

ПРОГРАММНОЕ СРЕДСТВО «BOTAI»

СОДЕРЖАНИЕ

ВВЕДЕНИЕ	2
1 ПОСТАНОВКА ЗАДАЧИ.....	5
1.1 Описание предметной области	5
1.2 Обзор существующих аналогов	6
1.3 Информационная база задачи	9
1.4 Функциональное назначение	10
2 ПРОЕКТИРОВАНИЕ ЗАДАЧИ	11
2.1 Алгоритм решения задачи	11
2.2 Логическое моделирование	12
2.3 Выбор и обоснование инструментов разработки	14
3 ПРОГРАММНАЯ РЕАЛИЗАЦИЯ.....	17
3.1 Физическая структура	17
3.2 Описание разработанных модулей	18
4 ТЕСТИРОВАНИЕ	22
5 ПРИМЕНЕНИЕ ПРОГРАММЫ.....	25
5.1 Назначения и условия применения.....	25
5.2 Руководство пользователя	25
ЗАКЛЮЧЕНИЕ	28
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ	29
ПРИЛОЖЕНИЕ А.....	30

Изм.	Лист	№ докум.	Подпись	Дата	Программное средство «BotAI» Пояснительная записка			Лит.	Лист	Листов		
Разраб.											2	36
Провер.												
Реценз.												
Н. Контр.												
Утверд.												

ВВЕДЕНИЕ

Бот с искусственным интеллектом (AI-бот) – это программа, которая имитирует человеческий разговор с помощью искусственного интеллекта. Он может использоваться для общения с людьми через мессенджеры, социальные сети или другие каналы коммуникации.

AI-боты используются в различных областях, включая клиентское обслуживание, продажи, маркетинг, управление процессами и другие. Они могут выполнять задачи, такие как ответ на часто задаваемые вопросы, предоставление информации, помощь в принятии решений и даже обработка заказов.

AI-боты обучаются на большом объеме данных, используя методы машинного обучения, такие как нейронные сети и алгоритмы обработки естественного языка. Они могут улучшать свою работу с течением времени, используя данные об интеракциях с пользователями, чтобы улучшить свои ответы и повысить уровень удовлетворенности клиентов.

В данном проекте необходимо разработать бот с использованием языка программирования C# и искусственного интеллекта.

Основной целью проекта является создание AI-бота, способного взаимодействовать с пользователями через популярный мессенджер Telegram.

Разработка Telegram бота с искусственным интеллектом на языке C# актуальна в контексте повышения эффективности коммуникации с пользователями, улучшения качества клиентского обслуживания и расширения возможностей бизнес-процессов.

Этот проект предоставляет возможность исследовать и применить передовые технологии и методы искусственного интеллекта, а также развить навыки разработки программного обеспечения на популярном языке программирования.

Задачи проектирования:

- изучение основных принципов работы Telegram ботов и их интеграции с платформой Telegram;
- изучение основ искусственного интеллекта, включая методы машинного обучения и алгоритмы обработки естественного языка;
- проектирование и разработка архитектуры бота с использованием языка программирования C# и API;
- реализация функциональности бота, включая ответы на вопросы, и обработка различных форматов данных;
- тестирование разработанного бота;
- документирование процесса разработки.

В разделе «Постановка задачи» отражено функциональное назначение проекта, поставлена задача на проектирование, сформулированы задачи, приведены примеры аналогов программного средства.

В разделе «Проектирование задачи» представлены алгоритм решения задачи и логическое моделирование.

В разделе «Программная реализация» описаны программная реализация проекта, а именно описание инструментов разработки программного средства, описание разработанных компонентов программного средства и описание разработанных модулей приложения.

Раздел «Тестирование» отражает проверку правильности и работоспособности отдельных функций и программной системы в целом.

Раздел «Применение» включает описание процесса установки и запуска приложения и системные требования к программному средству, руководство пользователя, отражающее основные возможности работы программы.

Раздел «Заключение» содержит обобщение проделанной работы, итоги и выводы.

В разделе «Список использованных источников» расположен перечень источников, цитируемых и изученных при написании проекта.

Приложение А включает результат работы программного средства.

1 ПОСТАНОВКА ЗАДАЧИ

1.1 Описание предметной области

Предметная область разрабатываемого программного средства – коммуникация и обработка запросов с использованием Telegram бота с искусственным интеллектом на языке программирования C#.

В данном контексте, предметная область охватывает взаимодействие пользователя с ботом через мессенджер Telegram и предоставление автоматизированных ответов, услуг и функций. Она включает в себя различные сферы жизни и работы, где применение такого бота может быть полезным и актуальным.

Искусственный интеллект – область компьютерной науки, которая занимается созданием интеллектуальных систем, способных выполнять задачи, требующие обычно человеческого интеллекта. В данном проекте искусственный интеллект используется для обработки и анализа входящих запросов, понимания естественного языка и предоставления соответствующих ответов.

Технология GPT (Generative Pre-trained Transformer) представляет собой нейронную сеть, способную генерировать текст, отвечая на вопросы и поддерживая диалог с пользователями. Модель обучается на большом объеме текстовых данных, чтобы обладать широким общим знанием и способностью понимать различные типы вопросов и запросов.

Технология распознавания речи (Speech Recognition) – это область искусственного интеллекта, которая занимается преобразованием звуковой речи в текстовую форму. Она использует алгоритмы и модели машинного обучения для анализа аудио-сигналов и определения содержащихся в них слов и фраз. Распознавание речи позволяет боту понимать и обрабатывать голосовые команды и запросы пользователей, что улучшает интерактивность и удобство использования программного средства.

Технология распознавания текста с фото (Optical Character Recognition) – технология, позволяющая автоматически распознавать текст на изображениях и преобразовывать его в редактируемый текстовый формат. Алгоритмы OCR обнаруживают и анализируют паттерны, соответствующие символам и словам на изображении, и восстанавливают содержащийся текст. Эта технология позволяет боту обрабатывать фотографии с текстом, например, снимки документов, вывести текст на экране для редактирования или использовать его для дальнейшего анализа и обработки.

Технология генерации изображений по описанию (Image Generation) – это область искусственного интеллекта и компьютерного зрения, которая использует

модели глубокого обучения для создания изображений на основе текстового описания. Алгоритмы генерации изображений позволяют боту понять описание сцены или объекта и синтезировать соответствующее изображение, которое наиболее точно соответствует описанию. Такая технология может использоваться, например, для создания иллюстраций или визуализации предметов, о которых говорит пользователь.

API Telegram предоставляет программный интерфейс приложения (API), который позволяет разработчикам создавать и интегрировать свои боты. В проекте используется API Telegram для взаимодействия между ботом и мессенджером.

Бот должен быть способен обрабатывать входящие запросы от пользователей, анализировать их и определять необходимые действия. Обработка запросов включает распознавание намерений пользователя, классификацию текста, извлечение ключевой информации и т. д.

1.2 Обзор существующих аналогов

OpenAI, компания-разработчик API ChatGPT предоставляет доступ к своим моделям машинного обучения через API для личного использования через одноименный чат-бот ChatGPT.

Чат-бот представляет собой диалоговое окно на веб-странице как на рисунке 1.1. Сервис содержит поле для ввода запроса пользователя и область генерации ответа нейросетью в режиме онлайн.

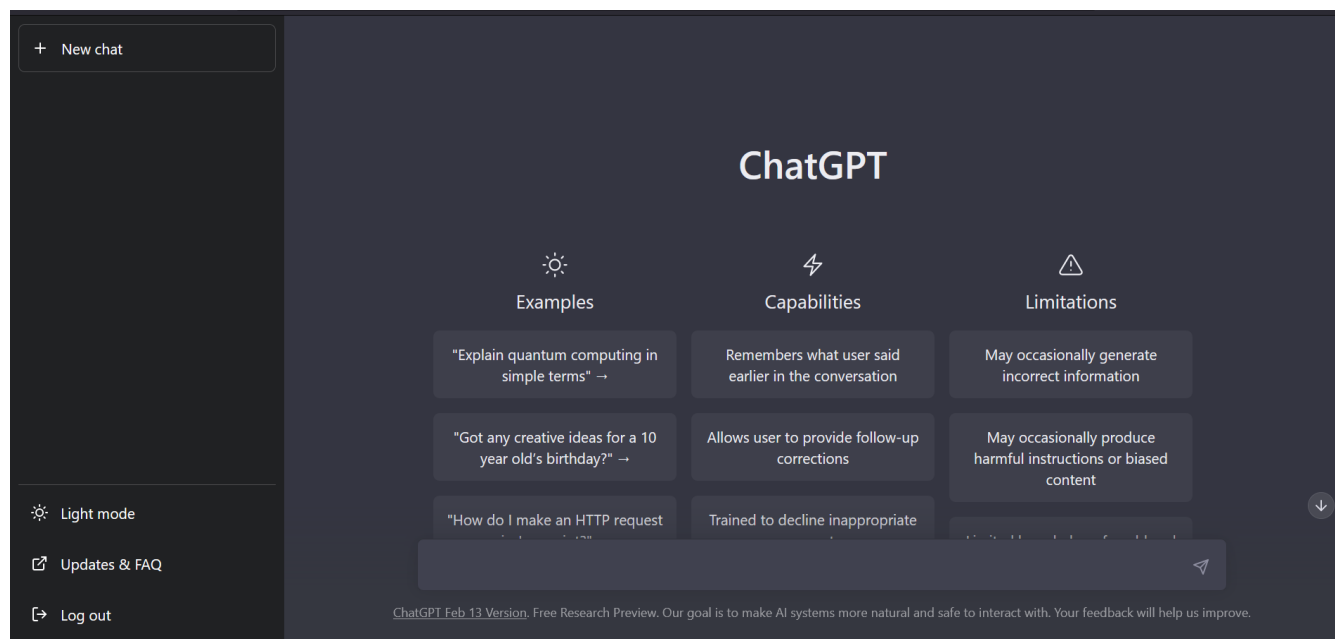


Рисунок 1.1 – Интерфейс ChatGPT

В случае, если ответ не соответствует ожиданиям пользователя, следует попробовать перефразировать вопрос. Важно понимать, что основной принцип работы языковой модели ChatGPT заключается в том, чтобы додумать и предсказать, что имел в виду пользователь. Поэтому чем более конкретный запрос, тем более релевантный будет результат. Чат редко отвечает, что не понял вопрос, обычно он в любом случае дает развернутый ответ.

