МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

Казахский национальный исследовательский технический университет имени К.И.Сатпаева

Институт автоматики и информационных технологий

Кафедра «Программная инженерия»

Бородкин Константин Евгеньевич

Создание системы ChatBot для автоматизации взаимоотношений «эдвайзер-студент» в университете

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
к дипломному проекту

5В070400 – Вычислительная техника и программное обеспечение

Алматы 2022

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

Казахский национальный исследовательский технический университет имени К.И.Сатпаева

Институт автоматики и информационных технологий

Кафедра «Программная инженерия»

ДОПУЩЕН К ЗАЩИТЕ
Заведующая кафедрой ПИ
канд. физ-мат. наук, профессор
_____ А.Н. Молдагулова
«_20_» ___05___ 2022 г.

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к дипломному проекту

На тему: «Создание системы Chatbot для автоматизации взаимоотношений «эдвайзер-студент» в университете»

по специальности 5В070400 – Вычислительная техника и программное обеспечение

Выполнил Бородкин К.Е.

Научный руководитель
Доктор PhD, сениор лектор,
_____ М. Сатымбеков
«_20_» ___05___ 2022 г.

Рецензент
Доктор PhD,
и. о. ассоциированного профессора
_____ Н.О. Мекебаев
«_____» _____ 2022 г.

Алматы 2022

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

Казахский национальный исследовательский технический университет имени К.И.Сатпаева

Институт автоматики и информационных технологий

Кафедра "Программная инженерия"

5В070400 – Вычислительная техника и программное обеспечение

УТВЕРЖДАЮ
Заведующая кафедрой ПИ
канд. физ-мат. наук, профессор
_____ А.Н. Молдагулова
«20» 05 2022 г.

ЗАДАНИЕ
на выполнение дипломного проекта

Обучающемуся *Бородкину Константину Евгеньевичу*
Тема: *Создание системы ChatBot для автоматизации взаимоотношений «эдвайзер-студент» в университете*
Утверждена *приказом проректора по академической работе* № 489-П/О от «24» 12 2021 г.
Срок сдачи законченного проекта «25» 05 2022 г.

Исходные данные к дипломному проекту: *техническое задание, описание БД в виде ER-диаграммы, описание необходимых функций проекта.*
Перечень подлежащих разработке в дипломном проекте вопросов:
а) разработка базы данных;
б) реализация функций для взаимодействия пользователя с ботом;
в) разделение пользователей на роли: эдвайзер и студент;
г) реалиация функций для работы с новостями;
д) создание возможности адаптации бота к конкретному студенту.
Перечень графического материала (с точным указанием обязательных чертежей): *представлены 23 слайда презентации.*
Рекомендуемая основная литература: *из 21 наименования.*

ГРАФИК

подготовки дипломного проекта

Наименование разделов, перечень разрабатываемых вопросов	Сроки представления научному руководителю и консультантам	Примечание
1. Анализ предметной области, разработка технического задания	15.01.2022	Выполнено
2. Выбор технологий для разработки	20.01.2022	Выполнено
3. Разработка базы данных	25.01.2022	Выполнено
4. Разработки логики взаимодействия	30.01.2022	Выполнено
5. Разработка функционала системы	23.03.2022	Выполнено
6. Тестирование и оптимизация	15.04.2022	Выполнено
7. Написание пояснительной записки к дипломному проекту	28.04.2022	Выполнено

Подписи

консультантов и нормоконтролера на законченный дипломный проект с указанием относящихся к ним разделов проекта

Наименования разделов	Консультанты, И.О.Ф. (уч. степень, звание)	Дата подписания	Подпись
Нормоконтролер	Жекамбаева М.Н. Доктор Ph.D., ассоциированный профессор	20.05.22	
Программное обеспечение	Марғұлан Қ. Магистр тех. наук, лектор	18.05.22	

Научный руководитель _____ Сатымбеков М.

Задание принял к исполнению обучающийся _____ Бородкин К.Е.

Дата «17» 11 2021г.

АННОТАЦИЯ

Данный дипломный проект посвящен разработке чат-бота, используя возможности актуальных языков программирования и библиотек для них. Также в нем применяются принципы и простые алгоритмы, связанные с ИИ и МО.

Проблема в коммуникациях между студентом и эдвайзером (куратором) в настоящее время стоит очень остро, так как у куратора может быть большое количество студентов, в результате, из-за человеческого фактора, невозможно будет своевременно и оказать качественную информационную поддержку и ответить на возникающие вопросы при большом количестве вопросов от студентов. Этот бот призван облегчить взаимодействие между людьми, снизить нагрузку на эдвайзера, быстро и своевременно отвечать на вопросы студентов и доставлять актуальную информацию, автоматизировать процессы, которые возможно выполнять без помощи человека.

Для разработки данного проекта используется язык программирования Python и его специальный фреймворк Aiogram в части, ответственной за взаимодействие с мессенджером. Для хранения информации используются возможности реляционной базы данных SQLite, которая уже встроена в Python, также язык запросов SQL для написания запросов к БД.

АНДАТПА

Бұл дипломдық жоба олар үшін қазіргі бағдарламалау тілдері мен кітапханалардың мүмкіндіктерін қолдана отырып, чат-ботты дамытуға арналған. Сондай-ақ, ол ЖИ және МО-мен байланысты принциптер мен қарапайым алгоритмдерді қолданады.

Студент пен эдвайзер (куратор) арасындағы қарым-қатынастағы Проблема қазіргі уақытта өте өткір болып отыр, өйткені куратордың көптеген студенттері болуы мүмкін, нәтижесінде адам факторына байланысты уақтылы және сапалы ақпараттық қолдау көрсету және студенттерден көптеген сұрақтар туындаған кезде туындаған сұрақтарға жауап беру мүмкін болмайды. Бұл бот адамдар арасындағы өзара іс-қимылды жеңілдетуге, эдвайзерге жүктемені азайтуға, студенттердің сұрақтарына тез және уақтылы жауап беруге және өзекті ақпаратты жеткізуге, адамның көмегінсіз орындауға болатын процестерді автоматтандыруға арналған.

Бұл жобаны әзірлеу үшін Python бағдарламалау тілі және оның арнайы aiogram шеңбері мессенджермен өзара әрекеттесуге жауапты бөлігінде қолданылады. Ақпаратты сақтау үшін Python-да ендірілген SQLite реляциялық деректер базасының мүмкіндіктері, сонымен қатар мәліметтер базасына сұраныстар жазу үшін SQL сұрау тілі қолданылады.

ANNOTATION

This diploma project is dedicated to the development of a chatbot, using the capabilities of current programming languages and libraries for them. It also applies principles and simple algorithms related to AI and ML.

The problem in communication between the student and the adviser (curator) is currently very acute, since the curator may have a large number of students, as a result, due to the human factor, it will be impossible to provide timely and high-quality information support and answer questions with a large number of questions from students. This bot is designed to facilitate interaction between people, reduce the burden on the adviser, quickly and timely answer students' questions and deliver up-to-date information, automate processes that can be performed without human help.

To develop this project, the Python programming language and its special Aiogram framework are used in the part responsible for interacting with the messenger. To store information, the capabilities of the SQLite relational database are used, which is already built into Python, as well as the SQL query language for writing queries to the database.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Чат-бот – это специальная программа, которая может частично заменить возможности человека в области общения. Еще его можно назвать виртуальным собеседником, иногда выступает в качестве помощника. Программа создана в том числе и для того, чтобы помочь студенту в решении каких-либо проблем, связанных с учебным процессом или ответить на интересующие вопросы. Бот в автоматическом режиме общается с пользователем, отвечает на его сообщения, анализирует ответы.

В наше время бизнес, государственные компании, образовательные учреждения уделяют большое внимание разработке подобных ботов. Так как это надежный, выгодный и быстрый способ помочь клиентам и пользователям в решении проблем. Он позволяет автоматизировать часть процессов. Также чат-бот может заменить человека в решении простых рутинных задач, таких как ответы на вопросы.

С развитием машинного обучения и искусственного интеллекта чат-боты также получили новый толчок и с каждым годом перечень и сложность решаемых задач растут.

1 Исследовательский раздел

1.1 Цель разработки

Основной целью проекта является разработка современного, гибкого чат-бота для университета, который поможет автоматизировать и облегчить взаимодействие между эдвайзером и студентом посредством ответов на часто задаваемые вопросы, отправки новостей и взаимодействия с базой данных.

