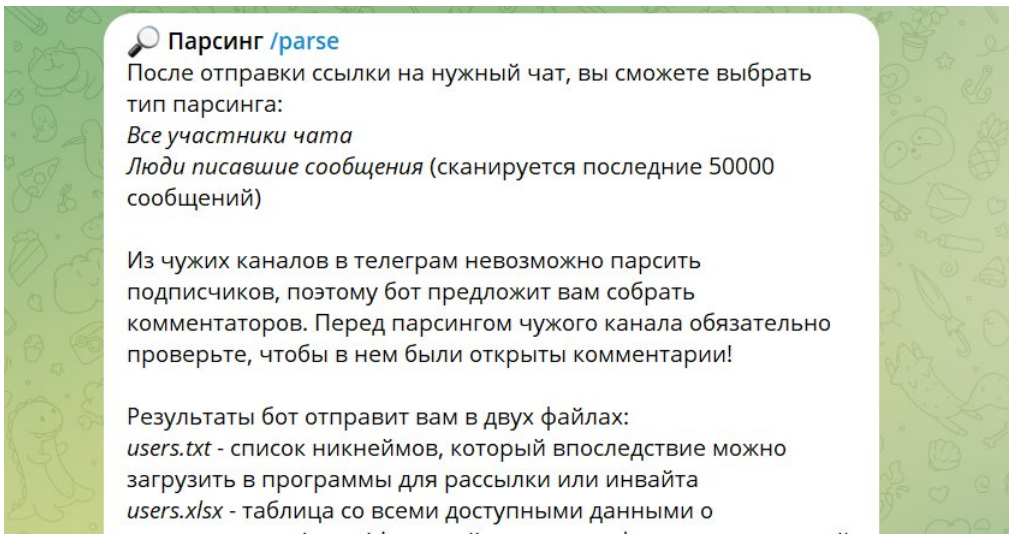

Рисунок 3 – Описание функционала бота parsertgbot

Аналогичен им также бот Telecobra4bot. Он способен спарсить паблики, пользователей или сообщения и сохранить их в виде файлов .txt и .csv. Ниже на рисунке 4 представлена его работа на примере парсинга пользователей одного из чатов в телеграме.

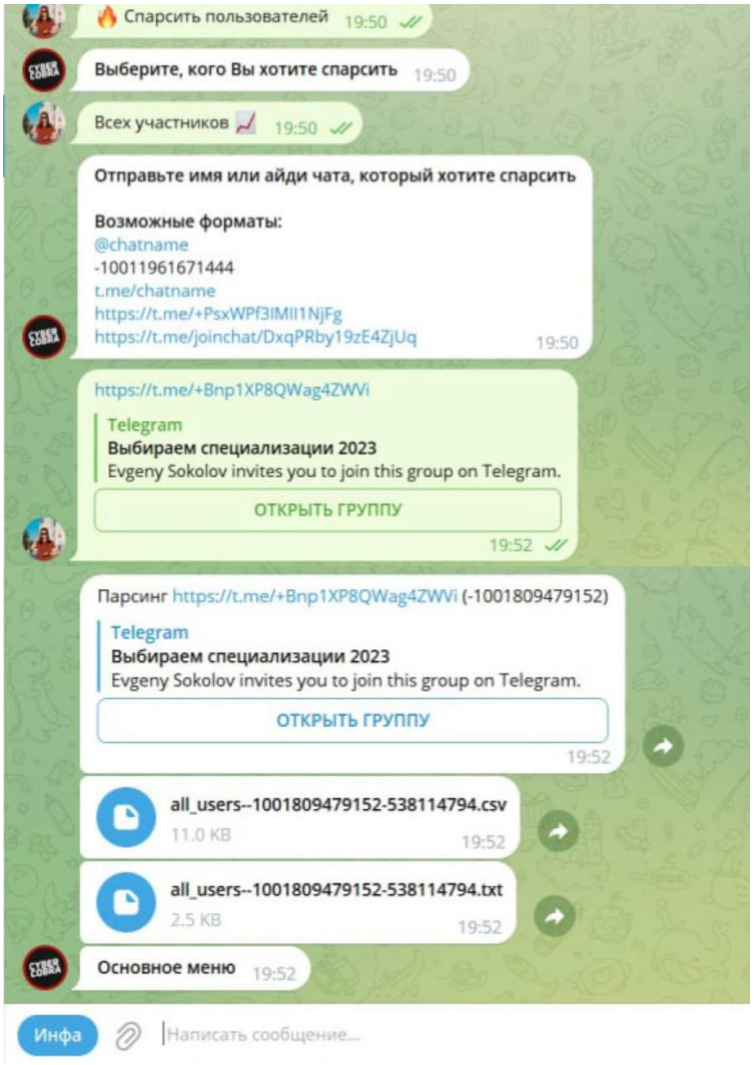

Рисунок 4 – пример работы бота Telecobra4bot

Существуют так же telegram-боты, направленные на парсинг сторонних ресурсов: различных маркетплейсов, новостных и тематических сайтов. Примерами таких ботов являются WBCON_PARSER_BOT, MarketplaceParserBot, huntersalesbot и другие. Данные боты предлагают пользователю следующие возможности: просмотр аналитики по карточке товара, в том числе среднемесячный оборот, оборот за последний месяц, изменения цены, поиск товара по артикулу, просмотр остатков по складу и по размеру, поиск скидок, выгодных предложений и так далее [1, 2].

Например, WBCON_PARSER_BOT позволяет получить полную информацию о товаре с одного из сайтов, представленные на рисунке 5. Пользователь вводит артикул товара, после чего получает данные о товаре: изготовителе, цвете, комплектации и т.п. Более того, бот предоставляет информацию о поставщике товара (его ИНН, адрес, рейтинг) и об остатке товара на том или ином складе.

Рисунок 5 – список сайтов доступных для WBCON_PARSER_BOT

Из всего вышесказанного можно сделать вывод, что, telegram полон ботов, занимающихся модерированием telegram-каналов или парсингом, но не обоими этими задачами одновременно. Таким образом, не один из них в точности не выполняет тот функционал, реализация которого является задачей данного проекта.

2. Теоретическая часть

2.1 Описание сценариев использования

Ниже на рисунке 1 приведена диаграмма вариантов использования, описывающая, какой функционал разрабатываемой программной системы доступен каждому актору. В данном случае акторами будут являться сам пользователь и бот.

