УДК 004.89

АВТОМАТИЗАЦИЯ ПРОЦЕССА СОЗДАНИЯ ТЕСТОВ ПРИ ПОМОЩИ БОТА НА ПЛАТФОРМЕ TELEGRAM С ИСПОЛЬЗОВАНИЕМ OPENAI

Исомадинов А.Х., студент гр. ИТб-202, III курс
Научный руководитель: Алексеевна Г.А., старший преподаватель
Кузбасский государственный технический университет
имени Т. Ф. Горбачева
г. Кемерово

Важной составляющей процесса обучения является контроль знаний. В условиях, когда непрерывно возрастает доля дистанционных образовательных технологий, значительная часть контроля знаний осуществляется с применением тестирования. Составление тестов, используемых для проверки уровня освоения изученного материала, задача трудоемкая. Актуальность автоматизация процесса составления тестовых задания сложно переоценить. В результате анализа основных направлений исследований в данном вопросе [1] было принято решение о создании AI-чат-бота на платформе Telegram. Такие боты очень популярны среди пользователей и могут способствовать получению новых знаний и улучшению организации рабочего процесса. Рассматриваемый AI-чат-бот автоматизирует процесс составления тестовых вопросов на основе текста, отправленного пользователем. Для реализации используется язык программирования Python, библиотеки OpenAI и aiogram.

OpenAI [2] – платформа искусственного интеллекта, которая предоставляет API для создания мощных NLP-приложений. Одним из главных преимуществ OpenAI является доступность передовых технологий искусственного интеллекта и возможность использовать их для создания решений в различных сферах деятельности. Aiogram – это библиотека Python, которая упрощает создание чат-ботов Telegram.

Для работы необходимо получить токен для работы с Telegram и ключ API OpenAI. Последовательность создания бота представлена на рисунке 1.

Реализованы три обработчика:

1. Реагирует на команду «/start» и создает клавиатуру с тремя кнопками: «Тесты», «Вопросы» и «Смысл» (рисунок 2), а затем отправляет приветственное сообщение пользователю и переходит в состояние ожидания выбора.

2. Реагирует на нажатие одной из трех кнопок на клавиатуре и сохраняет выбранное пользователем действие в контексте. Затем бот отправляет запрос на ввод текста и переходит в состояние ожидания ввода текста пользователем (рисунок 3).