1.2 Термины и сокращения

Термины и сокращения, которые используется в пояснительной записке к дипломному проекту приведены ниже в таблице 1.1.

Таблица-1.1 – Термины и сокращения, а также их определения

Сокращение или термин	Определение
БД	База данных
СУБД	Система управления базами данных
API	(сокр. от англ. Application Programming Interface) программный интерфейс приложения. Совокупность инструментов для обеспечения взаимодействия между собой разных программ
Библиотека	В программировании – специальные программы и объекты, объединенные между собой для разработки других программ
NLP	(сокр. от англ. Natural Language Processing) обработка естественного языка
МО	Машинное обучение
ИИ	Искусственный интеллект
Фреймворк	Специальная платформа, которая описывает структуру системы, облегчает разработку и объединение разных компонентов программы в проекте.
UML	(сокр. от англ. Unified Modeling Language) унифицированный язык моделирования.
ER	(сокр. от англ. Entity Relationship) сущность-связь

Продолжение таблицы 1.1

Сокращение или термин	Определение
ИИН	Индивидуальный идентификационный номер.
FSM	(сокр. от англ. Finite-State Machine) конечные автоматы
ПО	Программное обеспечение
ОС	Операционная система

1.3 Актуальность проблемы на момент написания дипломной работы

Business Insider провели исследование, по результатам которого, рынок чат-ботов будет активно развиваться в ближайшее время [1], также это подтвердило еще одно исследование от Credence Research [2], которое ожидает экспоненциального роста рынка чат-ботов в период с 2021 до 2027 года. Всемирно известная компания Oracle проводила исследование, касательно отношения клиентов к бизнесу и процессу взаимодействия с различными системами и оказалось, что более 50% всех клиентов рассчитывают на то, что услуги будут доступны в любое время суток, каждый день и 65% пользователей предпочитают взаимодействие с компаниями через мессенджеры [3].

Проблема автоматизации взаимоотношений между студентами и их кураторами на данный момент стоит очень остро, так как большое количество людей поступают в высшие учебные заведения, выделяется большое число грантов каждый год, особенно на технические специальности. Также, основываясь на опыте, который мы получили во время пандемии вируса, когда обучение и большая часть любых процессов перешли на удаленный формат работы, нагрузка на эдвайзеров увеличилась, потому что у студентов появилось большое количество вопросов, связанных с новой формой обучения. Следовательно, кураторам стало сложнее быстро и качественно консультировать студентов по каким-либо вопросам.

Также у студентов практически нет единого источника достоверной и своевременной информации. Существует сложность, поиска интересующей и необходимой, в то же время актуальной информации касательно учебного процесса. Поэтому, студент вынужден обращаться напрямую к своему куратору, не тратив время на поиски нужной информации. Так, как у эдвайзера большое количество студентов, ответы каждому персонально занимают много времени. В данном случае чат бот поможет сэкономить большое количество времени и сил.

Чат-бот предоставляет доступ к информации в любое время, в любой день недели, достаточно только иметь подключение к интернету. Это помогает освободить нерабочее время эдвайзеров и сделать информацию еще более доступной.

Чаще всего студенты задают однотипные вопросы, на которые приходится отвечать персонально. Бот может обрабатывать это все автоматически и гораздо быстрее, не используя труд человека. Согласно исследованию от MIT Technology Review, 9 из 10 компаний отметили улучшение скорости обслуживания своих клиентов при помощи чат-ботов [4].

Бот облегчает студенческий быт, позволяя меньше сил тратить на поиски информации и больше вкладываться в процесс получения знаний. Студенты будут всегда в курсе всех предстоящих событий и новостей, что положительно может сказаться на их мотивации учиться и на успеваемости.

Бот собирает большое количество информации от студентов, которую потом можно анализировать и использовать для дальнейшего улучшения системы.

Также, эдвайзерам порой необходимо своевременно доводить до студентов какую-либо информацию. Данную задачу также можно упростить при помощи использования чат-бота.

Современные компании видят большое будущее у данной технологии, это подтверждает еще одно исследование от Business Insider, согласно которому, 80% предпринимателей из различных сфер бизнеса заинтересованы в том, чтобы внедрить чат-ботов в их существующие процессы [5].

1.4 Анализ аналогичных систем

На момент начала работы над данным проектом, у Satbayev University не было аналогичных систем. Однако, в процессе разработки, появились два чат-бота в Телеграм, которые имеют схожий функционал.

Первый чат-бот можно найти по нику @su_help_bot. После выбора языка он предлагает пользователю клавиатуру с вопросами, на которые можно получить ответ.

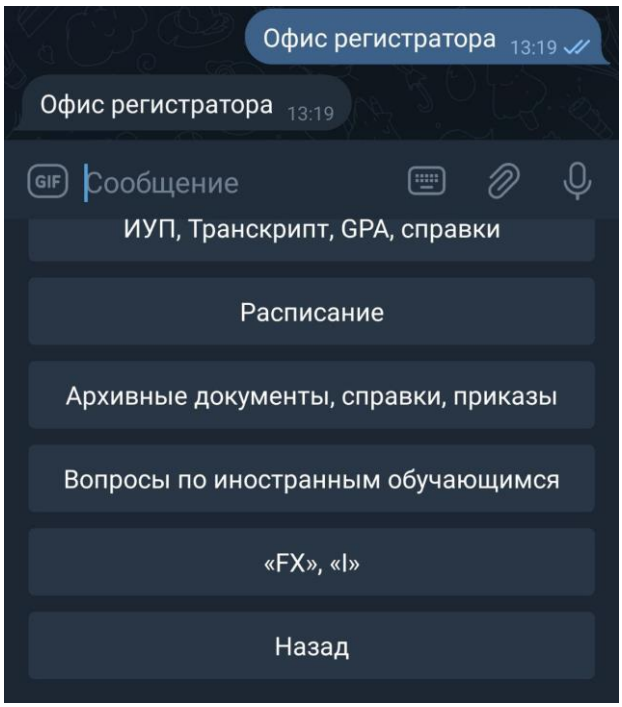

Рисунок-1.1 – Меню чат-бота @su_help_bot

При этом, навигация по данному меню вопросов занимает достаточно большое количество времени, так как большое количество ответов. Это немного ухудшает опыт взаимодействия с ботом.

Также есть возможность задать вопрос в свободной форме, но при этом отвечает студенту уже человек. То есть, сообщение в конечном счете получает какой-либо уполномоченный пользователь системы и после этого отвечает студенту. Это происходит не мгновенно и скорее всего зависит от нагрузки на операторов.

Любые другие типы сообщений никак не обрабатываются ботом.

Следующий бот, который был также создан в Satbayev University - @su_it_institute_bot. При начале работы, бот уточняет информацию о студенте, тип его обучения, институт, кафедру и специальность. После этого, студент может также получить ответы на вопросы, которые ему подаются в виде кнопок от бота.

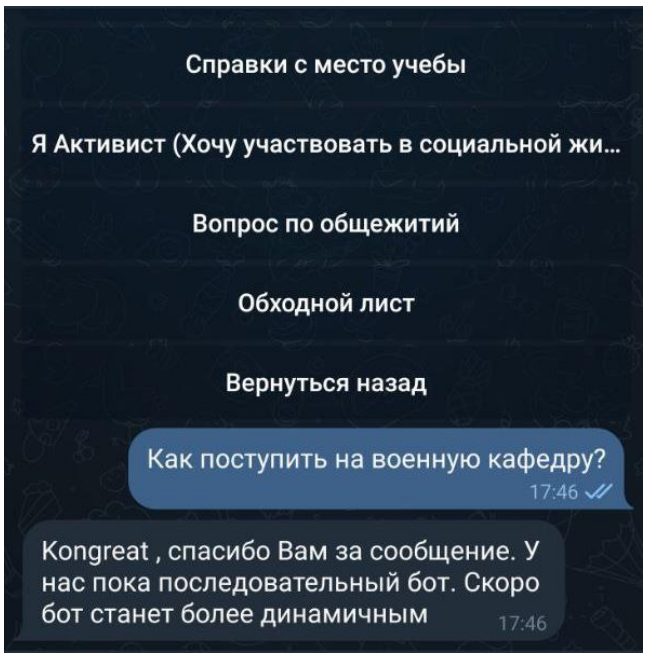

Рисунок-1.2 – Обработка ботом сообщений

При попытке отправить вопрос в свободной форме или же просто отправить сообщение, оно никак не обрабатывается и бот предупреждает, что он не может работать в таком режиме. То есть у него есть функционал только ответа на заранее определенные вопросы.