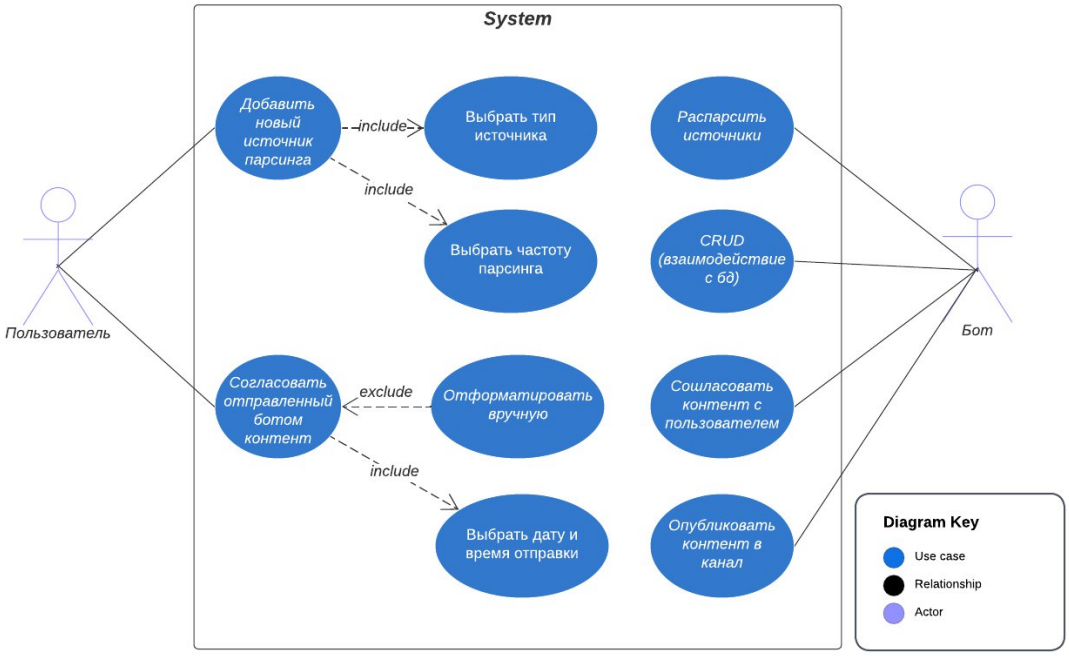

Рисунок 1 - Диаграмма вариантов использования

Ниже описан один из вариантов использования программы – публикация в канал из нового источника. В Таблице 1 представлены участники данного сценарияuse-case-a, предусловия, которые должны быть выполнены для корректного выполнения программы, а также постусловия, которые станут результатом работы кода.

2.2 Use-case “Публикация в канал из нового источника”

Таблица 1 - Предусловия и постусловия сценарияuse-case-a

Участники	Базовый пользователь (далее Пользователь)
Предусловия	Пользователь зашел в telegram. У пользователя есть телеграм-канал, бот является в нем админом со всеми правами.
Постусловия	В канале пользователя появляется публикация из нового источника

Основной поток

1. Пользователь использует команду `/set_channel`, чтобы установить канал для модерации
2. Пользователь пересылает боту сообщение из канала, который хотел бы модерировать
3. Пользователь использует команду `/add_new_source` чтобы добавить новый источник
4. Бот отображает 3 кнопки с выбором типа источника: телеграм-канал/чат, сообщество ВКвконтакте, YуouTtube канал
5. Пользователь выбирает нужный тип источника и отправляет ссылку на него
6. Пользователь использует команду `/parse`
    a. Бот очищает данные от рекламы
    b. Бот форматирует данные для красивого отображения в telegram
    c. Бот добавляет кнопки, с помощью которых пользователь сможет редактировать контент
    d. Бот отправляет контент пользователю
7. Пользователь просматривает полученный контент
    a. Пользователь нажимает на кнопку “Опубликовать это в канал” под понравившимся постом
    b. Пользователь удаляет все не понравившиеся посты
8. Бот выкладывает пост в телеграм-канал

Альтернативные потоки

1. Ссылка на источник, отправленная пользователем, некорректна
    a. Бот выводит сообщение об ошибке
    b. Переход к шагу 3 основного потока
2. Бот не обнаружил нового контента во время парсинга
    a. Бот отправляет пользователю сообщение о том, что парсинг был произведен, но нового контента не появилось
    b. Переход к шагу 6 основного потока

2.3 Описание требований к программному продукту

1.Функциональные требования

Бот должен обладать следующим набором функций:

1. Парсинг источников следующих типов: телеграм-каналов, телеграм-чатов, групп ВКонтакте~~вконтакте~~, YouTube~~youtube~~-каналов
2. Парсинг текстовых данных и изображений
3. Запуск парсинга источников по запросу пользователя
4. Очистка полученных данных от мусора и их сохранение
5. Форматирование данных для красивого отображения в telegram
6. Классификации контента по категориям и настроению
7. Отправка контента пользователю для согласования его публикации в канал
8. Публикация контента в канал

Также бот должен быть расширяемым, т.е. иметь возможность добавления новых источников контента

2. Нефункциональные требования

Бот должен:

1. Корректно работать со всеми версиями telegram начиная с v 4.0
2. Корректно работать в случае отсутствия нового контента в выбранных источниках
3. Иметь удобный для пользователя интерфейс
4. Быть написанным на языке Python с использованием библиотеки aiogram

2.4 Обоснование выбора технологий и средств разработки

Ботов для мессенджера telegram можно написать на многих языках программирования, например, на Python, JavaScript, Go и других [7]. Однако именно Python был выбран мной для реализации этого проекта по многим причинам. Во-первых, Python имеет простой и понятный синтаксис, который делает код проекта более легким для понимания и использования. Во-вторых, Python обладает широким набором уже готовых библиотек и инструментов для парсинга и взаимодействия с API различных приложений, что делает его удобным для разработки ботов. В-третьих, Python поддерживается многими платформами и операционными системами, что обеспечивает универсальность и гибкость для использования [6].

Для парсинга каждого из типов источников я использовала разные технологии.

Для ВКонтакте ~~Вконтакте~~ мной было выбрано API ВКонтакте~~Вконтакте~~, так как оно предоставляет удобный интерфейс для работы с данными ВКонтакте~~Вконтакте~~, включая доступ к постам, комментариям, фото и видео [10]. Более того, это API не накладывает никаких ограничений на количество постов, которое можно получать со страниц сообществ в день, а это та самая функция, которая и была мне нужна.