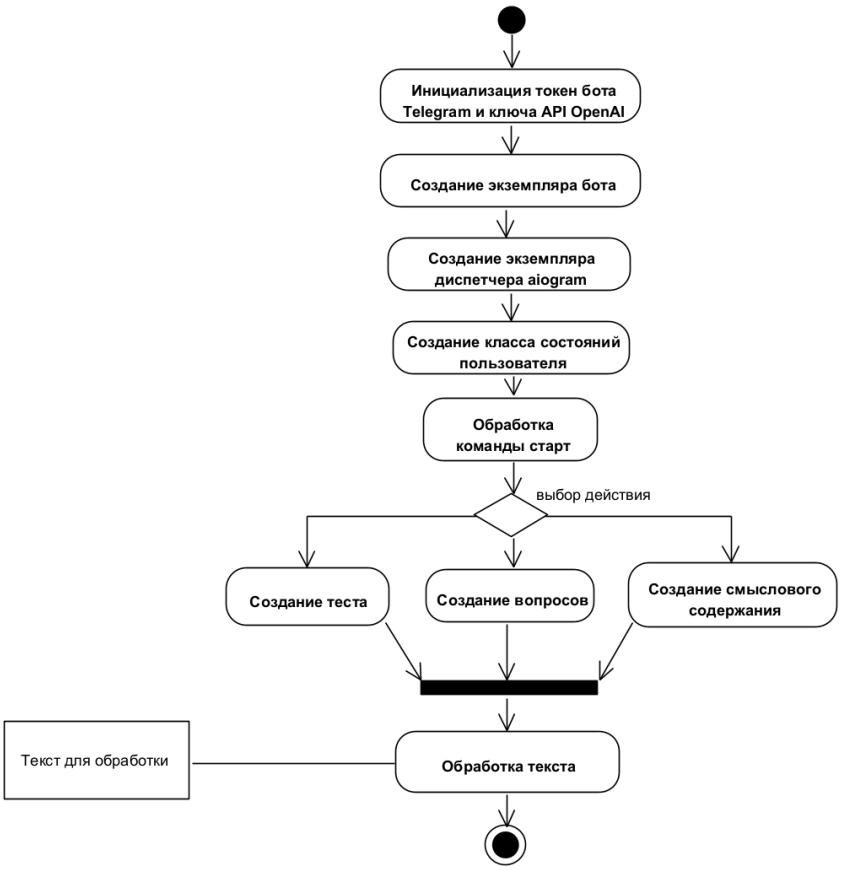

Рисунок 1

3. Обработчик ввода текстового сообщения пользователем в чат-бот, который использует API OpenAI для генерации ответа на основе введенного текста и выбранного действия.

Обработчик нажатия на кнопки использует лямбда-функцию, чтобы проверить, содержит ли сообщение текст соответствующий одной из трех кнопок на клавиатуре, когда пользователь находится в состоянии ожидания выбора. Это сделано для того, чтобы избежать конфликта с другими обработчиками, которые могут реагировать на сообщения в любое время.

Текст сохранятся в контексте с использованием библиотеки aiogram.

Затем формируется запрос к API OpenAI, в котором используется модель «text-davinci-003». Запрос зависит от выбранного действия. Если пользователь выбрал «Тесты», то в запросе указывается, что нужно составить 4 теста с вариантами A, B, C, D по тексту и в конце написать правильные ответы. Если пользователь выбрал «Вопросы», то в запросе указывается, что нужно составить 4 вопроса по тексту и ответы по вопросам. Если пользователь выбрал «Смысл», то в запросе указывается, что нужно сформировать краткое смысловое содержание.