Оба бота предлагают достаточно обширный функционал по ответу на вопросы, при этом нужно будет затратить большое количество времени на поиск интересующей темы. Боты никак не адаптируются под пользователя и предоставляют только общую информацию. Пользователь не может узнать свои данные. Также, они не обрабатывают вопросы в свободной форме или же вопросы, заданные не текстом. Помимо этого, они никак не задействованы в процессе общения между эдвайзерами и студентами.

2. Технологический раздел

2.1 Telegram

Telegram [6] — система, доступная на разных устройствах и ОС, которая дает возможность мгновенно обмениваться сообщениями, файлами, различными медиа, также вступать в групповые чаты и каналы, при наличии подключения к интернету.

Мессенджер получил большую популярность и увеличил количество пользователей в последнее время, из-за своей скорости работы, безопасности, а также большому количеству индивидуальных настроек. У Telegram открытый исходный код и есть API для разработчиков. Он активно поддерживается разработчиками, имеет большое количество библиотек у востребованных языков программирования для взаимодействия с его API.

2.2 SQLite

SQLite – простая в использовании, компактная, при этом производительная и отвечающая современным требованиям СУБД, в системе выбран данный продукт в качестве основной СУБД. Является самой популярной СУБД в мире. Имеет открытый исходный код. SQLite является реляционной СУБД и использует язык запросов SQL, со своими особенностями и дополнительными возможностями. На данный момент поддерживается стандарт языка SQL-92. Выбор в пользу данной СУБД был сделан, так как ее легко развернуть, она не требует отдельного процесса при работе, нет необходимости настраивать или как-либо администрировать БД, имеет небольшой вес и не нагружает систему [7].

2.3 Язык программирования

2.3.1 Python

Язык программирования, который использовался для написания кода, это Python версии 3.10. Этот язык программирования находится на пике своей востребованности и популярности, используется довольно часто во многих задачах. У него много фанатов среди разработчиков, множество библиотек и фреймворков под любые нужды. В основном, данный язык используется в тех системах, в которых нужно МО или ИИ. Также для написания кода для проекта использовалось большое количество библиотек.

2.3.2 Фреймворки и библиотеки

Для разработки системы используется специальный асинхронный фреймворк для Python, который называется Aiogram. Он использует возможности Telegram API, позволяет обрабатывать запросы от разных пользователей в одно время. Этот фреймворк имеет хорошую и понятную документацию, большое количество обучающего материала и поддержку от пользователей.

Для работы с вводом произвольных вопросов я использовал библиотеку returmextract, которую нашел в списке доступных библиотек для языка Python. Она оказалась удобной для выполнения конкретных задач в системе.

Также для работы с аудио используются библиотеки ffmpeg [8] и speechrecognition [9], которые в свою очередь, в своей имплементации используют методы машинного обучения и искусственного интеллекта.

2.4 Редактор кода

В качестве редактора для написания кода был выбран Visual Studio Code, созданный компанией Microsoft. Основной выбор пал на него, так как предоставляет большой функционал для настроек, имеет дружественный и понятный пользовательский интерфейс, огромное количество дополнений и имеет высокую скорость работы. Visual Studio Code сочетает в себе простоту редактора кода, но при этом, при помощи дополнений с инструментами для разработчиков, заметно расширяет свой функционал. Данный редактор кода доступен на всех популярных ОС. Имеет поддержку большого количества языков программирования, имеет подсветку синтаксиса, настраиваемые сочетания клавиш, интерактивный отладчик, совместимость с системами контроля версий, что было весьма важно при создании системы, а также имеется поддержка завершения кода IntelliSense [10].

2.5 Система контроля версий

Для того, чтобы иметь историю всех версий разработки проекта, свободно перемещаться между ними в случае необходимости, также иметь резервную копию исходного кода, которая будет находиться не на локальном устройстве, была выбрана система версионного контроля GitHub.

Создан закрытый, приватный репозиторий, доступ к которому имеет только разработчик системы. Все изменения, обновления в программе вносились

туда с соответствующими комментариями, чтобы потом было легче ориентироваться в них.

Работа в данной системе контроля версий была построена так: изначально у разработчика есть главная ветка, в которой происходят и хранятся все изменения. По мере обновления функционала данная ветка обновляется. В случае, если функционал, который был добавлен, имел экспериментальное назначение – создавалась отдельная ветка (копия главной), которая при надобности и подтверждении нужности данного функционала, потом сливалась в одно целое с главной.

Таким образом, без рисков для системы, всегда была возможность откатиться на предыдущие версии и добавлять новые возможности.

3. Проектная часть

3.1 Общая архитектура проекта

Проект разработан на основе современных технологий, описанных ранее. Общая схема взаимодействия технологий, используемых в системе, с точки зрения конечного пользователя приведена на рисунке 3.1.

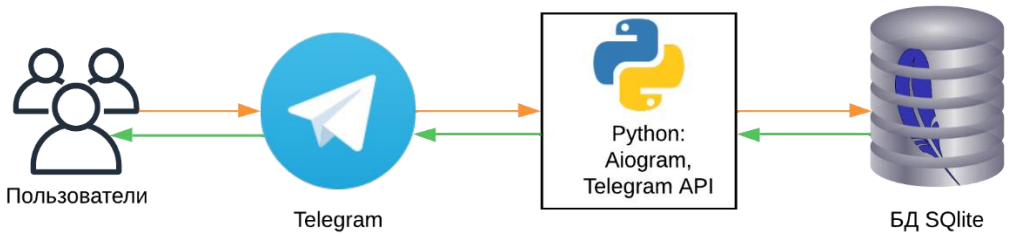

Рисунок-3.1 – Общая схема взаимодействия технологий между собой

Так как чат-бот написан для мессенджера Telegram, то пользователи взаимодействуют с ним при помощи данного мессенджера. Пользователи взаимодействуют с приложением или веб-версией Telegram, далее при помощи фреймворка Aiogram для языка программирования Python, запрос в асинхронном режиме обрабатывается в системе. После этого, идет обращение к базе данных, которая выдает необходимый пользователю результат, согласно данным пользователя и информация выводится на экран пользователя через мессенджер.

Система может идентифицировать пользователя, если его данные имеются в БД. В Telegram каждому пользователю присваивается уникальный идентификатор, он не изменяется по прошествию времени и закрепляется за пользователем. Во время запросов, он передается в систему. У всех пользователей в БД указан данный идентификатор и по нему можно найти более детальную информацию по студенту или эдвайзеру. Пользователи, чьего идентификатора не будет в БД не смогут воспользоваться системой.

3.2 Структура проекта на уровне кода

В языке программирования Python, код программы хранится в так называемых модулях и пакетах. Модули – файлы с расширением .py, в которых и находится непосредственно сам код. Пакеты [11] – это каталог, в котором модули объединяются логически. Используются для того, чтобы создавать пространства имен. Чтобы каталог считался пакетом, необходимо добавить в него файл «__init__.py» и импортировать нужные модули.

```
from keyboards import admin_kb
from database import sqlite_db
```

Рисунок-3.2 – Пример использования пакетов

В системе используется 3 основных пакета и 2 отдельных модуля.

Пакеты:

- database – содержит в себе логику и функции для взаимодействия с БД;
- handlers – содержит модули с обработчиками для студентов, эдвайзеров и общего чата;
- keyboards – клавиатуры, которые выводятся пользователю для взаимодействия с ботом.

Отдельные модули:

- bot.py – создание и запуск бота;
- create_bot.py – файл инициализации бота.

3.3 Принцип взаимодействия с серверами Telegram

Есть два основных принципа обмена данными между приложениями: long polling и webhook [12]. Обе технологии предназначены для получения данных о свежих событиях, которые происходят на сервере.

При использовании long polling – клиент опрашивает сервер, через определенные промежутки времени на предмет обновлений. Однако, это не значит, что клиент постоянно посылает запросы на сервер касательно новых событий. Приставка «long», как раз означает, что клиент посылает запрос, после этого, соединение остается открытым и сервер, спустя некоторое время, лишь когда новое событие действительно произойдет, отвечает ему. Данный метод прост в реализации не требует дополнительных настроек.