Достоинства:

- высокий уровень обработки естественного языка и генерации текста;
- широкий набор функций для общения с пользователем через текстовые сообщения;
- возможность интеграции с различными платформами и каналами коммуникации.

Недостатки:

- отсутствие специализированных функций для обработки изображений, аудио и видео;
- платный.

Quizard – это фотосканер, использующий искусственный интеллект для получения ответов на вопросы. Для начала необходимо сфотографировать вопрос, как на рисунке 1.2, а приложение даст на него ответ и объяснение решения, при необходимости.

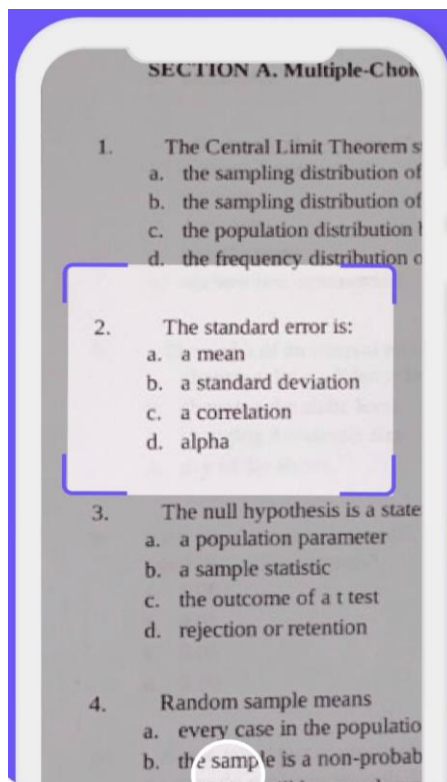


Рисунок 1.2 – Интерфейс Quizard

Достоинства:

- быстрое и удобное сканирование задач;
- поддержка различных типов вопросов, включая множественный выбор и краткие ответы;
- понятные объяснения решений.

Недостатки:

- ограничения функциональности, для сложных или творческих задач может потребоваться дополнительная помощь или самостоятельная работа;
- потенциальное использование для мошенничества.

Murf.ai на рисунке 1.3 является платформой, предоставляющей инструменты и решения для разработки чат-ботов с использованием искусственного интеллекта.

Сайт предлагают набор функций, таких как распознавание речи, обработка естественного языка, автоматическое формирование ответов и другие возможности, которые позволяют создавать интеллектуальных ботов с минимальным программированием.

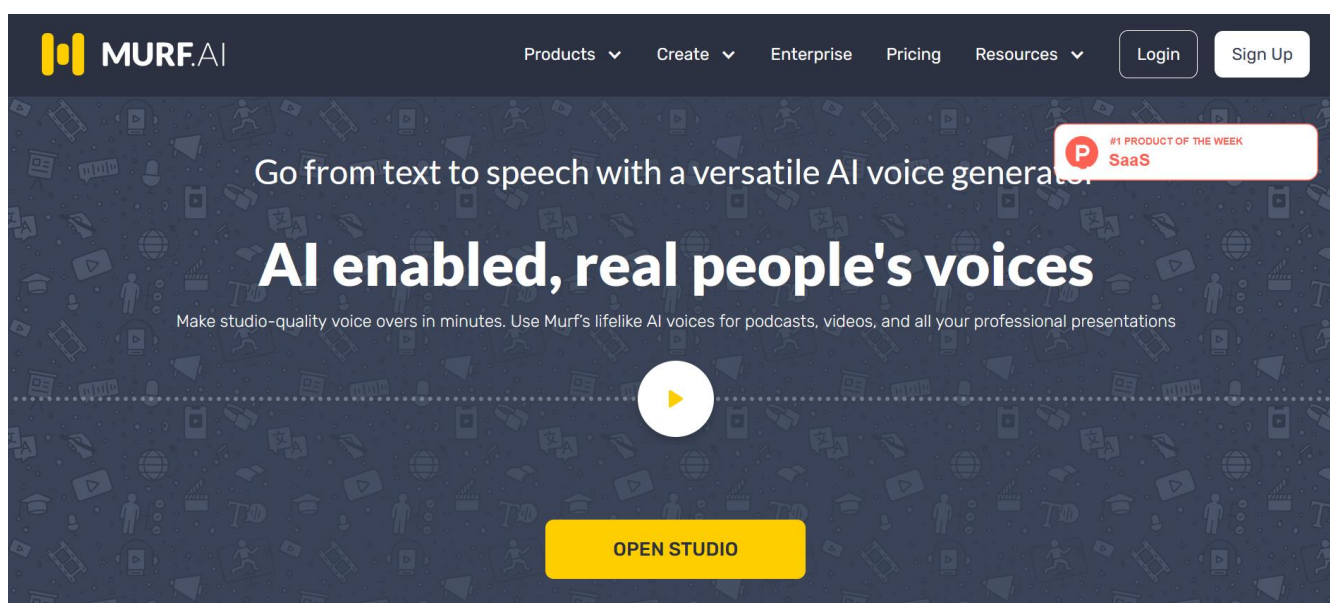


Рисунок 1.3 – Интерфейс Murf.ai

На платформе Murf.ai можно разрабатывать и настраивать ботов, определять их поведение, обучать модели и интегрировать их с различными каналами коммуникации, включая Telegram. Это удобный инструмент для создания ботов с искусственным интеллектом без необходимости полного программирования с нуля.

Достоинства:

- предоставление инструментов и решений для разработки чат-ботов с использованием искусственного интеллекта;
- возможность работы с распознаванием речи, обработкой естественного языка и формированием ответов;
- удобный интерфейс для создания и настройки ботов без необходимости полного программирования.

Недостатки:

- ограниченная функциональность в сравнении с полноценной разработкой с использованием языка программирования;
- ограничения в настройке и интеграции с другими системами и платформами.

Разрабатываемый Telegram бот на языке C# с искусственным интеллектом предлагает пользователям широкий спектр возможностей, позволяющих обмениваться информацией различными форматами, включая текстовые сообщения, изображения, аудио и видео.

Достоинства:

- широкий набор функций для обработки текстовых, голосовых, изображений и видео данных;
- интеграция с платформой Telegram для общения с пользователем;
- простота использования и доступность для широкого круга пользователей.

Недостатки:

- ограниченные общие возможности в сравнении со специализированными решениями;
- ограничения в масштабируемости и поддержке больших объемов данных;
- необходимость в дополнительной разработке и поддержке для расширения функциональности и интеграции с другими системами.

1.3 Информационная база задачи

Информационная база задачи состоит из входной и выходной информации.

Входная информация представляет собой загружаемые данные при запуске приложения.

Выходная информация – информация, которую пользователь получает в результате выполнения функций программы.

Информационная база задачи состоит из входной и выходной информации.

Входная информация представлена данными, отправляемыми пользователем. Это может быть текстовая информация, фото, видео, аудио, YouTube ссылки.

Форматы, которые программное средство не умеет обрабатывать воспринимаются как исключения и бот может оповестить пользователя, что не знает, что получил.

Выходной информацией являются текстовые ответы на запросы пользователя, изображения, фото, видео, аудио и стикеры.

1.4 Функциональное назначение

Функциональное назначение заключается в создании программного средства, основанного на технологиях разработки ботов для платформы Telegram и использовании различных API искусственного интеллекта.

Целью проекта является разработка многофункционального бота, способного эффективно взаимодействовать с пользователями и предоставлять им широкий спектр сервисов и возможностей. Проект также нацелен на углубление и систематизацию теоретических знаний и практических умений в области программирования, изучение этапов жизненного цикла программного средства и основ объектно-ориентированного программирования.

Для достижения поставленной цели были определены следующие задачи:

- разработка интерфейса и интеграция с Telegram. Приложение должно обеспечивать интуитивно понятный и удобный интерфейс для взаимодействия с пользователями через мессенджер Telegram. Это включает в себя настройку подключения к Telegram API и обеспечение надежной и безопасной передачи данных;

- обеспечение взаимодействия различных API искусственного интеллекта. Программное средство должно обладать способностью анализировать и обрабатывать различные типы данных, такие как текст, изображения, аудио и видео. Это включает в себя функции распознавания текста на фото, преобразования голоса в текст и обратно, а также функции связанные с обработкой и анализом видео;

- реализация функций обработки и анализа данных. Приложение должно предоставлять возможность пользователю выполнять различные операции с данными, такие как подводить итоги, изменять и сокращать текст, переводить на иностранные языки и многое другое. Для этого требуется разработать соответствующие алгоритмы и функциональность.

Таким образом, разрабатываемое приложение «BotAI» предлагает широкий спектр функций, включающих обработку различных типов данных, использование искусственного интеллекта и возможность настройки взаимодействия с ботом для удовлетворения различных потребностей пользователей.

2 ПРОЕКТИРОВАНИЕ ЗАДАЧИ

2.1 Алгоритм решения задачи

Разработанная программная система:

- содержит стартовое сообщение, запускающее работу бота;
- меню, при помощи которого можно посмотреть информацию о функциональных возможностях бота и вызывать функции «Video to audio», «Text to speech», «Text to image»;
- поле ввода сообщений пользователем;
- область отображения чата.

Программная система находится в режиме ожидания сообщения от пользователя. При получении сообщения определяется его формат: стартовое сообщение, пункт меню, текстовое сообщение, фото, аудио, видео, нажатие на кнопку или нераспознанный формат.

Если ботом было получено стартовое сообщение, в ответ отправляется приветствие.

Когда из меню выбрано просмотр описания функций бота, отправляется ответ в виде соответствующей информации.

На текстовое сообщение бот отвечает при помощи API OpenAI ChatGPT.

Фото обрабатывается в текст при помощи OCR. Полученный текст отправляется пользователю вместе с 3 кнопками: «answer» и «summary».

Кнопка «answer» вызывает ответ на текст с фотографии, «summary» подытоживает то, что было изложено с фотографированном тексте.

При получении аудио и видео, голос переводится в текст и отображается пользователю. Полученное сообщение можно подытожить, выбрав кнопку «summary».

Специфичная обработка текста вызывается функциями /tti и /tts, «Text to image», «Text to speech», соответственно. Для получения генерации фото на основе написанного текста или преобразования текста в речь, необходимо отправить сам текст вместе с необходимой функцией в одном сообщении.