Что касается YouTube~~Youtube~~, эта платформа активно использует различные механизмы защиты авторских прав и безопасности данных, чтобы предотвратить несанкционированный доступ к своим видео. Из-за этого получить полный доступ ко всей необходимой информации очень сложно. После тщательного, но безуспешного поиска обходных путей и способов получать данные с этого видеохостинга без ограничений, я все же остановилась на использовании YouTube ~~Youtube~~ API. Оно корректно работает и предоставляет доступ к видео, однако имеет ежедневный лимит на количество запросов.

Для получения информации из каналов и чатов телеграм я воспользовалась библиотекой «T~~t~~elethon». Одним из преимуществ использования «Telethon» для получения сообщений из других чатов является удобство и простота использования. Библиотека предоставляет простой и интуитивно понятный интерфейс для работы с Telegram API, что делает процесс получения сообщений быстрым и легким. Она также не накладывает никаких ограничений на число сообщений, которы~~е~~~~ое~~ можно расе~~п~~арсить [3].

Для реализации проекта понадобилось также хранить собранный контент в базе данных. Их выбор очень велик, например, можно использовать такие как SQLite, MySQL и PostgreSQL. Мой выбор пал на базу данных SQLite3. База данных SQLite3 является

легковесной, быстрой и удобной для использования в проектах на Python. Она позволяет хранить и обрабатывать данные локально, что делает ее подходящей для небольших и средних проектов. Одним из главных преимуществ использования SQLite3 в проектах на Python является его интеграция в Python. База данных SQLite3 поставляется с Python в качестве стандартной библиотеки, что позволяет использовать ее без дополнительной установки и настройки. Кроме того, SQLite3 поддерживает большинство функций SQL, что делает ее удобной для работы с различными типами данных и запросов. Кроме того, SQLite3 поддерживает большинство функций SQL, что делает ее удобной для работы с различными типами данных и запросов.

Также небольшая часть проекта была посвящена обработке текста. Стояла задача по тексту поста на русском или английском языке определить его эмоциональную окраску: является ли он позитивным, негативным или скорее нейтральным. В решении этой задачи мне пригодились библиотеки по обработке естественного языка (NLP) уже имеющиеся в Python. Для текста на английском языке мной была выбрана библиотека «TextBlob», так как она очень распространена и к ней есть документация, что делает её использование простым. Для русского же языка я использовала библиотеку «dostoevsky», так как в ней есть нужный для нашей задачи функционал. Более того, её использование очень быстро и удобно [9].

3. Разработка телеграм-бота

В данной главе будет описан процесс разработки бота. Для удобства он был разбит мной на 3 этапа: проектирование, реализация и тестирование. В первой части будет рассказано про структуру проекта: из каких частей он состоит, как устроены классы и база данных. Во второй части будут описаны принципы работы отдельных частей бота, будет подробно представлен алгоритм работы пользователя с ним, а также будет рассказано про мой опыт развёртывания бота. В третей же части будут описаны проведенные мной тесты для определения корректности работы программы.

3.1 Проектирование

Важной частью работы над проектом стало его проектирование. Согласно принципам ООП проект был разбит на классы, каждый из которых выполняет некоторые функции, объединенные общим смыслом. Подробная схема всех классов и имеющихся в них функций представлена ниже на Рис.2.