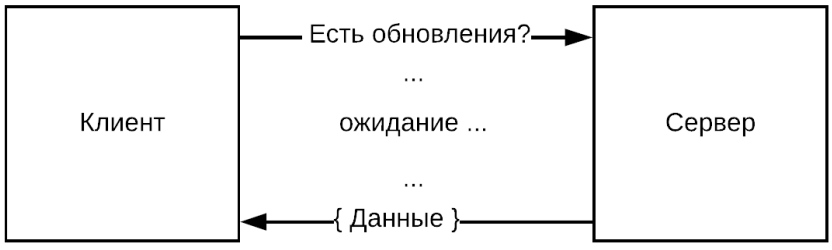

Рисунок-3.3 – Схема взаимодействия клиента и сервера long polling

Webhook работает по другому принципу. При использовании данного принципа, информация о событии отсылается сервером - клиенту, тогда, когда оно происходит на сервере. То есть у клиента нет постоянной необходимости

опрашивать сервер на предмет изменений. Сервер сам отправит данные, когда они появятся. Это помогает экономить количество запросов и трафика, также уменьшить нагрузку на сервер. Однако, для реализации данного метода, необходимы дополнительные настройки, создание промежуточного сервера, что значительно усложняет процесс разработки и поддержки.

Рисунок-3.4 – Схема взаимодействия клиента и сервера webhook

У данных принципов есть как свои преимущества, так и недостатки. Проанализировав потребности, а также особенности создаваемой системы, было принято решение использовать метод long polling. Он по умолчанию доступен из фреймворка и не требует дополнительных настроек или администрирования.

3.4 UML диаграммы

UML диаграммы [13] – это специальный вид диаграмм, широко применяемый в разработке программного обеспечения, который использует унифицированный язык моделирования. Благодаря ему описывается модель, которая в свою очередь описывает какие-либо действия или объект. Преимущество в том, что при помощи данных диаграмм можно описать проекты и системы совершенно разные с точки зрения масштаба и назначения, так как существует единый синтаксис и процесс построения. В основном используются для проектирования систем, реверс-инжиниринга, документации.

3.4.1 Диаграмма использования

Диаграмма использования или use-case диаграмма – это один из видов UML диаграмм, который описывает основные требования к системе. Он определяет варианты использования и поведение пользователей программного обеспечения. При помощи данной диаграммы можно определить то, как будет происходить действие, но она не дает точных инструкций для того, чтобы понять, как это будет реализовано. Основная особенность в том, что данная схема поможет лучше понять то, как будет выглядеть система с точки зрения

конечного пользователя. Данная диаграмма относительно простая, она не предоставляет слишком много информации, а концентрируется лишь на главных взаимодействиях между пользователями и системой. Также она не содержит информацию о порядке, в котором происходит взаимодействие между основными действующими лицами [14].

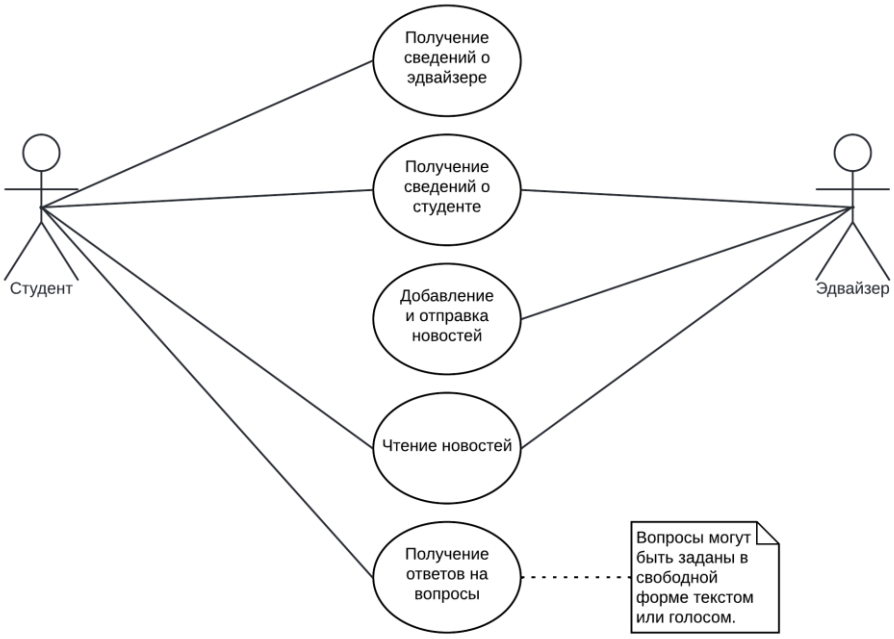

Рисунок-3.5 – Диаграмма использования

Функционал системы, описанный на рисунке 3.5 - является базовым и необходимым, так как основной целью является облегчение и автоматизация взаимодействия между студентами и эдвайзерами.

Получение сведений об эдвайзере – одна из главных функций системы, которая позволит студентам получать информацию и контакты эдвайзера, который курирует их.

Получение сведений о студенте – данная функция дает возможность студентам узнать персонализированную информацию о себе. Например, такую как успеваемость, задолженность. Эдвайзеры же в свою очередь могут получить информацию о своих студентах, их контакты, чтобы далее, при необходимости связаться.

Добавление и отправка новостей – позволяет эдвайзерам добавлять новости в базу. Также есть возможность выбрать и отослать в общую группу особенно важные новости, чтобы студенты обязательно с ними ознакомились.

Чтение новостей – данная функция предназначена в основном для студентов, но также и эдвайзеры могут ознакомиться со списком новостей. При этом, пользователям показываются только свежие новости, которые актуальны на данный момент времени.

Получение ответов на вопросы – также одна из главных функций системы. Позволяет студентам получать ответы на вопросы, чтобы не задействовать при этом эдвайзеров, тем самым снимая с них нагрузку и повышая скорость получения ответа. Стоит заметить, что вопросы могут быть заданы как при помощи специальной клавиатуры с заранее заготовленными фразами, так и при помощи открытых вопрос текстом или же через голосовые сообщения.

Главными актерами в данной системе являются: студент и эдвайзер. Оба из них являются активными.

3.4.2 Диаграмма ER

Этот вид UML диаграмм используется в основном при проектировании баз данных. Так как база данных является обязательной частью разрабатываемой системы, то необходимо описать ее структуру и логику при помощи данной диаграммы. Она показывает основные объекты, а также связи между ними.

Данная схема помогает определить:

- основные сущности в базе данных, отношения между ними;
- дает возможность предварительно оценить всю логику в целом;
- помогает преобразовать БД в реляционную;
- помогает лучше составить представление о том, какие данные должны храниться в базе [15].

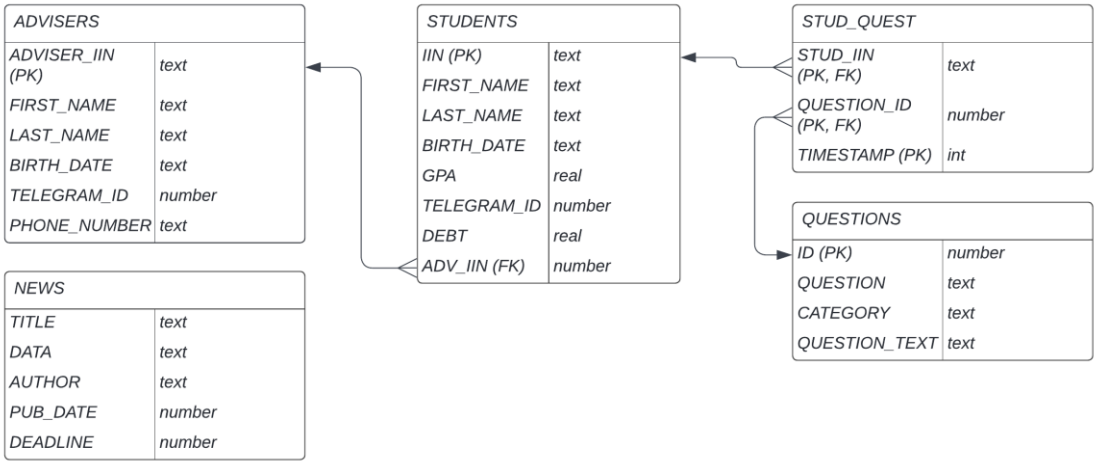

Рисунок-3.6 – Диаграмма ER

Для функционирования данной системы, минимальное необходимое количество объектов в базе данных – 5.