При вызове функции /vta, то есть «Video to audio», бот сперва отправляет сообщение о своей готовности, а потом переходит в режим ожидания отправки пользователем видео. После он преобразовывает отправленное видео в аудио.

Когда бот получает YouTube ссылку, он предлагает возможные с ней действия: предоставление возможности скачать видео, аудио, преобразовать сказанное в ролике в текст, подытожить сказанное. Для выбора действия необходимо нажать на соответствующую кнопку, после чего алгоритм начнет обработку.

Иногда, во время обработки запроса, бот отправляет стикеры, пока пользователь ожидает ответ.

Если бот получает формат, с которым не умеет работать, отправляется общее сообщение, что формат не определен. При иных ошибках отправляются сообщения о сбоях и их возможных причинах, при этом бот может продолжать свою работу.

2.2 Логическое моделирование

Визуальное моделирование в UML можно представить как процесс поуровневого спуска от общей и абстрактной концептуальной модели исходной системы к логической, а затем к физической.

Назначение диаграммы вариантов использования (Use Case) в вашем программном средстве состоит в моделировании и описании взаимодействия между актерами (пользователями или внешними системами) и самим программным средством. Она позволяет идентифицировать основные функциональные возможности и задачи, которые пользователи могут выполнять с помощью вашего бота.

Разработка диаграммы вариантов использования преследует цели:

- определить общие границы и контекст моделируемой предметной области на начальных этапах проектирования системы;
- сформулировать общие требования к функциональному поведению проектируемой системы;
- разработать исходную концептуальную модель системы для ее последующей детализации в форме логических и физических моделей;
- подготовить исходную документацию для взаимодействия разработчиков системы с ее заказчиками и пользователями.

При первом обращении к программному средству, пользователь отправляет стартовое сообщение, после чего может начинать работу. С этого момента на его усмотрение можно выбрать несколько команд, отправлять текстовые сообщения или файлы различных форматов.

При отправке команды /help, пользователь получит описание функциональных возможностей бота. Команда /tts вместе с текстом в одном сообщении вызовет метод, который отправит пользователю генерацию речи. При использовании команды /tti вместе с описанием необходимой генерации, бот отправит изображение, созданное на основе отправленного пользователем текста. Команда /vta предназначена для последующего отправления пользователем видео, которое бот сконвертирует в аудио.

На простое текстовое сообщение бот генерирует ответ. При отправке фото, приложение отправит распознанный текст вместе с кнопками «answer» и

«summary», при нажатии на которые бот отправить сообщение с ответом на вопрос и подведением итога, соответственно. Отправляя видео или аудио, пользователь получит транскрибированный текст, который можно обобщить нажатием на кнопку «summary». Пользователь имеет возможность отправить YouTube-ссылку и выбрать действие над ней: скачать аудио, скачать видео, транскрибировать и подвести итог.

В заключение, диаграмма вариантов использования является важным инструментом в разработке программного средства. Она позволяет моделировать и описывать взаимодействие между актерами и самим программным средством, определять основные функциональные возможности и требования к системе, а также создавать документацию. Диаграмма вариантов использования представлена в графической части проекта МРКК.502902.001ПЛ.

При моделировании поведения системы возникает необходимость детализировать особенности алгоритмической и логической реализации выполняемых системой операций. Для моделирования процесса выполнения операций в языке UML используются так называемые диаграммы деятельности.

После запуска чат-бота, он ожидает получения сообщения. При получении команды /help, бот отправляет описание функций.

При получении сообщения вместе с командой /tts, определяется язык текста, генерируется речь на основе текста и отправляется полученное аудио. При получении сообщения вместе с командой /tti, генерируется изображение по описанию и полученное фото.

При получении команды /vta бот отправляет сообщение об готовности получить видео и переходит в режим ожидания mp4. Полученное видео преобразуется в аудио и отправляется. Если бот получает текстовое сообщение, он также генерирует и отправляет ответ в текстовом формате.

При получении изображения, происходит распознавание текста с фото и он отправляется к в качестве ответа вместе с кнопками «answer» и «summary». При нажатии пользователем на эти кнопки происходит отправка ответа или краткое содержание. Если бот получает аудио или видео, он скачивает данные, которые может транскрибировать или по речи сгенерировать краткое содержание в зависимости от выбора пользователя. При получении YouTube-ссылки, бот может скачать и отправить видео или аудио, транскрибировать речь или подытожить речь.

В случае чат-бота, диаграмма деятельности помогает описать последовательность действий и взаимодействия с пользователем. Она позволяет наглядно представить, как бот обрабатывает полученные команды и сообщения, как осуществляется генерация речи, создание изображений, распознавание текста и другие функции. Диаграмма деятельности представлена в графической части проекта МРКК.502902.002ПЛ.

2.3 Выбор и обоснование инструментов разработки

Общая продуктивность любых инструментов создания программного обеспечения определяется следующими пятью важнейшими аспектами:

- качеством визуальной среды разработки;
- скоростью работы компилятора и быстродействием откомпилированных программ;
- мощностью языка программирования и его сложностью;
- гибкостью и масштабируемостью используемой архитектуры баз данных;
- наличием поддерживаемых средой разработки шаблонов проектирования и использования.

C# – это объектно-ориентированный язык программирования для создания приложений, работающих в среде .NET Framework, а также для разработки веб-приложений, игр и мобильных приложений [1].

C# поддерживает полноценное объектно-ориентированное программирование и строгую типизацию, что обеспечивает более высокую степень безопасности и надежности программного кода.

C# может быть использован для создания Telegram-ботов, используя библиотеки, такие как Telegram.Bot, которые облегчают работу с Telegram API. Боты могут выполнять широкий спектр задач, таких как автоматизация работы с данными, оповещения, игры и многое другое.

Достоинства C#:

- простота и легкость в изучении: C# имеет простой синтаксис и легко понимается даже новичками;
- объектно-ориентированность: C# поддерживает полноценное объектно-ориентированное программирование, что делает код более модульным и удобным в сопровождении;
- сильная типизация: C# предлагает строгую типизацию, что обеспечивает более высокую степень безопасности и надежности программного кода;
- интеграция с .NET Framework: C# интегрируется с платформой .NET Framework, что обеспечивает широкие возможности для разработки и упрощает работу с библиотеками;
- поддержка многопоточности: C# поддерживает многопоточность, что позволяет создавать многопоточные приложения для повышения производительности.

Недостатки C#:

- ограниченность платформ: C# работает только на платформах Microsoft, что ограничивает его использование на других операционных системах;

- низкая скорость выполнения: при выполнении больших вычислительных задач C# может быть медленнее, чем другие языки программирования;
- необходимость установки среды разработки: для разработки приложений на C# необходимо установить специальную среду разработки, что может быть неудобным для новичков;
- ограничения при использовании памяти: C# использует управляемую память, что может привести к задержкам при выполнении программы и ограничивает ее производительность;
- ограниченный доступ к аппаратному обеспечению: C# имеет ограниченный доступ к аппаратному обеспечению, что ограничивает его использование в некоторых сферах, таких как например системное программирование.

Python, объектно-ориентированный, высокоуровневый язык программирования общего назначения с динамической строгой типизацией и автоматическим управлением памятью, был использован для перевода аудио в текст. Поддерживает императивное, процедурное, структурное, объектно-ориентированное программирование, метапрограммирование, функциональное программирование [6].

Python поддерживает кроссплатформенную совместимость; это означает, что разработанные на Python приложения могут работать на различных платформах, таких как Windows, MAC, Linux, Raspberry Pi и т.д.

Разработчики могут написать разумное программное приложение, написав несколько строк кода на Python. Поскольку Python работает в системе интерпретатора, код может быть выполнен сразу после написания, что делает его очень удобным для создания прототипов.

Преимущества Python:

- низкий порог вхождения;
- большое количество сред разработки;
- гибкость;
- расширяемость;
- простота синтаксиса;
- интерпретируемость.

Недостатки:

- язык бесплатен только для небольших фирм, индивидуальных программистов, стартапов и учащихся.

Visual Studio – это интегрированная среда разработки (IDE) от компании Microsoft, которая позволяет разрабатывать приложения для различных платформ, таких как Windows, Android, iOS, macOS и многих других.

Visual Studio является одним из самых популярных IDE для разработки приложений на языке C# и Python.

Для разработки на C# в Visual Studio необходимо установить пакет .NET Framework или .NET Core, которые обеспечивают среду выполнения и библиотеки для языка C#. Visual Studio имеет встроенный дизайнер форм и инструменты для создания графических интерфейсов, что делает разработку приложений на C# более удобной и эффективной.

Достоинства Visual Studio:

- интуитивно понятный интерфейс: Visual Studio имеет интуитивно понятный интерфейс, который облегчает работу с IDE и ускоряет процесс разработки;
- обширные функциональные возможности: Visual Studio имеет широкий спектр инструментов для разработки приложений, таких как инструменты для отладки, автоматической сборки, анализа кода, создания интерфейсов, управления проектами и многие другие;
- поддержка различных языков программирования: Visual Studio поддерживает большое количество языков программирования, таких как C++, C#, Python, Java, JavaScript и многие другие;
- расширяемость: в Visual Studio можно устанавливать и использовать сторонние плагины и расширения, которые расширяют функциональные возможности IDE;
- поддержка командной разработки: Visual Studio обеспечивает возможность коллективной работы над проектами, позволяя использовать системы контроля версий, такие как Git, SVN и другие.

Недостатки Visual Studio:

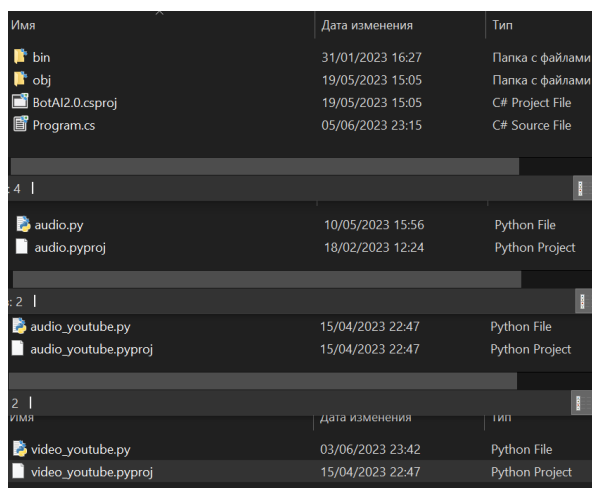
- большой размер: Visual Studio имеет большой размер и потребляет много ресурсов, что может замедлить работу на более старых компьютерах;
- высокая стоимость: полная версия Visual Studio является платной, что может быть недоступно для многих разработчиков;
- сложность настройки: некоторые функции IDE могут быть сложны для настройки и использования, особенно для новичков.