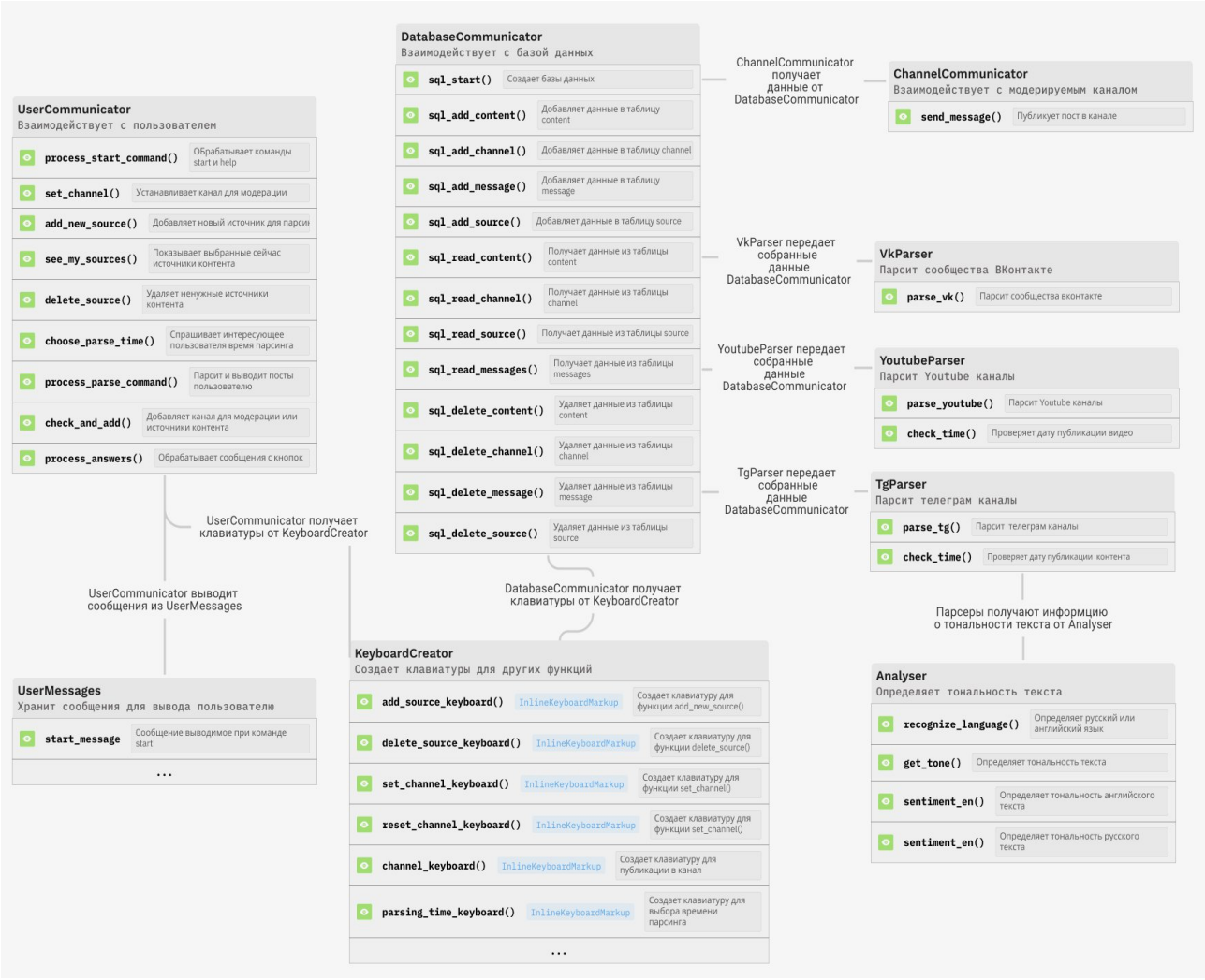

Рисунок 2 - Диаграмма классов

1. UserCommunicator – класс, отвечающий за коммуникацию с пользователем. Он обрабатывает поступающие от пользователя запросы, выводит собранный контент, сообщает пользователю о произошедших ошибках и т.д.
2. DatabaseCommunicator – взаимодействует с базой данных. Выполняет запросы на добавление, чтение и удаление для всех имеющихся таблиц.
3. Channel Communicator – взаимодействует с модерируемым каналом. Публикует в него контент, который согласовал пользователь.
4. VkParser, YoutubeParser и TgParser – классы, занимающиеся парсингом источников соответствующего типа.
5. KeyboardCreator – класс, создающие клавиатуры для более удобного взаимодействия с пользователем.
6. UserMessages – класс, хранящий в себе все сообщения, которые могут понадобиться для взаимодействия с пользователем
7. Analyser – класс, занимающийся классификацией текста постов на положительный, негативный или нейтральный

Каждый из классов расположен в своем файле с соответствующим названием, а файлы в свою очередь объединены по смыслу в папки. В папке handlers находятся UserCommunicator, DatabaseCommunicator и Channel Communicator, в папке parsers – парсеры VkParser, YoutubeParser и TgParser, а в папке utils – все вспомогательные классы KeyboardCreator, UserMessages и Analyser. Помимо классов имеются основные файлы бота: main.py и create_bot.py. Первый является основным файлом программы, который и запускается для начала работы бота. Второй же создает экземпляр бота и вынесен в отдельный файл лишь для избежания циклических импортов.

Стоит также объяснить структуру используемой базы данных. Подробная диаграмма базы данных представлена ниже на рисунке 3:

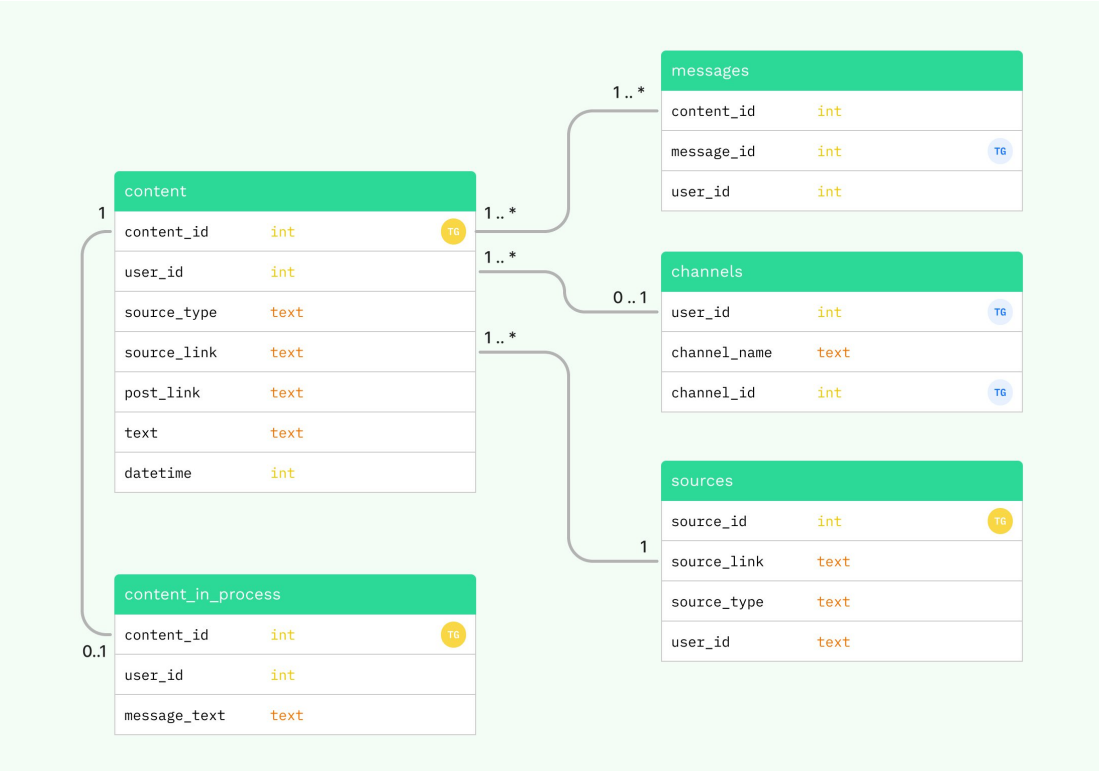

Рисунок 3 - Диаграмма базы данных

База данных состоит из пяти таблиц:

1. Content – таблица, хранящая в себе информацию о собранных из источников постах.
2. Messages – таблица, содержащая данные об сообщениях с контентом, отправленных пользователю на согласование. Необходима для отслеживания связи между единицей контента и действием, которое пользователь хочет с ней произвести.
3. Channels – таблица, сопоставляющая каждому пользователю модерируемый им канал. Допускает наличие у пользователя не более одного канала для модерации.
4. Sources – таблица, хранящая данные о источниках, из которых хочет получать контент каждый из пользователей.
5. Content_in_process – таблица, хранящая сведения о контенте, с которым сейчас работает пользователь. Например, текст какого именно поста пользователь пытается отредактировать. Данная таблица необходима для налаживания связи между таблицей объектами контента и объектами сообщений.

3.2 Реализация

Классы и база данных проекта реализованы в полном соответствии со спроектированным планом. Код проекта находится в гитхаб репозитории по ссылке https://github.com/alexxandra-u/telegram_channel_moderator , а также хостится на сервисе railways.app. Воспользоваться ботом можно найдя его в телеграм по названию tg_channel_moderator или по имени @myy_admin_bot.

3.2.1 Парсинг контента

Про способ, которым получаются данные из каждого типа источника более подробно рассказано в пункте 2.4 данного отчета. Здесь же хотелось бы остановиться на более общих аспектах парсинга.