Таблица «Students» – является основной, в которой содержатся данные о студентах, а также идентификаторы эдвайзеров. Одним из главных полей считается «telegram_id». Данный идентификатор присваивается на стороне мессендежера и является уникальным для каждого пользователя. Первичным

ключом является поле с ИИН студента, который является также уникальным значением для каждого студента. Внешним ключом является поле с ИИН эдвайзера, которое ссылается на таблицу с информацией об эдвайзерах. Данная связь имеет вид один ко многим, так как у студента может быть только один эдвайзер, а у эдвайзера, в свою очередь, может быть много студентов.

Таблица «Advisers» - содержит информацию о каждом из эдвайзеров. Принцип работы с полем «telegram_id» такой же, как и в таблице со студентами. У каждого эдвайзера будет уникальное значение данного поля, а также ИИН. ИИН является первичным ключом.

Таблица «News» - содержит данные о новостях. Поля с автором и датой публикации проставляются автоматически в системе, при добавлении новостей. Важным полем является «Deadline». Данное поле отвечает за актуальность новостей. Актуальными считаются те новости, дедлайн которых еще не прошел.

В связи с особенностями выбранной БД, а именно – отсутствием типа данных для дат, пришлось записывать даты в числовом виде, в формате «годмесяцдень». Например, число «20220504» будет соответствовать 4 мая 2022 года.

Таблица «Questions» - содержит информацию о вопросах, а также ответы на вопросы, которые задают студенты. Вопросы разделены по категориям.

Таблица «Stud_Quest» является промежуточной таблицей между студентами и вопросами для осуществления связи многие ко многим. Также является неким аналогом лога вопросов. Туда заносятся данные о том, какие вопросы и в какое время задавали студенты, при обращении студента к боту. Имеет составной первичный ключ, для обеспечения уникальности. Внешними ключами являются ИИН студента и идентификационный номер вопроса.

3.5 Описание функционала

3.5.1 Токен бота и безопасность

Для того, чтобы система могла взаимодействовать с ботом, посылать запросы на сервера Телеграм, необходимо было получить специально сгенерированный, уникальный токен. Говоря простым языком, человек, который владеет токеном, имеет полный контроль над ботом. Чтобы это не произошло, и злоумышленники не получили токен, необходимо позаботиться о его безопасности. В этих целях, токен чат-бота локально и на хостинге был помещен в системные переменные среды сессии. Данные об этих переменных хранятся в ОС и получить доступ к ним сложнее. В том числе это сделано и для того, чтобы не хранить токен в общей доступности в исходном коде.

Также было создано специальное изолированное виртуальное окружение [16] для Python, в котором как раз переменная окружения и устанавливается равной токену бота.

На рисунке 3.6 показан локальный скрипт запуска чат-бота, в котором активируется ранее созданное виртуальное окружение, и в этом окружении устанавливается переменная при помощи команды «set».

```
@echo off

call %~dp0smart_adviser_bot\diploma_venv\Scripts\activate

cd %~dp0smart_adviser_bot

set BOT_TOKEN=

python bot.py

pause
```

Рисунок-3.7 – Скрипт запуска чат-бота

3.5.2 Аутентификация пользователя

Для того, чтобы пользователь смог начать взаимодействовать с ботом, ему необходимо написать боту в личные сообщения команду «/start». Далее, это действие можно будет делать и из общего чата, при этом сообщение удалится и бот ответит пользователю в личные сообщения.

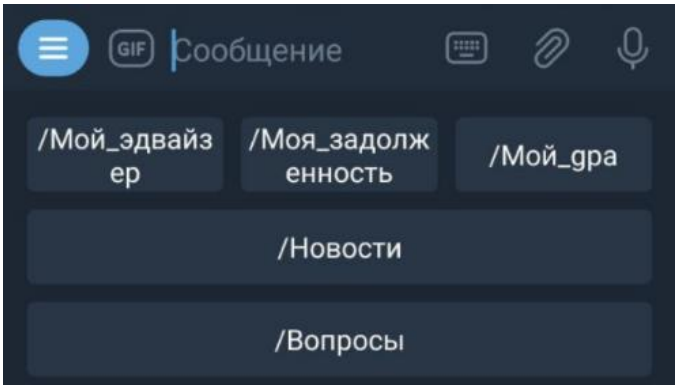

Рисунок-3.8 – Клавиатура для взаимодействия студента с чат-ботом

Здесь же происходит разделение пользователей на роли, чтобы войти как обычный пользователь, используется команда, написанная выше. Если же нужно попасть в панель администрирования, которая предназначена для эдвайзеров, то необходимо ввести команду «/admin» или «/adviser». При этом, доступ к данному меню имеют только пользователи со статусом администратора в общей группе с ботом. Команду для входа в панель администрации надо писать в общую группу.

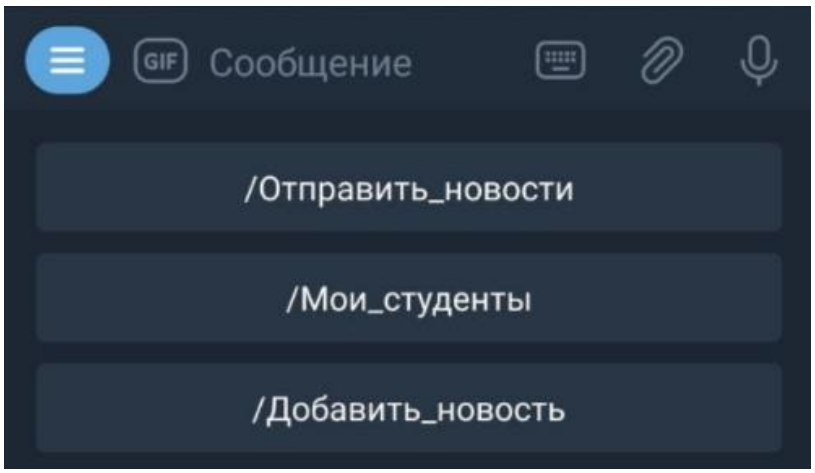

Рисунок-3.9 – Панель администрации

3.5.3 Фильтр нецензурной лексики

Для обеспечения корпоративной культуры делового общения, а также наведения общего порядка в чате данный бот также имеет функцию модерирования сообщений. Это значит, что при отправки любым пользователем группы нецензурных слов в чат, данное сообщение будет немедленно удалено и пользователь будет предупрежден, о недопущении подобных выражений.

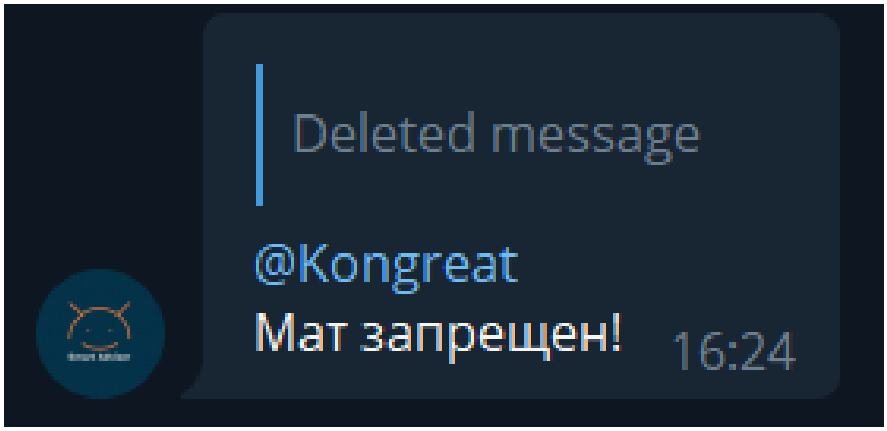

Рисунок-3.10 – Пример обработки сообщений, содержащих мат

Даная функция работает следующим образом: бот получает текст сообщения, далее приводит его к одному регистру и удаляет ненужные символы, токенизирует (разбивает на отдельные слова) текст, сравнивает слова с имеющейся локальной базой нецензурных слов. Если слово есть в том списке, то сообщение удаляется.