3 ПРОГРАММНАЯ РЕАЛИЗАЦИЯ

3.1 Физическая структура

Физическая структура программного обеспечения представляет собой систему каталогов, расположенную на локальной Windows-управляемой машине.

Код разрабатываемого программного средства «BotAI» располагается в файлах cs и py. Все папки и файлы отображены на рисунке 3.1.



Имя	Дата изменения	Тип
bin	31/01/2023 16:27	Папка с файлами
obj	19/05/2023 15:05	Папка с файлами
BotAI2.0.csproj	19/05/2023 15:05	C# Project File
Program.cs	05/06/2023 23:15	C# Source File
4		
audio.py	10/05/2023 15:56	Python File
audio.pyproj	18/02/2023 12:24	Python Project
2		
audio_youtube.py	15/04/2023 22:47	Python File
audio_youtube.pyproj	15/04/2023 22:47	Python Project
2		
Имя	Дата изменения	Тип
video_youtube.py	03/06/2023 23:42	Python File
video_youtube.pyproj	15/04/2023 22:47	Python Project

Рисунок 3.1 – Файлы BotAI

Файл CS – это файл исходного кода, написанный на C#.

Файл PY – это программный файл или сценарий, написанный на Python, интерпретируемом объектно-ориентированном языке программирования. Он может быть создан и отредактирован с помощью текстового редактора, но для его запуска требуется интерпретатор Python. Файлы PY часто используются для программирования веб-серверов и других административных компьютерных систем.

Открывать и редактировать сценарии PY можно с помощью любого текстового редактора или редактора исходного кода. Редакторы исходного кода предоставляют полезные инструменты подсветки синтаксиса и редактирования кода, которые облегчают просмотр и редактирование сценариев Python.

В исходном коде на C# присутствуют зависимости от различных библиотек, таких как системные библиотеки .NET, библиотеки для работы с Telegram API, библиотеки для обработки звука и изображений, а также библиотеки для работ с текстом и исключениями.

Исходный код на Python также имеет зависимости, включая библиотеки для работы с файловой системой, искусственным интеллектом и скачивания видео с YouTube.

3.2 Описание разработанных модулей

Код реализует Telegram-бота с широким спектром функций, таких как обработка текстовых сообщений, ссылок на YouTube видео, аудио- и видео-сообщений, а также изображений. Бот способен отвечать на различные команды, преобразовывать текст в речь и изображения, а также выполнять транскрипцию аудио и видео и получение краткого содержания текста.

Он использует внешние инструменты и сервисы, такие как FFmpeg, API VoiceRSS, API Stable Diffusion, OpenAI GPT, Whisper и IronOCR, YouTube DLP, API Detect Language для выполнения специфических задач, а также API Telegram. Бот обрабатывает ошибки и информирует пользователей об ошибках, которые могут возникнуть в процессе обработки сообщений.

Класс Program представляет программную реализацию для запуска Telegram-бота с использованием Telegram Bot API [8]. В основном методе Main создается экземпляр класса TelegramBotClient и передается токен Telegram-бота в качестве аргумента конструктора.

Создается экземпляр класса MyBot, который представляет реализацию бота. Этот объект будет использоваться для обработки обновлений от Telegram и выполнения соответствующих действий.

Вызывается метод StartReceiving у клиента TelegramBotClient, который запускает процесс получения и обработки обновлений от Telegram. В качестве аргументов передаются методы a.Update и a.Error, которые определены в классе MyBot и обрабатывают соответственно полученные обновления и ошибки.

После запуска процесса получения обновлений бот ожидает ввода строки с помощью метода Console.ReadLine(). Это позволяет боту оставаться активным и принимать обновления, пока не будет введена команда для остановки.

Класс MyBot содержит обработку сообщений в боте и выполнение соответствующих действий в зависимости от содержания сообщения.

Если текст сообщения равен «/start», отправляется приветственное сообщение и возвращается из метода.

Если текст сообщения равен «/help», отправляется описание функций бота и возвращается из метода.

Когда пользователь отправляет ссылку на YouTube видео, проверяется наличие ссылки в сообщении и ее сохранение в переменную link. Затем создается

инлайн-клавиатура с несколькими кнопками и пользователю отправляется сообщение с этой клавиатурой.

При обработке нажатия на кнопку выполняются определенные действия, такие как загрузка и отправка видео или аудио, транскрипция аудио в текст, а также получение краткого содержания текста. Для выполнения этих действий используются внешние скрипты на языке Python и библиотека `yt_dlp`.

Команда `«/tts»` предназначена для преобразования текста в речь (Text-to-Speech). Если сообщение содержит только `«/tts»` без дополнительного текста, отправляется информационное сообщение с инструкцией. В противном случае извлекается текст из сообщения путем удаления команды `«/tts»`.

Затем происходит определение языка текста с использованием API Detect Language для определения языка. Полученный язык используется для установки параметров речи, таких как кодек аудио, формат аудио, скорость и т.д. [2].

Далее инициализируется объект `VoiceProvider`, который отвечает за генерацию речи на основе текста с использованием API VoiceRSS [9]. Устанавливаются обработчики событий `SpeechFailed` и `SpeechReady`. При готовности речи происходит сохранение аудиофайла в формате MP3.

Затем размер аудиофайла проверяется на соответствие политике Telegram, и если размер не превышает 50 МБ, файл отправляется пользователю в чате. В противном случае отправляется сообщение о том, что Telegram не позволяет отправлять файлы размером более 50 МБ.

Команда `«/ti»` необходима для преобразования текста в изображение (Text-to-Image). Если сообщение содержит только `«/ti»` без дополнительного текста, отправляется информационное сообщение с инструкцией. В противном случае извлекается текст из сообщения путем удаления команды `«/ti»`.

Затем вызывается метод `TextToImage`, который принимает текст и выполняет преобразование в изображение. Внутри метода выполняется HTTP POST-запрос к удаленному API сервису Stable Diffusion для создания изображения на основе текста [7]. В запросе передаются API-ключ, параметры изображения и текст, который будет использоваться для создания изображения.

Полученный ответ в формате JSON распарсивается, и извлекается поле `«output»`, содержащее ссылку на сгенерированное изображение. Затем изображение скачивается по этой ссылке и отправляется пользователю.

Если процесс генерации изображения прошел успешно и файл сохранен, то файл отправляется пользователю в чате с помощью метода `SendPhotoAsync`. В противном случае отправляется сообщение о неудаче в генерации изображения.

Если сообщение содержит `«/vta»`, отправляется информационное сообщение с инструкцией действий для преобразования видео в аудио и бот ожидает получение видео (`BotState.IsWaitingForVideo(message.Chat.Id)` возвращает `true`). Если

сообщение содержит видео (`message.Video != null`), выполняется процесс преобразования видео в аудио.

Сначала получается информация о файле видео (`file`) с использованием метода `GetFileAsync`. Затем проверяется размер файла видео, и если он не превышает 20 МБ, видео загружается с помощью метода `DownloadFileAsync` в указанное место на локальном диске. Большие видео невозможно обрабатывать из-за ограничений Telegram для ботов без передачи данных на сервер Telegram.

После успешной загрузки видео файл конвертируется в аудио с использованием FFmpeg [3]. Создается новый процесс `ffmpegProcess`, который запускает команду FFmpeg для конвертации видео в аудио. Результат отправляется пользователю.

В классе, реализующем ожидание ботом видео-сообщения определены два статических метода и одно статическое поле.

Метод `IsWaitingForVideo` принимает идентификатор пользователя (`userId`) и проверяет, ожидает ли бот видео от данного пользователя. Возвращает `true`, если пользователь находится в состоянии ожидания видео, и `false` в противном случае.

Метод `SetWaitingForVideo` принимает идентификатор пользователя (`userId`) и флаг состояния ожидания видео (`isWaiting`). Он устанавливает состояние ожидания видео для данного пользователя. Если пользователь уже существует в словаре `waitingForVideoStates`, то его состояние обновляется согласно переданному флагу. Если пользователь не существует в словаре, то он добавляется вместе с переданным флагом состояния.

Статическое поле `waitingForVideoStates` представляет собой словарь, где ключом является идентификатор пользователя (`userId`), а значением – флаг состояния ожидания видео (`isWaiting`). В этом словаре хранится состояние ожидания видео для каждого пользователя.

Текстовые сообщения, отправленные пользователем обрабатываются при помощи API OpenAI [5]. Таким образом бот отвечает на любые текстовые вопросы.

Если приходит фото, то бот сохраняет его, отправляет на обработку OCR (оптическое распознавание символов) и получает текстовый результат. Функция OCR выполняет распознавание текста на изображении с помощью библиотеки IronOCR [4].

Если результат пустой или равен `null`, бот отправляет соответствующее сообщение. Если результат не пустой, бот отправляет его пользователю с кнопками «answer» и «summary».

Если пользователь нажимает кнопку «answer», бот вызывает функцию `callOpenAI`, передает ей текст и получает ответ от модели OpenAI. Если есть стикер, бот отправляет его с ответом вместе. Затем бот отправляет ответ пользователю.

Если пользователь нажимает кнопку «summary», бот добавляет текст вопроса к оригинальному тексту и вызывает функцию callOpenAI для получения краткого ответа. Затем бот отправляет краткий ответ пользователю.

Аудио- и видео-сообщения обрабатываются одинаково. Бот также проверяет размер файла. Если размер файла меньше или равен 20 МБ, бот отправляет пользователю сообщение о начале обработки и вызывает функцию ConvertToText, которая конвертирует аудио или видео в текстовый формат. Затем бот отправляет полученный текст пользователю с кнопкой «summary», благодаря которой все сказанное можно сократить и подытожить. Результат сохраняется в переменной для дальнейшего использования.

Обработку аудио и видео выполняет Python фрагмент. Он использует библиотеку OpenAI Whisper для преобразования аудио и видео в текст. Файл передается в функцию process_content, где происходит вызов API OpenAI для транскрибации аудио.