Данные в источниках рассматриваются, начиная с наиболее новых и заканчивая более старыми. Рассмотрение прекращается, когда время с момента публикации поста начинает превышать время, выбранное пользователем на этапе настройки парсинга. В связи с этим посты сохраняются в базу данных в обратном хронологии порядке и таким же образом позже отправляются пользователю на согласование. Для определения времени во всех частях программы используется единый формат: unix время, т.е. количество секунд, прошедших с полуночи 1 января 1970 года.

На этапе парсинга также происходит отчистка данных от рекламы. Точнее говоря, посты, напоминающие рекламные посты, просто не сохраняются в базу данных и не отправляются пользователю. В приложении ВКонтакте каждый пост имеет параметр `marked_as_ads`, который обозначает, является ли пост рекламным или нет. Соответственно посты, такие что `post[«marked_as_ads»] == 1`, игнорируются. Для telegram и YouTube проверка несколько проще: проверяется наличие или отсутствие в посте слова «реклама». Это связано с тем, что в последнее время многие блогеры и публичные личности, ведущие телеграм каналы, отмечают рекламные тексты специальными хэштегами, например, хэштегом #реклама. Соответственно данная проверка во многих случаях сможет корректно определить рекламный пост.

3.2.2 Анализ текста на эмоциональную окрашенность

Анализ текста на английском языке производится с помощью библиотеки «TextBlob». С помощью пары функций, увидеть которые можно в коде программы, мы получаем два параметра для текста: p_pos и p_neg. Каждый из них варьируется от 0 до 1 и

отвечает за позитивную и негативную эмоциональную окраску текста соответственно. При этом сумма этих параметров не обязательная равна одному. Поэкспериментировав с различными фразами, я пришла к выводу, что оптимальнее всего будет относить к нейтральным фразам те, где параметры p_pos и p_neg отличаются не более чем на 0.1. В остальных же случаях я относила текст к той категории, параметр для который был наибольший.

Текст на русском языке я анализировала с помощью библиотеки «dostoevsky». Она, в отличие от библиотеки «Textblob», может выдавать для текста разные параметры из следующего списка: positive, negative, neutral, skip и speech. Skip означает, что нейронная сеть не смогла разобрать эту фразу и скорее всего там отсутствует эмоциональный контекст. Speech же означает разговорные фразы без эмоций [9]. Оба этих параметра в ходе анализа я расценивала как neutral. В данном случае сумма всех пяти перечисленных параметров равнялась нулю, так что для определения тональности текста я решила использовать следующий принцип: если значение одного из параметров positive или negative превосходит 0.4, то тексту присваивается соответствующая тональность. В противном случае текст считается нейтральным.

3.2.3 Алгоритм работы с ботом

Далее опишем полный алгоритм работы пользователя с ботом. Данное описание и картинки будет приведено для Telegram Desktop, однако оно остается верным и для мобильного приложения.

Сначала пользователь находит бота в телеграм одним из способов, описанных выше. Далее он может ознакомиться с небольшим текстом, посвящённым функционалу данного бота, после чего нажать кнопку start. Как выглядит интерфейс на данном этапе можно увидеть ниже на рисунке 4.

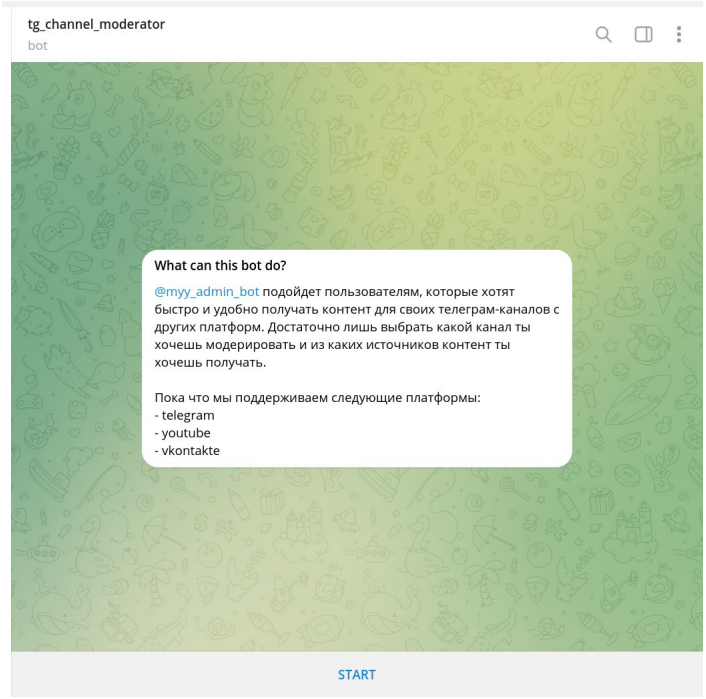

Рисунок 4 — стартовая страница работы с ботом

После нажатия кнопки start пользователю будет показана небольшая справочная информация, предлагающая ему перейти к установке канала. Пользователь переходит к команде `/set_channel`. Продолжая действовать согласно новым появляющимся справкам, пользователь пересылает боту сообщение из канала, который он хочет модерировать. На этом этапе важно не забыть сделать бота администратором в этом канале и выдать ему все возможные права, чтобы бот имел возможность делать публикации в канал. Более подробно процесс настройки канала для модерации представлен на рисунке 5.

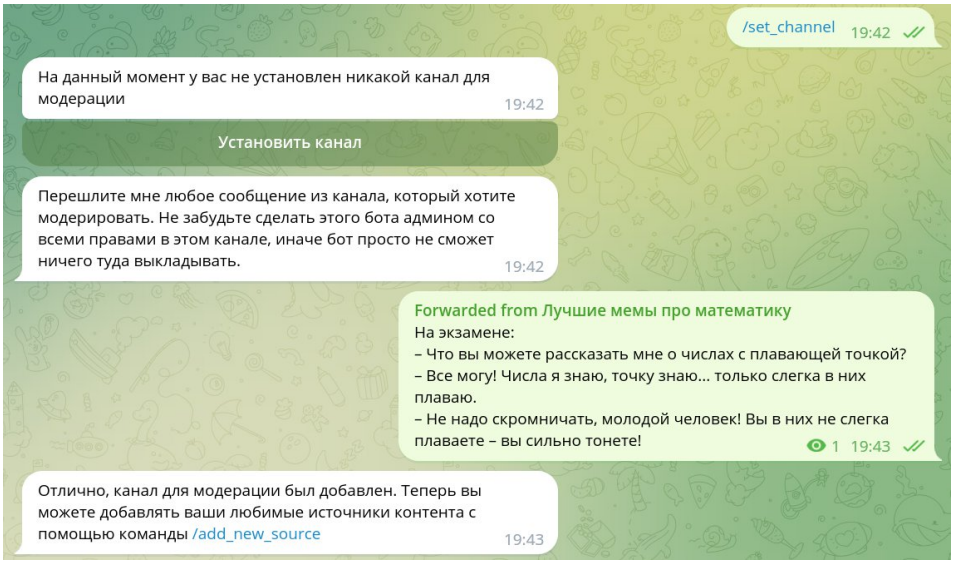

Рисунок 5 — процесс установки канала для модерации.