3.5.4 Получение ответов на вопросы

Данная функция в чат-боте работает через взаимодействие с БД. Студент вводит интересующий его вопрос, далее он обрабатывается и из БД поступает ответ на этот вопрос.

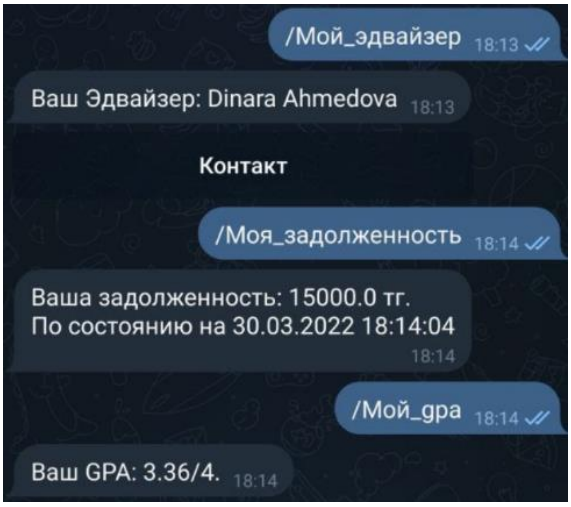

Рисунок-3.11 – Пример ответа чат-ботом на вопросы

В случае же, если пользователь задает персонализированные вопросы, например о своей успеваемости или эдвайзере, то программе передается уникальный telegram_id пользователя, который сверяется с БД и выбирается информация по данному идентификатору. Логика работает также и со стороны эдвайзера, при получении данных о своих студентах.

3.5.5 Новости

Данный функционал реализован также при помощи взаимодействия с базой данных. Студент, по команде, получает список актуальных новостей. Актуальными считаются новости, для которых текущая дата будет меньше, чем дата дедлайна, указанная в БД. При этом, из-за особенности используемой СУБД, а именно – отсутствия типа данных «DATE», дата записывается в числовом формате «yyyymmdd». Например, дата «5 апреля 2022 года» будет выглядеть как «20220405». Новости с ближайшим дедлайном будут выводиться студенту первыми. Студент увидит название новости, ее текст и дату публикации.

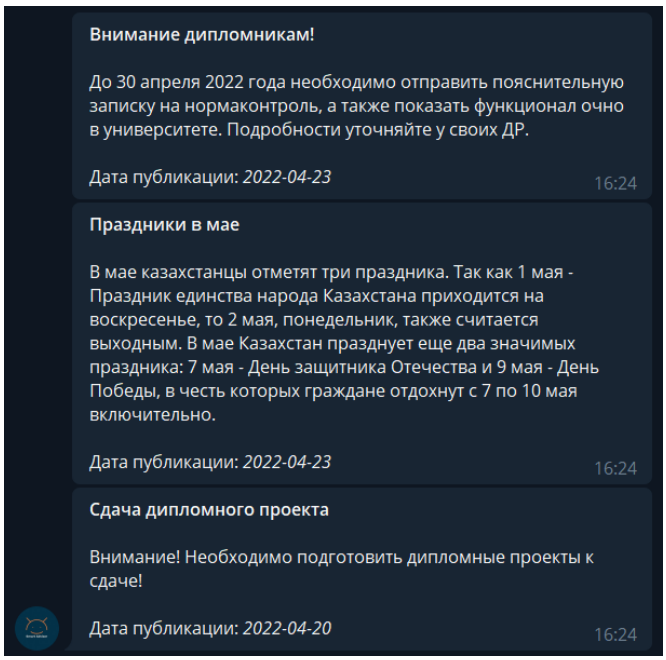

Рисунок-3.12 – Получение студентом новостей

При добавлении новости со стороны эдвайзера используется машина конечных состояний или же конечные автоматы. Данная возможность является встроенной в фреймворк Aiogram. FSM удобно использовать в данном случае, так как чат-бот будет ждать от пользователя ввода конкретной информации и после переходить в следующее состояние, тем самым двигаясь по очереди. При этом, есть возможность отменить добавление новости в любой момент.

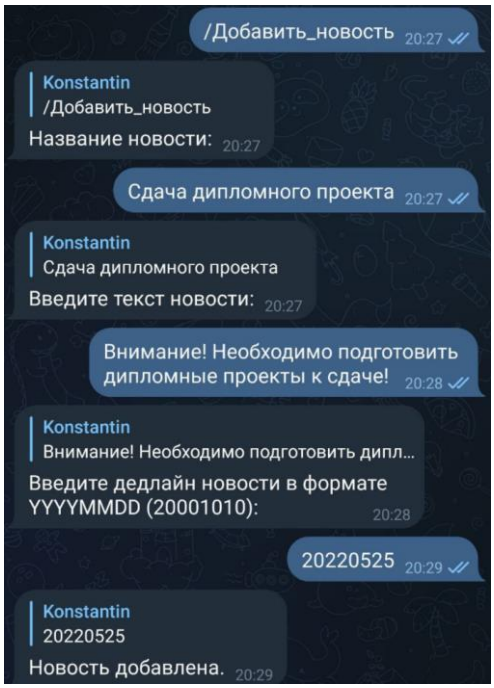

Рисунок-3.13 – Добавление новости эдвайзером

При необходимости, эдвайзер может отправить любую новость из базы данных в общую группу, путем нажатия кнопки в соответствующем меню.

3.5.6 Хостинг

Хостинг – это возможность арендовать сервер, для размещения своих файлов. Данный сервер работает постоянно и имеет подключение к сети. Это делается с целью, чтобы обеспечить бесперебойный доступ к системе 24 часа в сутки. При этом, файлы системы будут находиться на удаленных серверах.

Для этой цели был арендован виртуальный выделенный сервер за небольшую плату с достаточно скромными характеристиками, однако его хватит для демонстрации работы системы, а также небольшого количества пользователей. Его характеристики: 1 ядро процессора, 1 гигабайт оперативной памяти, 10 гигабайт твердотелого накопителя, и пропускная способность сети 250 мегабит в секунду.

После успешной оплаты, сервер стал доступен практически сразу. Туда были перенесены локальные файлы системы, выполнена первоначальная настройка, установка необходимого ПО, так как сервер работает на ОС Linux. Необходимо было внести мелкие правки в скрипты бота из-за особенностей ОС. Также токен бота помещен в переменные среды. Были установлены надежные пароли на подключение.

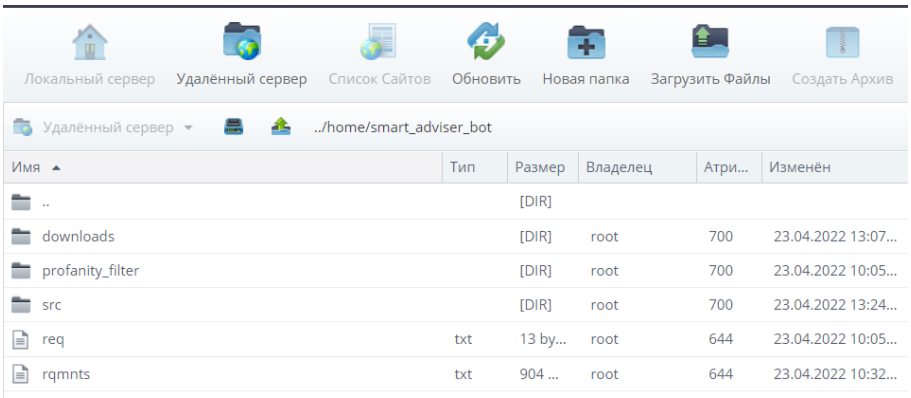

Рисунок-3.14 – Хранилище удаленного сервера

4 Экспериментальный раздел

4.1 Адаптация к студенту

Искусственный интеллект [17] – это способность системы выполнять задачи, которые ассоциируются с наличием интеллекта и имитируют разумное поведение человека в конкретной ситуации. Например, ему присущи такие функции как возможность рассуждать, извлекать смысл, учиться на основе предыдущих событий.

Одной из задач при проектировании системы было создать такого чат-бота, который адаптировался бы к конкретному пользователю и его запросам. Например, при входе в систему, чтобы студенту предлагались наиболее релевантные ответы на вопросы, чтобы чат-бот попытался предугадать, что будет интересно пользователю в данный момент. В системе это реализовано так, что при повторной авторизации пользователя в меню чат-бота, ему будут даны подсказки по вопросам, на основании тех, которые он задавал ранее. То есть система обучается на основе предыдущего опыта и накопленных данных.