Если формат сообщения неизвестен (не является текстом, фотографией, видео или аудио), то бот отправляет пользователю стикер с сообщением «I don't know this format» и выводит в консоль соответствующее сообщение. В случае ошибки при выполнении этого действия, бот отправляет пользователю сообщение об ошибке и выводит информацию об ошибке в консоль.

В заключение, описанные методы демонстрирует реализацию гибкого и функционального Telegram-бота, способного выполнять различные задачи, такие как преобразование текста в речь и изображения, транскрипцию аудио и видео, получение краткого содержания текста. Бот понимает различные типы сообщений, включая текстовые сообщения, ссылки на YouTube видео, аудио- и видео-сообщения, а также изображения. Он также обрабатывает возможные ошибки и информирует пользователей о них в процессе обработки сообщений. Реализация может быть доработана и расширена в соответствии с конкретными потребностями. Он демонстрирует использование Telegram Bot API и различных внешних сервисов для создания ботов с расширенным функционалом.

4 ТЕСТИРОВАНИЕ

Тестирование программного обеспечения – это метод проверки соответствия фактического программного продукта ожидаемым требованиям и обеспечения отсутствия дефектов в программном продукте. Оно включает в себя выполнение компонентов программного обеспечения/системы с использованием ручных или автоматизированных инструментов для оценки одного или нескольких интересующих свойств. Целью тестирования программного обеспечения является выявление ошибок, пробелов или отсутствующих требований в отличие от фактических требований.

Таблица 4.1 – Результаты тестирования

Название операции	Входной параметр	Ожидаемый результат	Фактический результат
Команда «/start»	«/start»	Отправка сообщения с текстом «Hi, this is an AI chat room. You can ask me any question, and I can work with some graphic and audio information」 ☛_☛ \」\r\n For more details about my features call /help from the menu.»	Бот отправляет сообщение с текстом «Hi, this is an AI chat room. You can ask me any question, and I can work with some graphic and audio information」 ☛_☛ \」\r\n For more details about my features call /help from the menu.»
Команда «/help»	«/help»	Отправка сообщения с текстом, описывающим возможности бота	Бот отправляет сообщение с описанием возможностей бота
Обработка команды «/tts»	«/tts Hello, world!»	Отправка аудио-файла с голосовым сообщением «Hello, world!»	Бот отправляет аудио-файл с голосовым сообщением «Hello, world!»

Продолжение таблицы 4.1

Название операции	Входной параметр	Ожидаемый результат	Фактический результат
Обработка YouTube-ссылки	YouTube-ссылка	Отправка сообщения с кнопками «download video», «download audio», «transcribe», «summarize»	Бот отправляет сообщение с кнопками «download video», «download audio», «transcribe», «summarize»
Команда «/tti»	«/tti Convert this text to an image»	Отправка изображения, соответствующего тексту «Convert this text to an image»	Бот отправляет изображение, соответствующее тексту «Convert this text to an image»
Обработка видео-файла для конвертации в аудио	«/vta», видео-файл (формат mp4)	Отправка аудио-файла, полученного из видео	Бот отправляет аудио из видео
Транскрибация аудио	Аудио-файл (формат mp3), выбор «transcribe»	Отправка текстового сообщения с транскрипцией аудио	Бот отправляет текстовое сообщение с транскрипцией аудио
Суммаризация видео	Видео-файл (формат mp4), выбор «summary»	Отправка текстового сообщения с кратким обобщением текста	Бот отправляет текстовое сообщение с кратким обобщением текста
Работа с YouTube-ссылкой	«download audio», «download video», «transcribe», «summarize»	Отправка видео, отправка аудио, текстовое сообщение с транскрипцией видео, текстовое сообщение с пересказом видео	Бот отправляет видео, аудио, текстовое сообщение с транскрипцией видео, текстовое сообщение с пересказом видео

Продолжение таблицы 4.1

Название операции	Входной параметр	Ожидаемый результат	Фактический результат
Обработка фото	Фото с текстом	Отправка текста с фотографии	Бот отправляет текст с фотографии
Работа с фото	«answer», «summary»	Отправка ответа на текст с фотографии, подытоживание текста с фото	Бот отправляет ответ на текст с фотографии, подытоживает текст с фото
Отправка эмоджи	Эмоджи	Отправка текстового сообщения	Бот отправляет ответ текстом
Отправка стикера	Стикер	Отправка стикера и текстового сообщения «I don't know this format»	Бот отправляет стикер и текстовое сообщение «I don't know this format»
Обработка поврежденного видео-файла	Поврежденный видео-файл (формат mp4)	Сообщение «Sorry, an error occurred while processing your request.»	Бот отправляет сообщение Сообщение «Sorry, an error occurred while processing your request.»
Отправка фото без текста	Фото без текста	Сообщение «Nothing on the image»	Бот отправляет сообщение «Nothing on the image»

В ходе тестирования программного обеспечения были проверены различные операции и их ожидаемые результаты. В целом, тестирование программного обеспечения было успешным, и большинство операций выполнились согласно ожиданиям. Однако, были выявлены некоторые ситуации, в которых бот сообщал о возникновении ошибок или отсутствии текста/информации. Таким образом, дальнейшее тестирование программного обеспечения позволило устранить эти ограничения и обеспечить более надежное и полноценное функционирование.

5 ПРИМЕНЕНИЕ ПРОГРАММЫ

5.1 Назначения и условия применения

Telegram-бот BotAI предназначен для обработки различных типов сообщений, включая текстовые сообщения, ссылки на YouTube видео, аудио- и видео-сообщения, а также изображения. Бот предоставляет широкий спектр функций, позволяющих пользователю взаимодействовать с разными медиа-форматами и получать соответствующие ответы и результаты обработки.

Для использования данного Telegram-бота пользователю необходимо удовлетворять следующим техническим требованиям:

1) Устройство: доступ к устройству, способному запустить приложение Telegram. Это может быть смартфон (Android, iOS) или компьютер (Windows, macOS, Linux).

2) Интернет-соединение: стабильное подключение к интернету для связи с серверами Telegram и обмена сообщениями с Telegram -ботом.

3) Установленное приложение Telegram: установить и настроить официальное приложение Telegram на устройстве. Оно доступно для скачивания из соответствующих магазинов приложений (Google Play Store, App Store, Windows Store) или с официального веб-сайта Telegram.

4) Аккаунт в Telegram: зарегистрировать аккаунт в Telegram, следуя процедуре регистрации, которая включает в себя ввод номера мобильного телефона и подтверждение кода.

5) Доступ к боту: возможность найти и получить доступ к Telegram -боту. Это может быть достигнуто путем поиска бота по его имени или уникальному идентификатору.

6) Знание команд: ознакомление с доступными командами и функциональностью Telegram -бота, чтобы правильно взаимодействовать с ним и получить желаемые результаты.

5.2 Руководство пользователя

Для начала работы с ботом требуется запустить приложение Telegram на устройстве. При необходимости ввести логин и пароль, чтобы войти в учетную запись Telegram.

После входа в приложение в главном окне отображается список чатов и контактов, а также поиск. Для использования телеграм-бота его необходимо найти его в списке контактов или использовать поиск на рисунке 5.1, чтобы найти бота по имени «@dfdvgmkfi_bot».

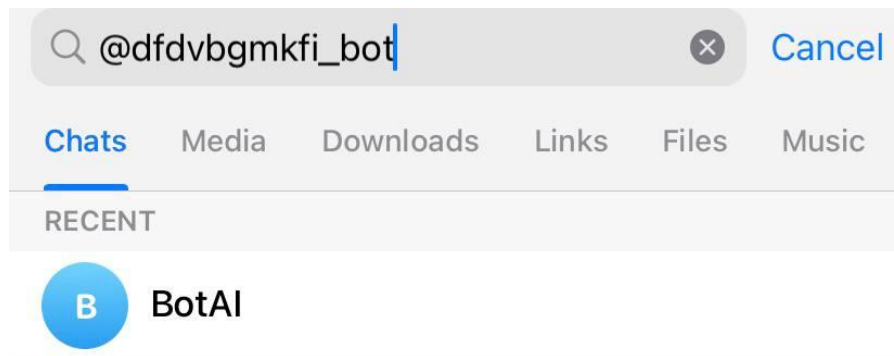


Рисунок 5.1 – Поиск бота

При первом общении с Telegram-ботом отправляется приветственное сообщение и инструкции о том, как пользоваться ботом. Для взаимодействия с Telegram-ботом можно использовать команды на рисунке 5.2, текстовые сообщения или другие доступные функции, предоставляемые ботом.

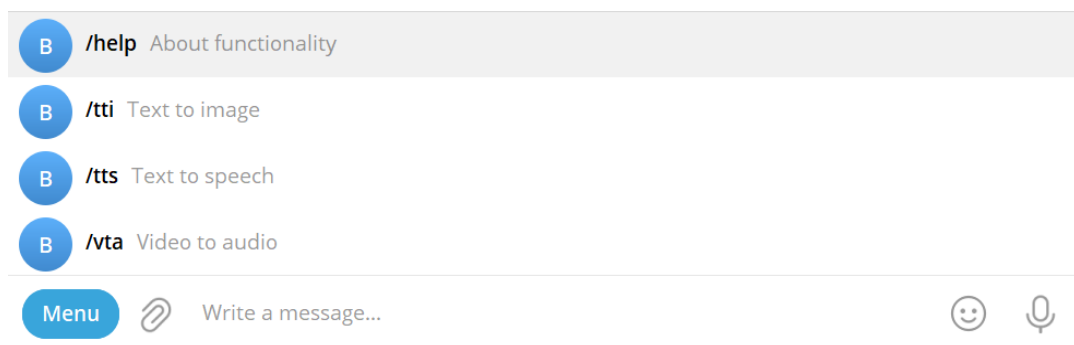


Рисунок 5.2 – Меню

Бот может предоставлять информацию, отвечать на запросы, выполнять определенные задачи или предлагать выбор из различных вариантов действий. Чтобы узнать все доступные функции, показанные на рисунке 5.3 и команды бота, можно использовать команду /help.