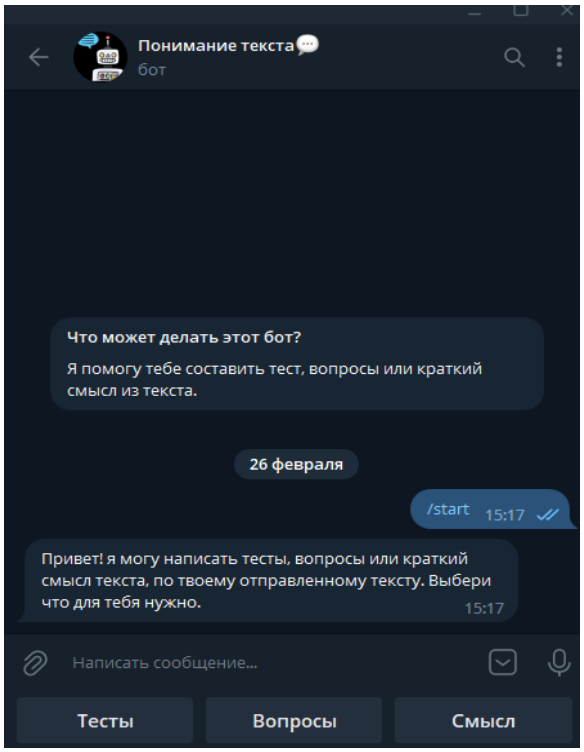

Рисунок 2

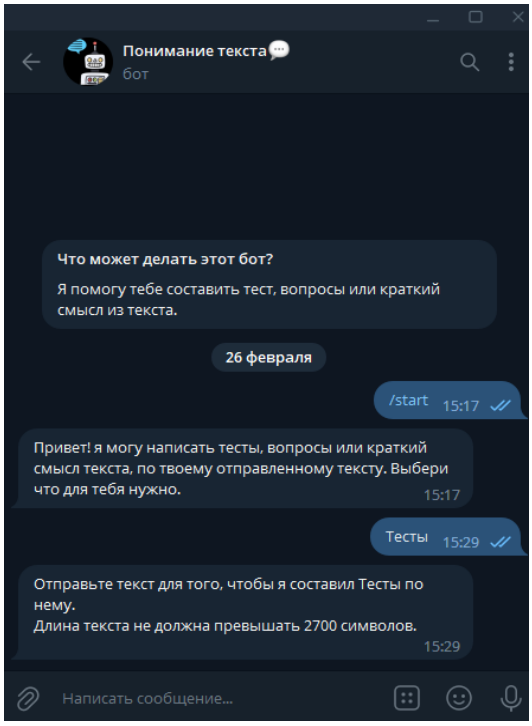

Рисунок 3

Для получения ответа от API OpenAI, используется библиотека openai. Ответ передается пользователю в виде сообщения.

По завершению формирования результата выполнения задания формируется кнопка «Назад» (на рисунке 1 не отображена), которая позволяет вернуться к выбору действий (рисунок 4).

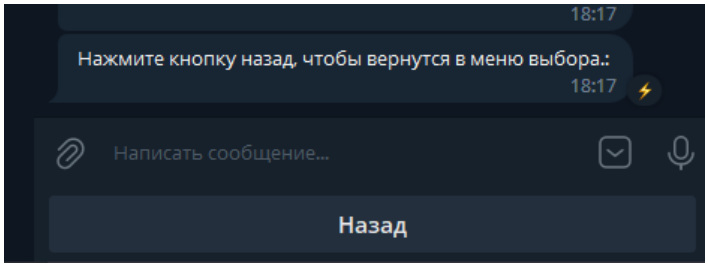

Рисунок 4

Код, представленный на рисунке 5, запускает бесконечный цикл опроса Telegram API с помощью функции start_polling из библиотеки aiogram. Параметр dp указывает на экземпляр класса Dispatcher, который содержит обработчики для обработки входящих сообщений. Параметр skip_updates=True указывает на пропуск обновлений, которые пришли до запуска бота.

```
if __name__ == '__main__':
    executor.start_polling(dp, skip_updates=True)
```

Рисунок 5

Конструкция if __name__ == '__main__': используется для того, чтобы код был выполнен только в случае, если модуль был запущен как главный файл программы, а не импортирован в другой модуль.

Пример текста для обработки приведен на рисунке 6, результаты работы представлены на рисунках 7-9.

В результате работы над проектом были выявлены некоторые недостатки OpenAI:

1. Ограниченный доступ к функциям. Некоторые функции доступны только для платных пользователей, что может быть проблемой для небольших компаний и стартапов.

2. Ограниченные возможности интеграции. Некоторые инструменты OpenAI могут быть ограничены в интеграции с другими системами, что может вызвать проблемы при разработке решений, использующих несколько систем.

3. Безопасность и конфиденциальность. Использование искусственного интеллекта может привести к серьезным проблемам с конфиденциальностью и безопасностью данных, если не предпринимать соответствующие меры защиты.

4.Ограниченная точность. Как и любая система искусственного интеллекта, OpenAI не всегда дает точные результаты и может допускать ошибки. Это может стать проблемой при разработке высокоточных решений.