Если же пользователь раньше уже модерировал какой-то канал с помощью данного бота, то команда `/set_channel` предложит пользователю на выбор два варианта:

оставить текущий канал или установить новый канал. Выбор можно совершить с помощью удобных кнопок, которые показаны на рисунке 6.

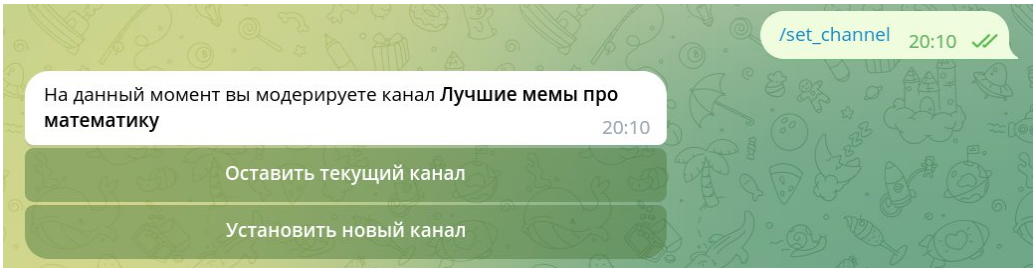

Рисунок 6 — изменение модерируемого канала

Следующий этап — добавление источников парсинга, то есть тех ресурсов, откуда пользователь хочет брать контент. Преступить к выбору источников можно с помощью команды `/add_new_source`. Бот предложит пользователю выбрать один из трёх видов источников, которые поддерживаются на текущий момент: телеграм канал, youtube канал или сообщество ВКонтакте. Осуществить выбор можно с помощью кнопок. После этого будет выведена информация о формате, в котором пользователь должен отправить ссылку на источник. В случае корректности отправленной ссылки новый источник будет добавлен в базу данных. Описанный процесс добавления источника продемонстрирован на рисунке 7.

Рисунок 7 — Процесс добавления нового источника парсинга

Процедуру добавления новых источников можно производить неоднократно, пока не будут выбраны все интересующие источники. Посмотреть список уже добавленных ранее источников можно воспользовавшись командой `/see_my_sources`. Удалить лишнее можно нажав команду `/delete_source`. В этом случае будут выведены кнопки, каждая из которых отвечает за привязанный к пользователю ресурс контента. Для удаления достаточно лишь нажать на нужные кнопки. Данный процесс изображен на рисунке 8.

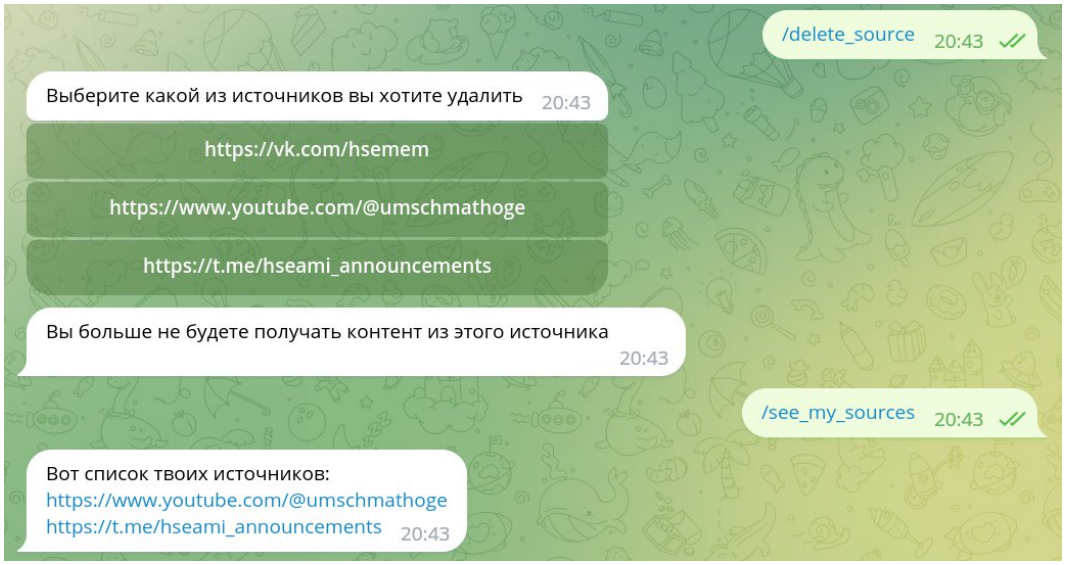

Рисунок 8 — Удаление источника и просмотр имеющихся источников

После данного этапа уже можно переходить к парсингу, то есть к получению новых данных из выбранных пользователем источников. Для начала парсинга нужно воспользоваться командой `/parse`. Далее появятся кнопки, с помощью которых пользователю будет предложено выбрать время, за которое необходимо будет найти контент. Например, пользователь сможет получить контент, выложенный за последнюю неделю или за последний день. Далее также с помощью кнопок пользователь выбирает контент какой эмоциональной окраски его интересует: позитивный, негативный, нейтральный или весь. Эта функция может быть полезна, например, для владельцев новостных телеграм каналов, которые хотят сохранять объективность и нейтральный тон публикуемых новостей. С работой бота на данном этапе можно ознакомиться на рисунке 9.