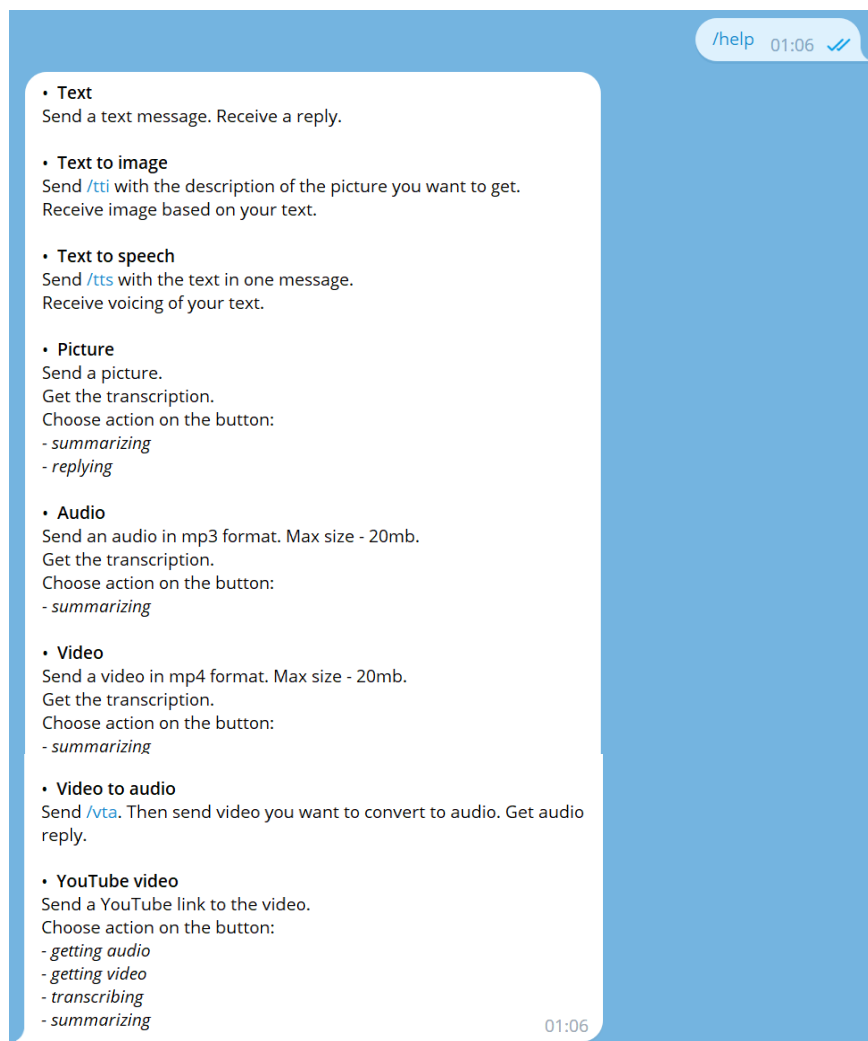


Рисунок 5.3 – Команда /help

Чтобы использовать функции без команд можно сразу отправлять файлы в соответствующих форматах, например фото, аудио, видео, ссылка на YouTube видео. Бот генерирует ответы на все текстовые сообщения, включая эмоджи. Для использования функций преобразования текста в речь или изображение, необходимо отправлять текст и команду в одном сообщении. Для использования функции преобразования видео в аудио, требуется сначала отправить команду /vta, а следующим сообщением видео.

При возникновении вопросов, получить помощь можно обратившись к разработчику в описании.

ЗАКЛЮЧЕНИЕ

В рамках проекта был разработан Telegram-бот с использованием языка программирования C# и искусственного интеллекта. Целью проекта было создание AI-бота, способного взаимодействовать с пользователями через мессенджер Telegram.

В процессе разработки были достигнуты следующие задачи: изучение основ работы Telegram-ботов и их интеграции с платформой Telegram, изучение основ искусственного интеллекта, проектирование архитектуры бота, реализация функциональности и тестирование.

Бот способен обрабатывать различные форматы сообщений, включая текстовые сообщения, фото, аудио и видео. Он использует API OpenAI ChatGPT для ответов на текстовые сообщения и преобразование речи в текст. Также бот осуществляет обработку фото с помощью OCR и предоставляет функции преобразования текста в изображение и речь.

В процессе тестирования была проверена правильность и работоспособность отдельных функций и программной системы в целом. Было обнаружено, что бот успешно выполняет свои задачи и обрабатывает различные форматы сообщений.

Результатом работы является готовое программное средство, способное эффективно взаимодействовать с пользователями через Telegram-мессенджер. Бот обладает потенциалом для применения в различных областях, где автоматизированные ответы и функции на основе искусственного интеллекта могут быть полезными и актуальными.

В процессе разработки проекта были использованы передовые технологии и методы искусственного интеллекта, что позволило расширить знания и навыки в этой области. Проект также дал возможность практически применить полученные знания и развить навыки программирования на языке C#.

В заключение, программное средство успешно реализует задачи и функции, поставленные перед ним, и предоставляет пользователям удобный способ использования различных видов искусственного интеллекта через Telegram. Дальнейшее развитие проекта может включать расширение функциональности бота, оптимизацию его работы и применение более сложных алгоритмов искусственного интеллекта для более точных и полезных ответов.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

- [1] C# Programming: A Step-by-step Guide For Absolute Beginners / Brian Jenkins, – Independently published, 2019 – 196 с.
- [2] Detect Language [Электронный ресурс]. – Режим доступа: <https://detectlanguage.com/> – Дата доступа: 14.05.2023.
- [3] FFmpeg documentation [Электронный ресурс]. – Режим доступа: <https://ffmpeg.org/documentation.html> – Дата доступа: 27.05.2023.
- [4] IronOCR [Электронный ресурс]. – Режим доступа: <https://ironsoftware.com> – Дата доступа: 02.03.2023.
- [5] OpenAI [Электронный ресурс]. – Режим доступа: <https://openai.com> – Дата доступа: 02.03.2023.
- [6] Python Artificial Intelligence Projects for Beginners: Get up and running with Artificial Intelligence using 8 smart and exciting AI applications / Joshua Eckroth, – Packt Publishing, 2018 – 162 с.
- [7] Stable Diffusion [Электронный ресурс]. – Режим доступа: <https://stability.ai/> – Дата доступа: 12.04.2023.
- [8] Telegram Bots Book [Электронный ресурс]. – Режим доступа: <https://telegrambots.github.io/book/> – Дата доступа: 04.06.2023.
- [9] VoiceRSS [Электронный ресурс]. – Режим доступа: <https://voicerss.org/api/> – Дата доступа: 15.05.2023.