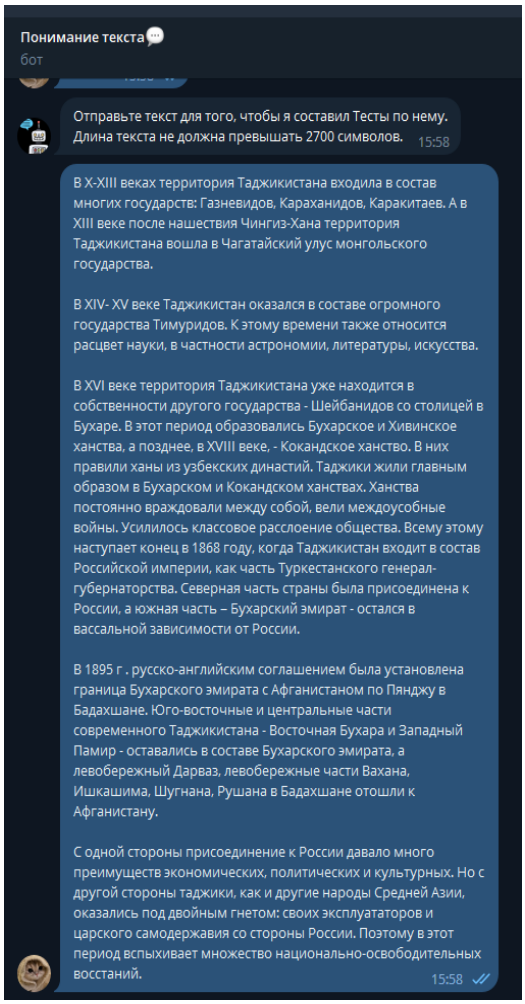

Рисунок 6

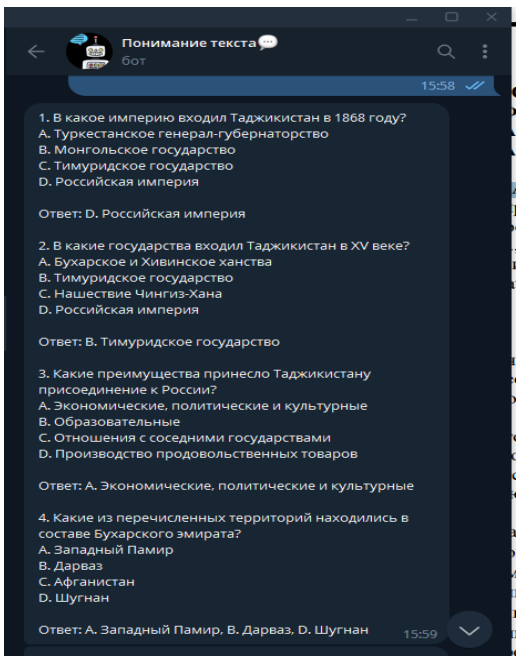

Рисунок 7

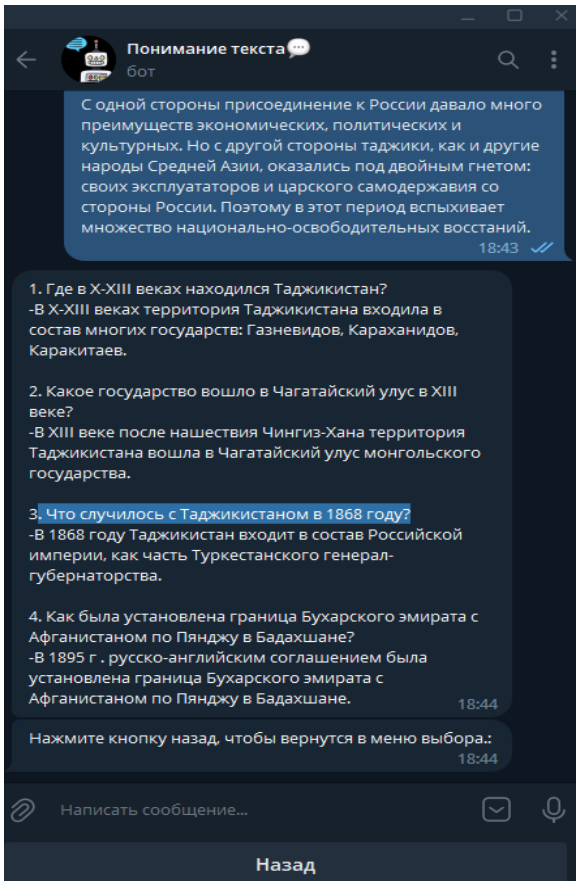

Рисунок 8

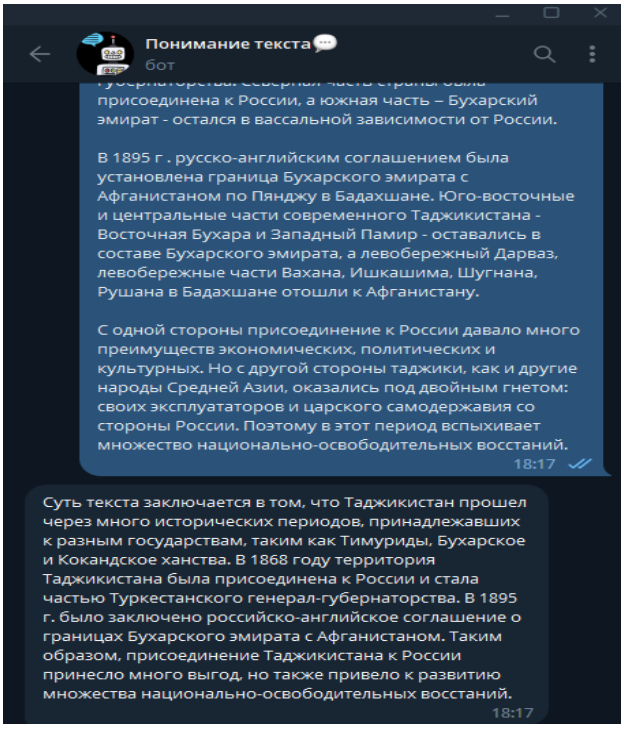

Рисунок 9

В данной статье было рассмотрено использование библиотеки OpenAI для генерации текстов на основе некоторых шаблонов и запросов от пользователя. Это открывает большие возможности для различных приложений, таких как генерация новостных статей, рекламных текстов и т.д. Также реализованное решение может быть использовано в учебном процессе для снижения нагрузки по составлению тестовых вопросов на преподавательский состав и в других ситуациях, когда необходимо оценить уровень полученных знаний.

Кроме того, рассмотрены преимущества и недостатки работы с OpenAI. Среди преимуществ можно выделить высокую точность генерации текстов, возможность работы с неструктурированными данными и простоту использования API. Однако, существуют и некоторые ограничения в применении: высокую стоимость при использовании всех функций, ограниченность доступа к API и необходимость в большом объеме обучающих данных для оптимальной работы моделей.