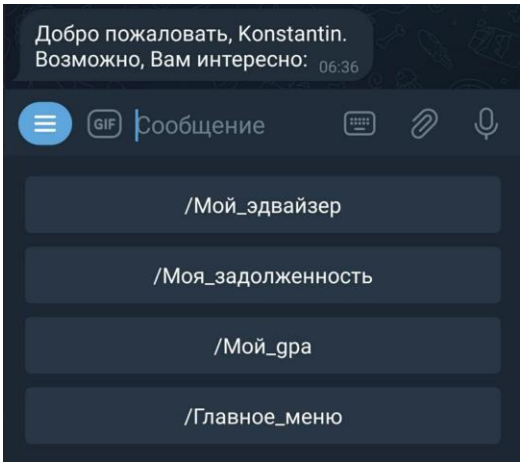

Рисунок-4.1 – Подсказки при повторном входе

Принцип работы, следующий: вопросы пользователя логируются в специальный объект в БД. У каждого вопроса есть категории. При повторном входе в меню чат-бота выполняется запрос к БД, в котором извлекается самая часто задаваемая категория вопросов по данному пользователю. Если же пользователь ранее не задавал вопросов, то ему выдается стандартная начальная клавиатура, общая для всех пользователей. Это получается реализовать, так как каждый пользователь идентифицирован в системе. Тем самым, чат-бот получается более гибким и удобным в использовании. Эту способность системы адаптироваться и учиться на основе прошлого опыта, можно назвать своего рода искусственным интеллектом.

stud_iin	question_id	cnt
test	4	4
test	3	3
test	5	3
test	1	2
test	6	2
test	7	2
test	2	1

Рисунок-4.2 – Результат запроса для определения часто задаваемых вопросов у студента

4.2 Обработка вопросов, заданных в свободной форме

4.2.1 Обработка текстовых сообщений

Занимаясь разработкой данной системы, была найдена специальная библиотека [18], для извлечения ключевых слов, которая хорошо работала с небольшими предложениями на русском языке. Для английских текстов существует большое количество моделей и библиотек, однако для русского языка, таких, чтобы хорошо справлялись с небольшими отрывками текста пока очень мало и их сложно найти в открытом доступе.

Принцип работы данной библиотеки построен на морфологическом анализе и частеречной разметке. Также в ней используются основные действия, которые также применяются в NLP. Общий принцип работы данной библиотеки показан на рисунке 4.3.

Рисунок-4.3 – Общий принцип работы библиотеки для извлечения ключевых слов

Морфологический анализ – это выяснение грамматических характеристик слов при помощи заранее заданного набора алгоритмов [19].

Частеречная разметка – разметка слов текста на основе их части речи и грамматических характеристик [20].

В системе данная библиотека используется для того, чтобы пользователь мог задавать вопросы в свободной форме. Общий принцип заключается в том, что студент задает вопрос в поле для отправки сообщения, отправляет сообщение боту с вопросительным знаком в конце, тогда бот понимает, как нужно обрабатывать данное сообщение. После этого, из вопроса пользователя извлекаются ключевые слова, которым находится соответствие с заранее определенными категориями и разметкой вопросов. Далее, выдаются подсказки на основе извлеченных слов.

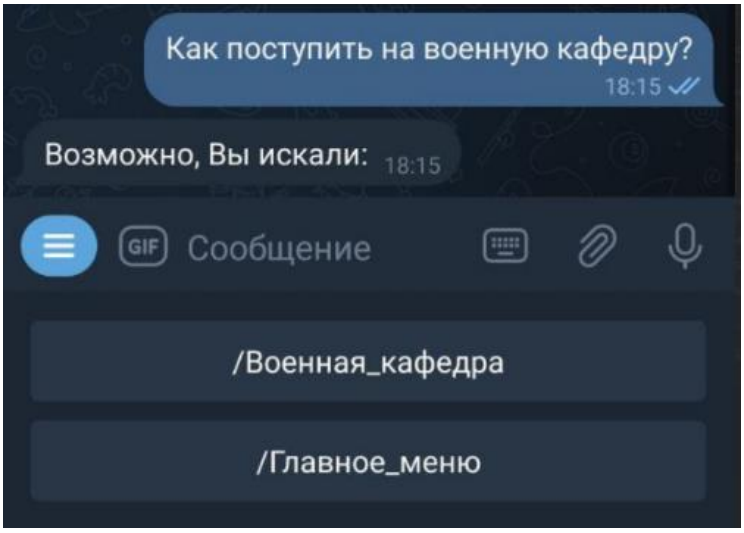

Рисунок-4.4 – Пример ответа на свободный вопрос

4.2.2 Обработка голосовых сообщений

Еще одной особенностью созданного чат-бота является возможность задавать вопросы при помощи голосовых сообщений, что может значительно экономить время и позволяет задавать вопросы находясь в пути или при невозможности использовать традиционные способы ввода информации. Для реализации данной функции используется библиотека speech_recognitnion, которая использует API от компании Google, и переводит голос в текст. Также используются ffmpeg для преобразования аудио файла в нужный формат. Общий принцип работы данной функции показан на рисунке 4.5.

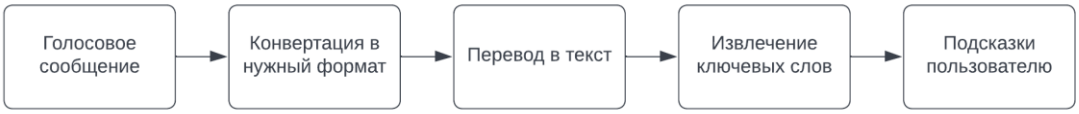

Рисунок-4.5 – Схема работы функции по ответу на голосовые вопросы

Преобразование речи в текст работает следующим образом [21]:

- звук по своей природе — это колебания, или же волны, которые имеют определенные характеристики. Данные волны улавливаются и переводятся в цифровой формат;
- звуки извлекаются из аудиофайлов, измеряются и фильтруются, потом выделяются нужные отрезки;
- звук сегментируется на мелкие части и сравнивается с фонемами. Фонемы – звуковые единицы, которые отличают слова друг от друга;
- эти фонемы пропускаются через нейросеть и на основе математической модели сравнивается с известными словами и фразами;
- далее все это преобразуется в текст и выдается пользователю.

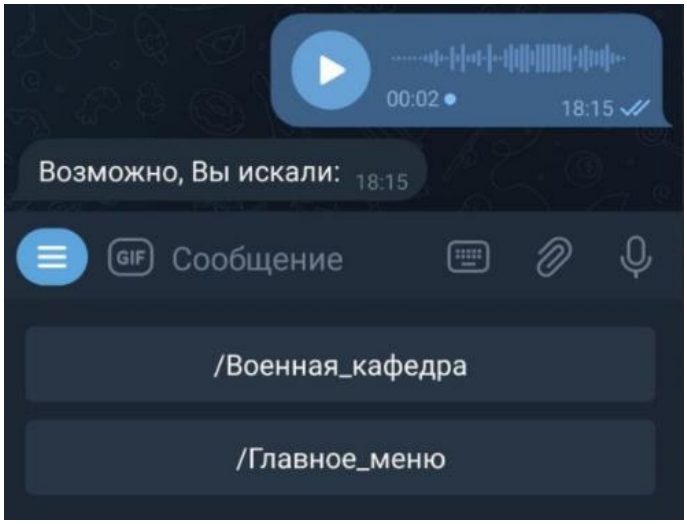

Рисунок-4.6 – Пример ответа на вопрос, заданный голосом

Таким образом, при реализации данной функции задействованы возможности ИИ и МО. При этом, если же алгоритмам не удалось перевести аудио файл в текст, то пользователь будет уведомлен об этом. Если же, система не смогла найти подходящие ответы на вопросы, студент также будет поставлен в известность.

ЗАКЛЮЧЕНИЕ

В результате выполнения данной дипломной работы получилось реализовать все поставленные задачи, использовать методы машинного обучения и искусственного интеллекта и успешно создать систему, основной целью которой является автоматизация взаимоотношений между студентами и их эдвайзерами.

В процессе разработки в систему постепенно добавлялись новые функции. Например, проект был помещен на хостинг. Также появились новые возможности по обработке сообщений от студентов в свободной форме текстовыми или голосовыми сообщениями.