ПРИЛОЖЕНИЕ А
(обязательное)
Результат работы программного средства

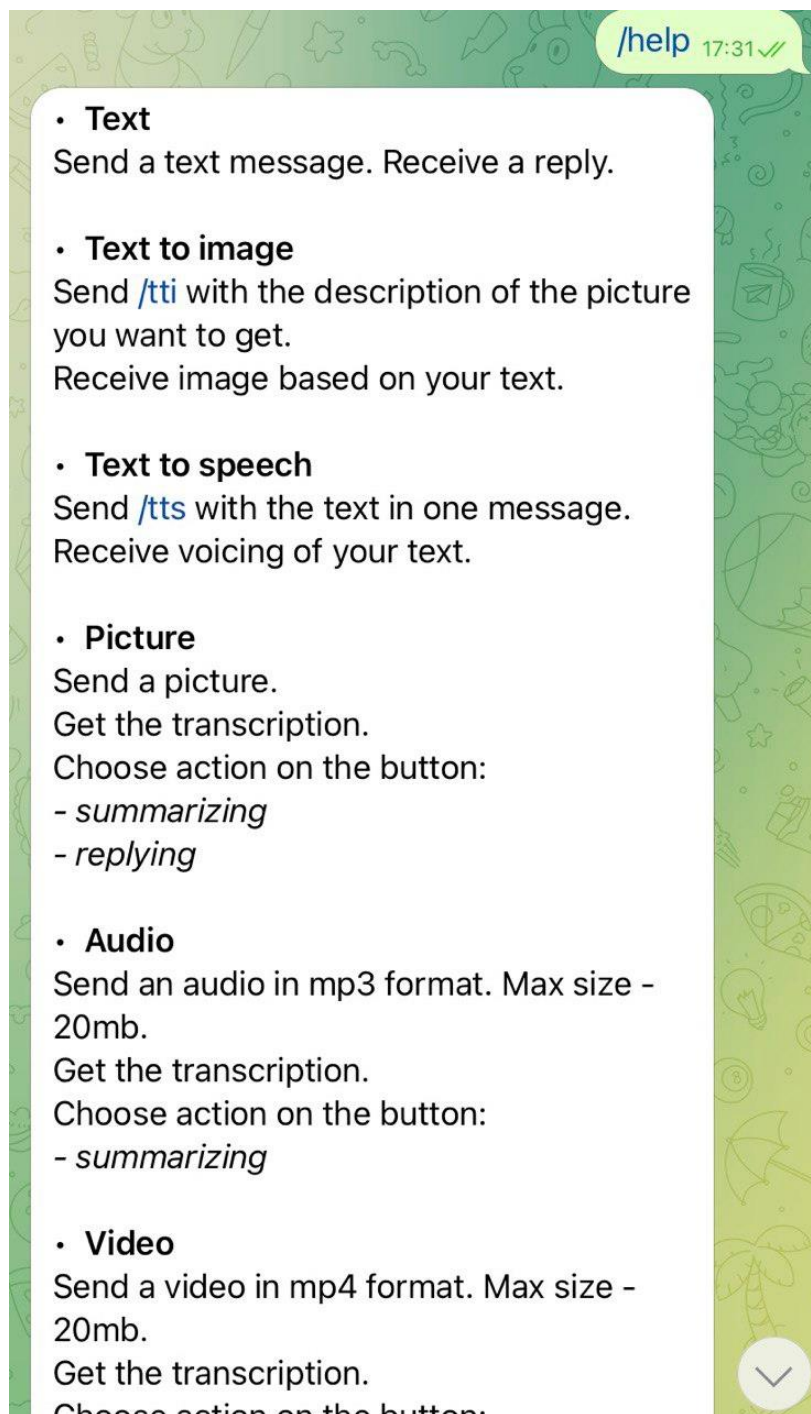


Рисунок А1 – Команда /help

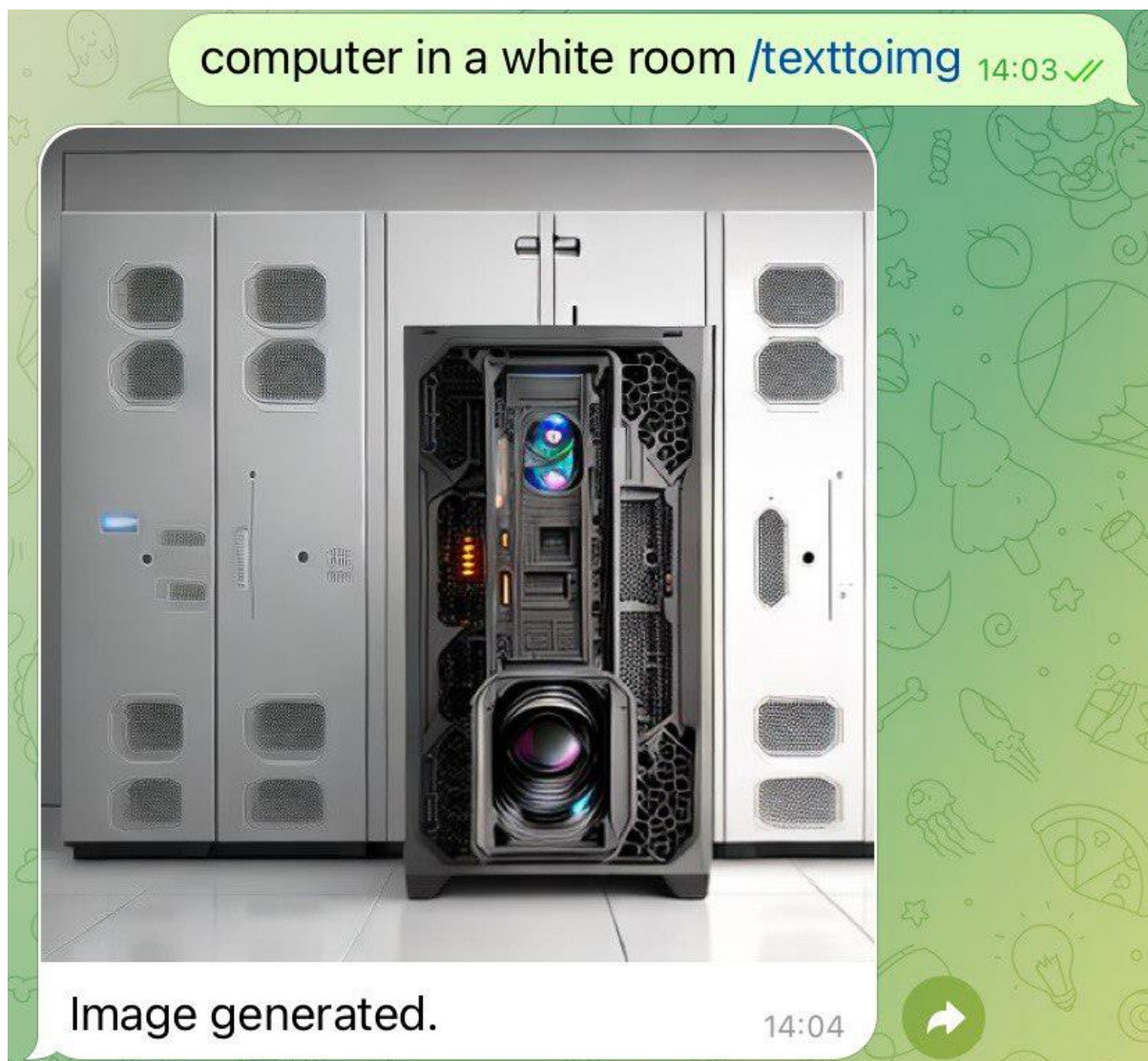


Рисунок А2 – Команда «Text to image»

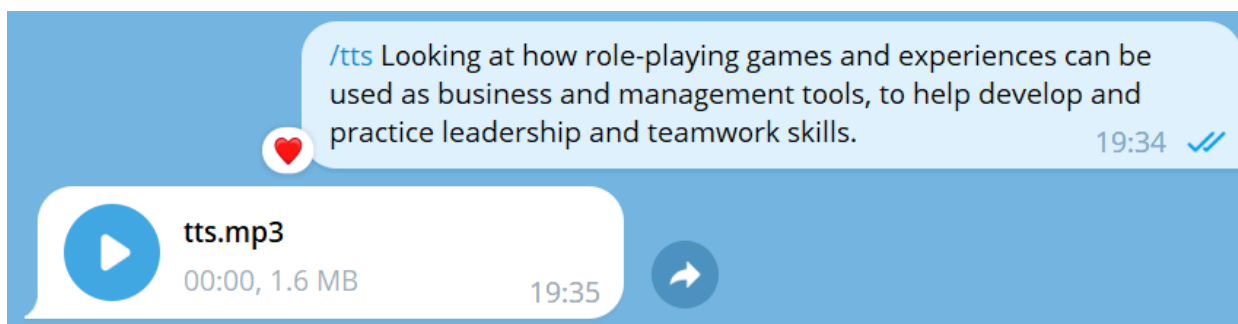


Рисунок А3 – Команда «Text to speech»