Таким образом, использование OpenAI для генерации текстов может быть очень полезным и эффективным, но также может требовать дополнительных ресурсов при выборе подходящей модели и настройке ее параметров.

Список литературы:

1. Швецов, А.Н., Куртасов, А.М. Метод автоматизированной генерации контрольно-тестовых заданий из текста учебных материалов [Электронный ресурс] // Вестник Череповецкого государственного университета. – 2014. – № 7.– Режим доступа: https://cyberleninka.ru/article/n/metod-avtomatizirovannoy-generatsii-kontrolno-testovyh-zadaniy-iz-teksta-uchebnyh-materialov/viewer, свободный. (Дата обращения 04.03.2023)

2. Официальная документация OpenAI [Электронный ресурс]. – Режим доступа: https://beta.openai.com/docs/, свободный. (Дата обращения 09.03.2023)

3. Hands-On Chatbots and Conversational UI Development, Srini Janarthanam [Электронный ресурс]. – Режим доступа: https://www.packtpub.com/product/hands-on-chatbots-and-conversational-ui-development/9781801072978, свободный. (Дата обращения 18.03.2023)

3. How to Build a Telegram Chatbot in Python with OpenAI's GPT-3, Twilio[Электронный ресурс]. – Режим доступа: https://www.twilio.com/blog/how-to-build-a-telegram-chatbot-in-python-with-openais-gpt-3, свободный. (Дата обращения 09.03.2023)