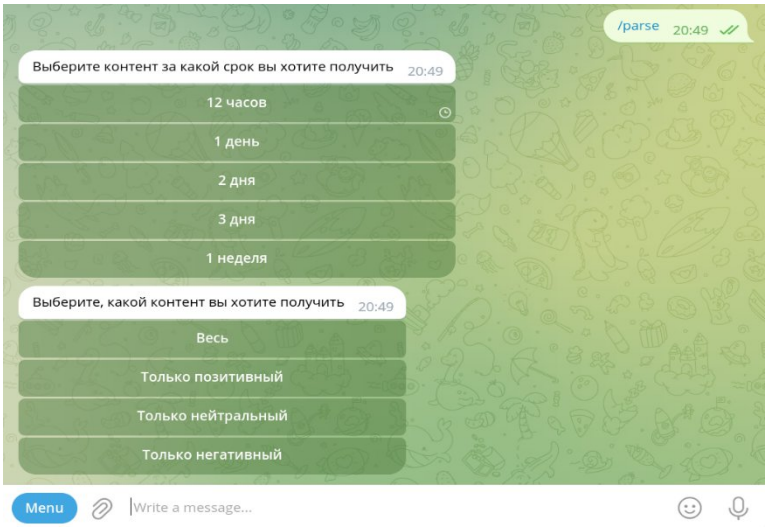

Рисунок 9 — выбор параметров парсинга

После нажатия описанных выше кнопок бот приступает к парсингу. Он выводит собранный контент в обратном хронологии порядке — начиная с новых и заканчивая старыми новостями. В конце выводится сообщение о конце парсинга. Если контента с выбранными параметрами не нашлось, то выводится соответствующее сообщение. Далее с помощью кнопок, прикрепленных к каждому посту, пользователь может принимать решения о дальнейшей судьбе поста. Во-первых, он может удалить его из истории чата и из базы данных нажав на кнопку «Удалить». Во-вторых, он может мгновенно опубликовать его в свой телеграм канал с помощью кнопки «Опубликовать это в канал». В-третьих, он может отредактировать текст поста, чтобы убрать какие-то лишние элементы или поправить форматирование. Однако на редактирование есть некоторые ограничения: пользователь не может удалить ссылку на источник, откуда взят текст, а также не может изменить текст более чем на 10%. Если пользователь совершит одно из этих действий, то бот выведет соответствующее предупреждение и не даст сохранить новый текст поста. Это сделано с целью защиты авторского права и сохранения основного смысла поста. Описанный этап работы с ботом представлен на рисунке 10.

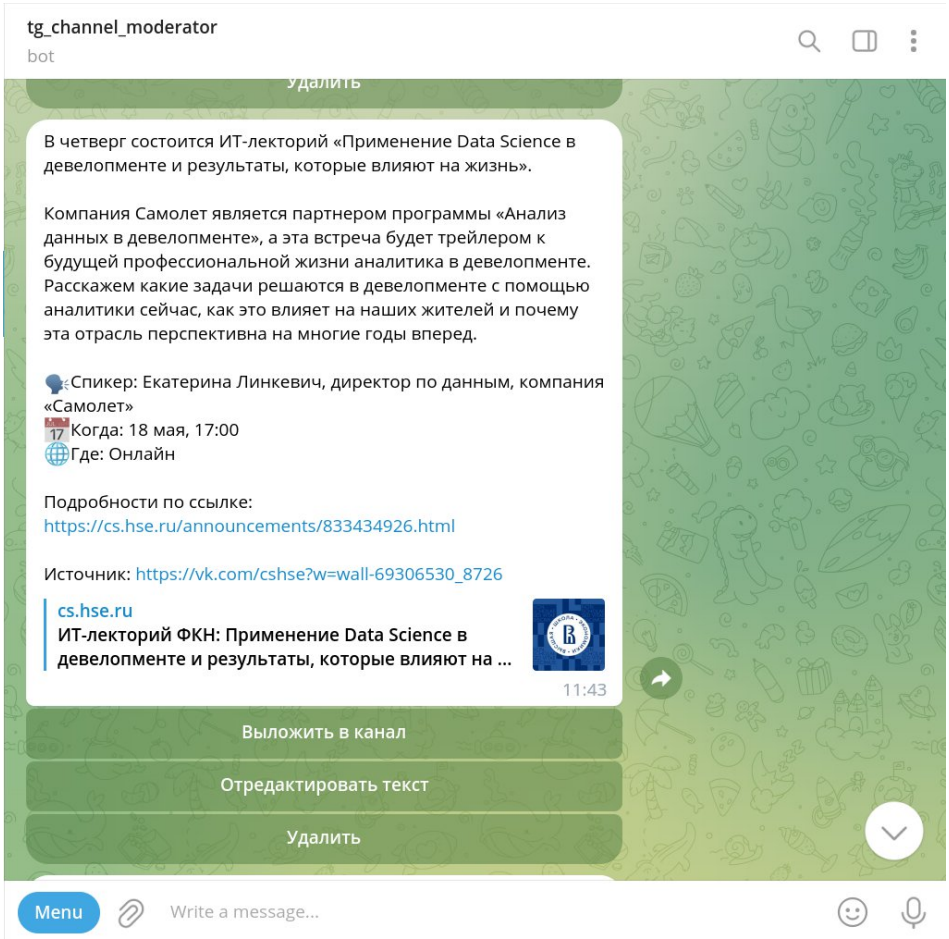

Рисунок 10 — Варианты работы с постом

На этом основной этап работы с ботом завершается. Далее пользователь может повторять любой из описанных этапов по своему усмотрению: менять модерируемый канал, добавлять и удалять источники, производить парсинг данных.

3.2.4 Развёртывание бота

Чтобы бот работал постоянно и им можно было пользоваться в любое время, было необходимо развернуть его на каком-то сервисе. Изначально я пыталась сделать это с использованием платформы PythonAnywhere, которая как раз и предназначена для проектов на языке Python. Однако по мере роста числа файлов в проекте было выявлено некоторое неудобство данного сервиса: при внесении изменений в коде программы приходилось вручную вносить их и на сервисе, удаляя устаревшие версии файлов и добавляя новые. Это доставляло много неудобств и было неэффективно. Более того, бесплатная версия хостинга ограничивала возможности доступа к сторонним ресурсам, что делало невозможными запросы на получение данных от YouTube и Telegram. Изучив альтернативные варианты, я остановила свой выбор на сервисе Railway: он предоставляет возможность брать код напрямую из репозитория на GitHub и тут же запускать работу проекта. При появлении новых коммитов в основной ветке репозитория этот сервис автоматически компилирует

новый проект и запускает его, таким образом пропадает необходимость делать какие-либо манипуляции вручную.

Однако, чтобы воспользоваться этим сервисом, пришлось разобраться, как правильно хранить приватные данные проекта: различные API-ключи и коды доступа. Ведь код берется напрямую с GitHub, а публикация этих переменных в открытом репозитории абсолютно недопустима. В итоге я воспользовалась переменными окружения, отдельно внеся их в настройки проекта на Railway.app.