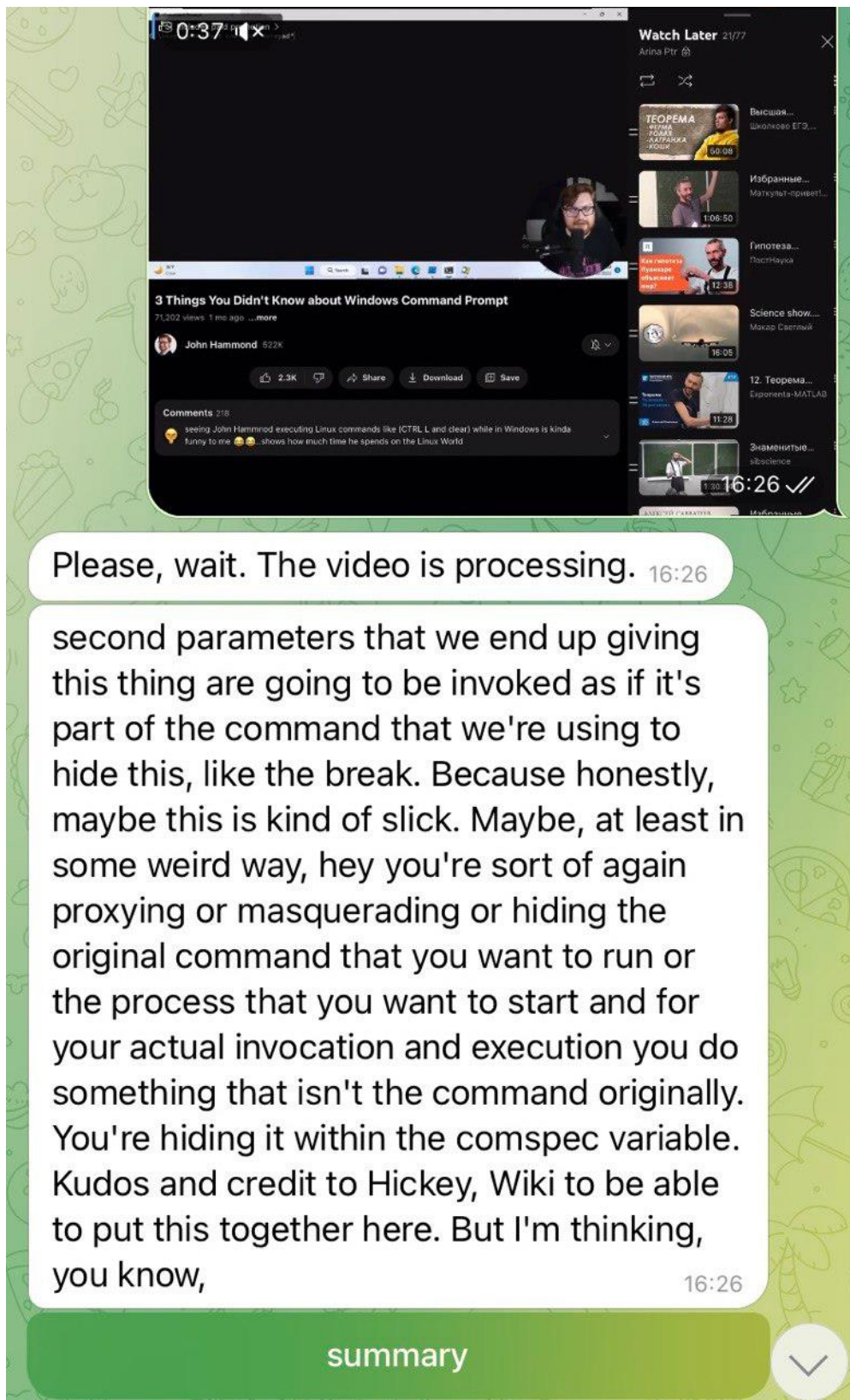


Рисунок А4 – Распознавание речи

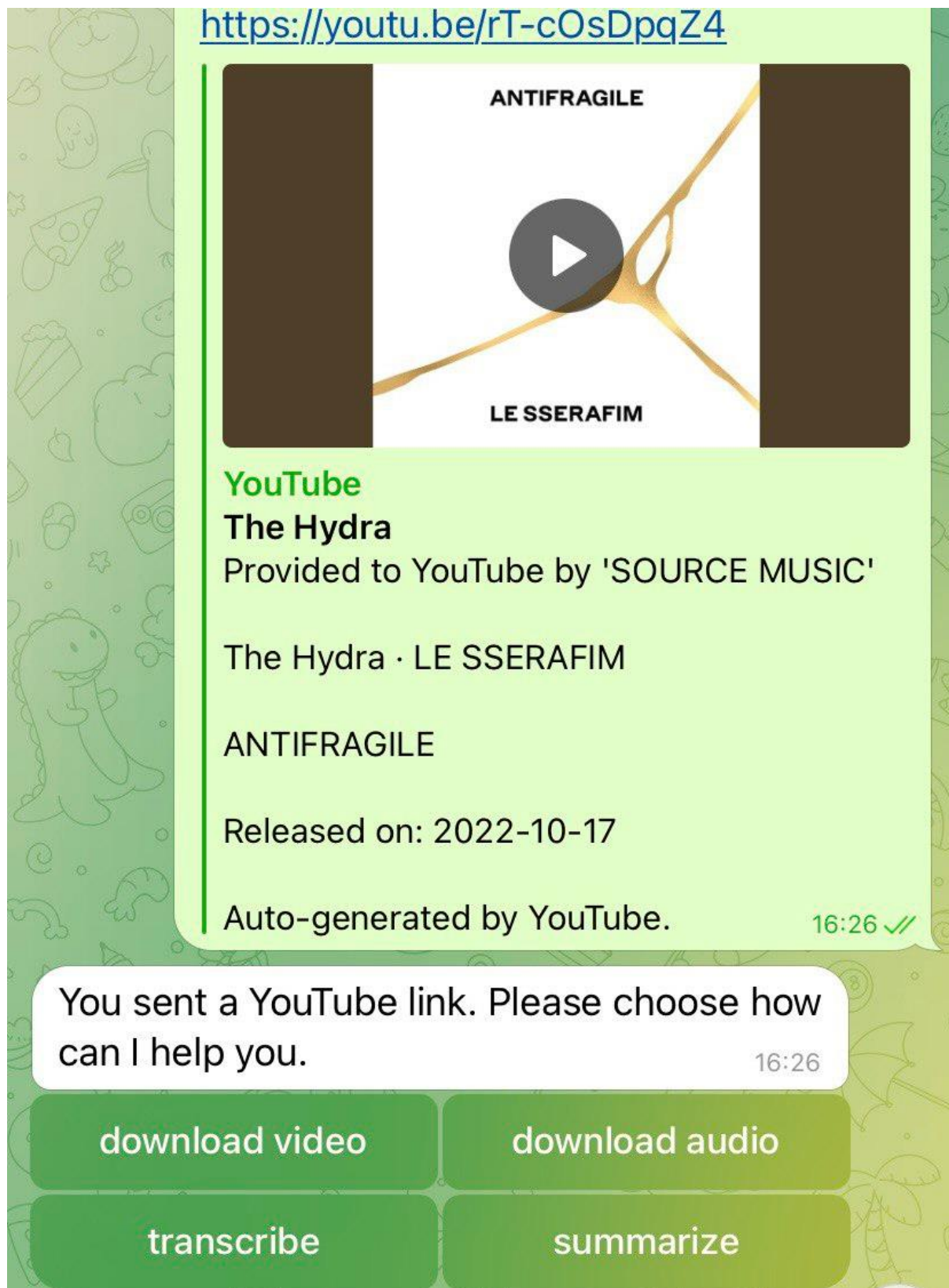


Рисунок А5 – Работа с YouTube-ссылкой

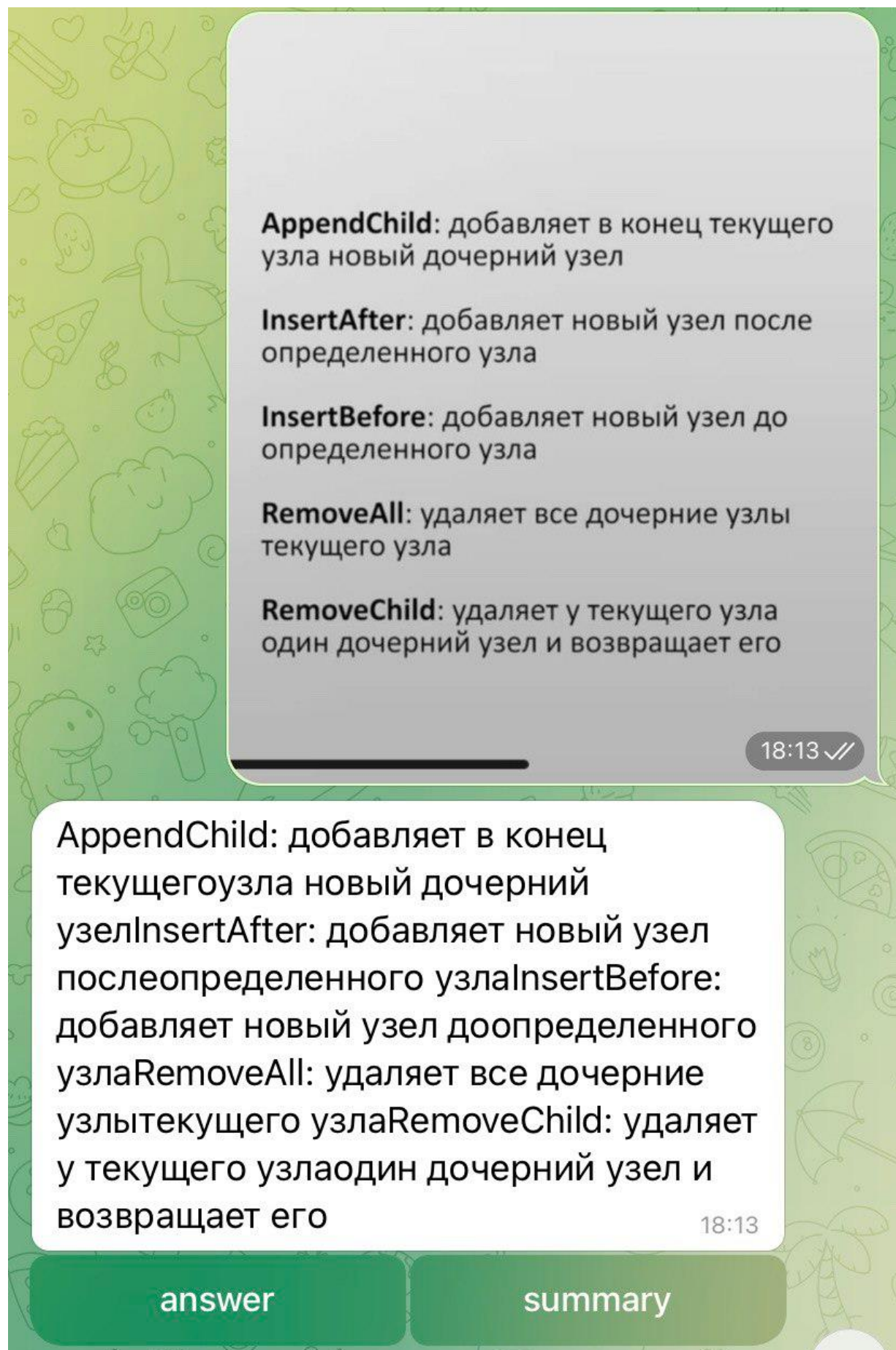


Рисунок А6 – Распознавание текста на изображении

напиши полный код hello world на c#

16:24 ✓✓

```
using System;

namespace HelloWorld
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Hello World!");
            Console.ReadKey();
        }
    }
}
```

16:24

Рисунок А7 – Общение с искусственным интеллектом

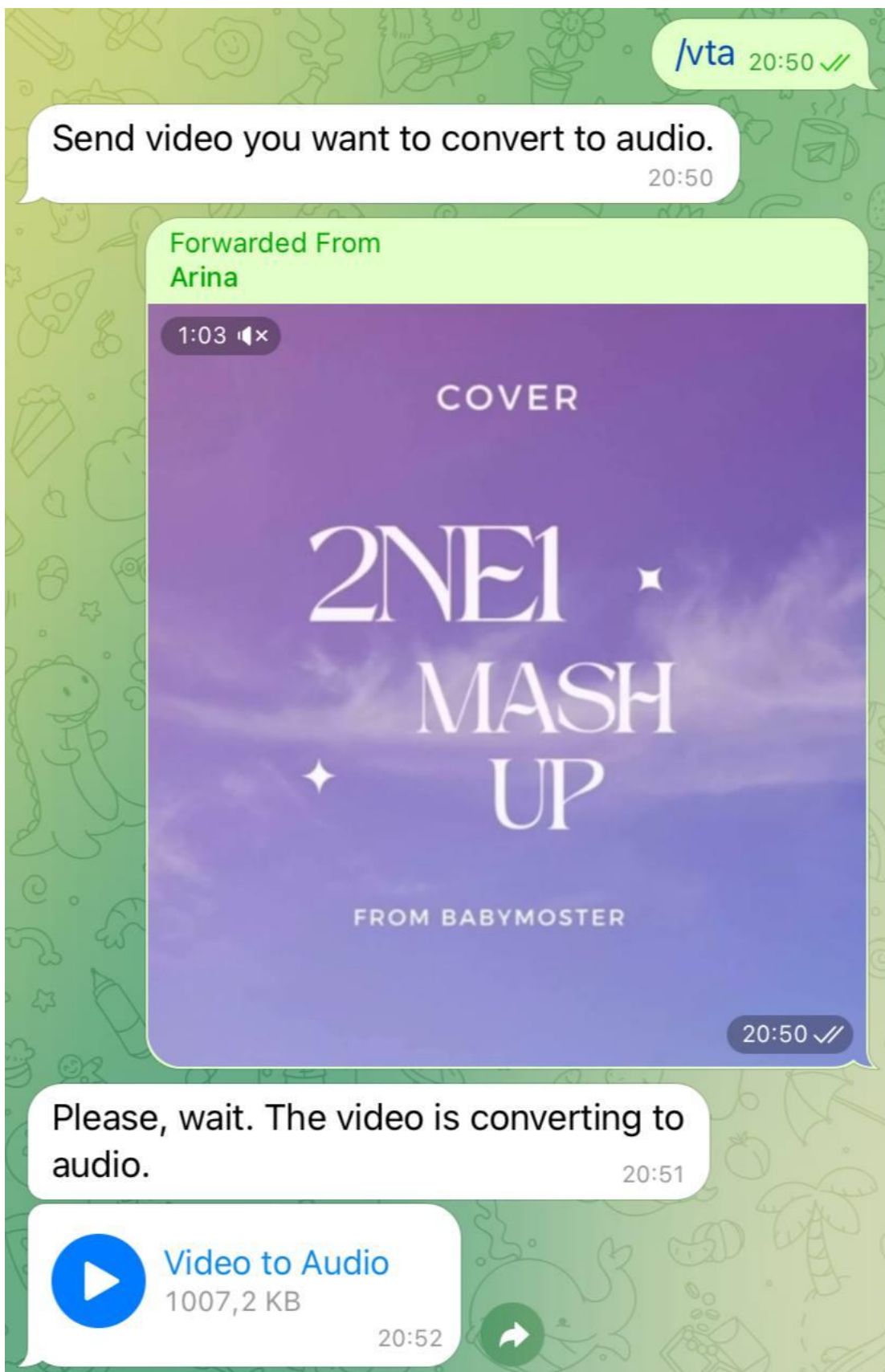


Рисунок А8 – Команда `/vta`