3.3 Тестирование

Целью тестирования является проверка работоспособности разработанной программы. На этой стадии было необходимо проверить корректное функционирование кода в различных ситуациях и соответствие его требованиям, выдвинутым в техническом задании. Мной было проверено поведение приложения при следующих действиях пользователя:

1. Установка модерируемого канала согласно подсказкам бота
2. Отправка пользователем произвольного текста вместо пересылки сообщения из телеграм канала при установке модерируемого канала
3. Установка модерируемого канала без выдачи боту прав администратора
4. Изменение модерируемого телеграм-канала
5. Установка публичного или частного телеграм канала для модерации
6. Добавление источника парсинга согласно подсказкам бота
7. Неоднократное добавление одного и того же источника парсинга
8. Отправка некорректной ссылки на источник парсинга
9. Просмотр источников парсинга
10. Удаление одного или нескольких источников парсинга
11. Парсинг за каждый из предложенных временных периодов
12. Парсинг контента каждого из предложенных видов (всего/только позитивного/только нейтрального/только негативного)
13. Парсинг одного или нескольких сообществ вконтакте
14. Парсинг одного или нескольких каналов на ~~youtube~~YouTube

15. Парсинг одного или нескольких телеграм каналов
16. Парсинг нескольких источников данных одновременно
17. Парсинг поста, написанного с использованием языка разметки markdown
18. Парсинг источника, где нет новых данных
19. Одноразовое и многоразовое редактирование текста поста
20. Публикация поста в телеграм канал
21. Удаление поста
22. Отправка боту произвольного текста вместо следования инструкциям
23. Работа с ботом с различных телеграм-аккаунтов

Во всех перечисленных случаях бот отрабатывает корректно: бот выполняет положенные действия в случае корректного запроса от пользователя или выводит нужное сообщение об ошибке, после чего продолжает свою работу.

4. Анализ и оценка полученных результатов

В результате работы были выполнены все функциональные и нефункциональные требования к боту, указанные в разделе 2.3 данного отчёта. Бот выполняет положенные функции и корректно работает на всех проведенных тестах.

В боте корректно реализовано деление контента по эмоциональной окраске текста, что безусловно является его важным преимуществом перед другими ботами. Более того бот идентифицирует и не показывает пользователю рекламные посты. В условиях переизбытка поступающей нам информации очень важно иметь возможность фильтровать контент и оставлять только тот, что нам действительно нужен.

Однако есть и некоторые недочеты, исправить которые пока не получилось. Например, не был найден способ парсить ~~Youtube~~YouTube-каналы без дневного лимита. Так, пока проект небольшой и им пользуется небольшое количество людей, этого лимита будет вполне достаточно, однако по мере роста проекта эту проблему придётся как-то решать. Возможно придётся задействовать не официальное API ~~Youtube~~YouTube, а некоторые сторонние библиотеки и приложения для доступа к видео без ограничений.

Также пока не очень велик список доступных платформ для парсинга: telegram, ~~Youtube~~YouTube и ~~вконтакте~~ВКонтакте. Наверняка многие пользователи хотели бы брать данных и из других видов источников.

Заключение

На данный момент реализована значительная часть функционала. Бот запущен на сервере и его можно использовать в телеграм, найдя его по ссылке https://t.me/myy_admin_bot или по имени @myy_admin_bot. Для парсинга доступны такие источники, как сообщества ~~ВКонтакте~~ВКонтакте, чаты telegram и ~~Youtube~~YouTube каналы. Каждый пользователь может модерировать один телеграм канал: планировать отправку в него сообщений.

В будущем проект планируется расширить, добавить в него новый функционал. Например, к нему можно будет добавить поддержку парсинга других источников данных, таких как Instagram, TikTok и Facebook. Будет также добавлена возможность модерации одним пользователем нескольких телеграм каналов одновременно, будет расширен функционал взаимодействия с каналом, например, планирование публикации поста на определённое время, удаление старого поста, просмотр статистики по каналу.

Список источников

1. Подборка чат-ботов для маркетплейсов [Электронный ресурс], 2022. URL: https://botcreators.ru/blog/podborka-chat-botov-dlja-marketplejsov/ . (Дата обращения: 24.01.2023)
2. Бот для маркетплейсов: Вайлдберриз, Озон и других маркетплейсов [Электронный ресурс], 2021. URL: https://wbcon.ru/2021/10/31/bot-parser-mp-telegram/#. (Дата обращения: 24.01.2023)
3. Андрей Петрович, Лучшие способы как спарсить контакты из канала и чата в Телеграме [Электронный ресурс], 2022. URL: https://tgfaq.ru/sparsit-kontakty-iz-kanala.html. (Дата обращения: 24.01.2023)
4. Роман Юрьевич, Бот для канала в Telegram [Электронный ресурс], 2022. URL: https://vc.ru/marketing/470490-bot-dlya-kanala-v-telegram. (Дата обращения: 16.01.2023)
5. LegGnom, Python: Scrapy, Selenium, Beautiful Soup что лучше для парсинга веб сайтов [Электронный ресурс], 2019. URL: https://dev-gang.ru/article/python-scrapy-selenium-beautiful-soup-czto-luczshe-dlja-parsinga-veb-saitov-rmv3n1q3us/ (Дата обращения: 21.02.2023)
6. Бот в Telegram на Питоне от А до Я [Электронный ресурс], 2022. URL: https://otus.ru/journal/bot-v-telegram-na-pitone-ot-a-do-ya/ (Дата обращения: 21.02.2023)
7. tmat, Всё, о чём должен знать разработчик Телеграм-ботов [Электронный ресурс], 2021. URL: https://habr.com/ru/post/543676/ (Дата обращения: 21.02.2023)
8. Пётр Ивлев, 16 сервисов по ведению чатов и каналов в Telegram [Электронный ресурс], 2020. URL: https://vc.ru/services/128424-16-servisov-po-vedeniyu-chatov-i-kanalov-v-telegram (Дата обращения: 08.05.2023)
9. Егоров Егор, Dostoevsky — анализ тональности в Python за 5 минут [Электронный ресурс], 2021. URL: https://egorovegor.ru/analiz-tonalnosti-s-python-i-dostoevsky (Дата обращения: 12.05.2023)
10. Сергей Бондаренко, Гайд по API ВК: как подключить и использовать [Электронный ресурс], 2023. URL: https://smmplanner.com/blog/gaid-po-api-vk-kak-podkliuchit-i-ispolzovat/ (Дата обращения: 17.05.2023)

Приложения

Приложение А

Инструкция по установке/развертыванию приложения

Приложение не предполагает самостоятельной установки и развертывания пользователем. Бот запущен на хостинге и им можно воспользоваться, найдя его в телеграм по названию tg_channel_moderator или по нику @myy_admin_bot.