Во время анализа будущей системы, а также имеющихся аналогов, было выяснено, что рынок чат-ботов активно развивается и компании стараются повсеместно использовать их для улучшения взаимодействия между представителями компаний и клиентами. При этом, имеющиеся аналоги либо являются платными и практически не работают с русским языком, либо обладают весьма ограниченным функционалом или же используются в других областях. Так как проблема, на которой сосредоточена данная система, является достаточно узкоспециализированной, то для ее решения сложно найти аналоги.

Созданная система является легко масштабируемой, доступной со всех современных ОС. При необходимости ее можно будет легко интегрировать в уже существующие системы, также дополнить функционал, если это потребуется.

На данный момент система ограничена данными, которые имеются в локальной БД, однако интеграция с базами данных университетов позволит значительно расширить функционал системы, добавить новые возможности и улучшить ее работу и полезность, облегчить учебный быт студентов и эдвайзеров, тем самым вынести процесс автоматизации взаимоотношений на новый уровень.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1 Global Chatbot Market Anticipated to Reach $9.4 Billion by 2024 – Robust Opportunities to Arise in Retail & eCommerce // Электронная версия на сайте https://markets.businessinsider.com/news/stocks/global-chatbot-market-anticipated-to-reach-9-4-billion-by-2024-robust-opportunities-to-arise-in-retail-ecommerce-1028759508.

2 Chatbots Market By Type (Stand-alone, Web-based), End User (Large, Small and Medium-sized Enterprises) – Growth, Sharem Opportunities & Competitive Analysis, 2021-2027 // Электронная версия на сайте https://www.credenceresearch.com/report/chatbots-market.

3 Humans + bots: Tension and opportunity // Электронная версия на сайте https://www.technologyreview.com/2018/11/14/239924/humans-bots-tension-and-opportunity/.

4 Chatbots 101 // Электронная версия на сайте https://www.oracle.com/us/technologies/mobile/chatbot-infographic-3672253.pdf.

5 80% of businesses want chatbots by 2020 // Электронная версия на сайте https://www.businessinsider.com/80-of-businesses-want-chatbots-by-2020-2016-12.

6 Telegram // Электронная версия на сайте https://ru.wikipedia.org/wiki/Telegram.

7 Особенности SQLite // Электронная версия на сайте https://unetway.com/tutorial/sqlite#:~:text=Особенности%20SQLite,в%20одном%20кросс-платформенном%20диске.

8 Документация ffmepeg // Электронная версия на сайте https://ffmpeg.org/ffmpeg.html.

9 Документация библиотеки speech_recognition // Электронная версия на сайте https://github.com/Uberi/speech_recognition#readme.

10 Документация Visual Studio Code IntelliSense // Электронная версия на сайте https://code.visualstudio.com/docs/editor/intellisense.

11 Python: Основы программирования // Электронная версия на сайте https://vk.cc/ccVYBa.

12 Long Polling vs. Webhooks // Электронная версия на сайте https://grammy.dev/guide/deployment-types.html#how-does-long-polling-work.

13 Марголин А. UML для бизнес-моделирования: зачем нужны диаграммы процессов // Электронная версия на сайте https://evergreens.com.ua/ru/articles/uml-diagrams.html.

14 What is Use Case Diagram? // Электронная версия на сайте https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-use-case-diagram/.

15 Peterson R. ER Diagram: Entity Relationship Diagram Model, DBMS Example // Электронная версия на сайте https://www.guru99.com/er-diagram-tutorial-dbms.html.

16 Что такое виртуальное окружение? Зачем оно? Какие бывают аналоги? // Электронная версия на сайте https://dvmn.org/encyclopedia/qna/12/chto-takoe-virtualnoe-okruzhenie-zachem-ono-kakie-byvajut-analogi/.

17 Copeland B.J. Artificial Intelligence // Электронная версия на сайте https://www.britannica.com/technology/artificial-intelligence.

18 Документация библиотеки rutermextract для Python // Электронная версия на сайте https://pypi.org/project/rutermextract/.

19 Козиев И. Морфологический анализатор и Part-Of-Speech Tagger // Электронная версия на сайте http://www.solarix.ru/for_developers/docs/morphology_analyzer.shtml.

20 Частеречная разметка // Электронная версия на сайте https://ru.wikipedia.org/wiki/Частеречная_разметка.

21 Что такое преобразование речи в текст? // Электронная версия на сайте https://aws.amazon.com/ru/what-is/speech-to-text/#:~:text=When%20sounds%20come%20out%20of,an%20analog%20to%20digital%20converter.

Приложение А
(обязательное)

Техническое задание

А.1.1 Техническое задание на разработку системы ChatBot для автоматизации взаимоотношений «эдвайзер-студент» в университете

Настоящее техническое задание распространяется на разработку системы ChatBot для автоматизации взаимоотношений «эдвайзер-студент» в университете, которая предназначена для использования в высшем учебном заведении для автоматизации и улучшения качества, а также облегчения взаимодействия между студентами и их кураторами. Предполагается, что данную систему будут использовать эдвайзеры и студенты. Автоматизация данных взаимоотношений позволит улучшить качество и скорость оказания услуг и также снять часть нагрузки с эдвайзеров путем ответов на часто задаваемые вопросы чат-ботом.

А.1.1 Основание для разработки

Система разрабатывается на основании устного распоряжения дипломного руководителя.

А.1.2 Назначение

Система предназначена для автоматизации и улучшения взаимодействия между эдвайзерами и студентами в университете, также для сбора, хранения и обработки информации.

А.1.3 Требования к функциональным характеристикам

При помощи языка Python, асинхронного фреймворка для него Aiogram, который использует возможности Telegram API, нужно создать чат бота для мессенджера Telegram.

Должна использоваться реляционная база данных для хранения информации о пользователях.

Бот используется как участник для общего чата с эдвайзером и студентами, оттуда пользователи начинают взаимодействие с ним.

У пользователей бота есть разные роли: эдвайзер (администратор, модератор) и студент (обычный пользователь). Эдвайзер имеет доступ к базе данных новостей, добавляет их, также может получить информацию о своих студентах. Студент может получать ответы на общие вопросы, персонализированные вопросы, читать новости при помощи взаимодействия с БД.

Система должна быть размещена на хостинге, который обеспечит бесперебойный доступ к ней в любое время суток.

Система должна обеспечивать выполнение данных функций

- ответы на часто задаваемые вопросы;
- разграничивать роли пользователей;
- добавление, сохранение и рассылка новостей;
- адаптация системы к конкретному пользователю на основании его предыдущих действий.

А.1.4 Требования к надежности

Обеспечить высокую скорость работы, разграничить возможности пользователей с разными ролями, обеспечить конфиденциальность и сохранность пользовательских данных, не предоставлять токен бота в открытом виде в исходном коде.

Приложение Б
(обязательное)

Текст программы

Полный исходный код системы доступен на GitHub по ссылке: https://github.com/Kongreat/smart_adviser_bot.

Ниже приведен код основных, особенно важных функций.

1. Sqlite_db.py – логика взаимодействия с базой данных:

```
# Функция для подключения к БД.
def sql_start():
    global db, cur
    db = sq.connect('smart_adviser.db')
    cur = db.cursor()
    if db:
        print('DB connected...')
        # Создание основных таблиц, если файл БД пустой.
    db.execute('CREATE TABLE IF NOT EXISTS students2\
                (iin TEXT PRIMARY KEY, \
                first name TEXT, \
                last_name TEXT, \
                birth_date TEXT, \
                gpa REAL, \
                telegram_id INT, \
                debt REAL, \
                phone_number TEXT, \
                adv_iin TEXT, \
                FOREIGN KEY (adv_iin) REFERENCES advisers2(adviser_iin))')

    db.execute('CREATE TABLE IF NOT EXISTS advisers2\
                (adviser_iin TEXT PRIMARY KEY, \
                first_name TEXT, \
                last_name TEXT, \
                birth_date TEXT, \
                telegram_id INTEGER, \
                phone_number TEXT)')

    db.execute('CREATE TABLE IF NOT EXISTS news2 \
                (id INTEGER PRIMARY KEY,\
                title TEXT,\
                data TEXT,\
                author TEXT,\
```

Продолжение приложения Б

```
                pub_date TEXT)')

            db.execute('CREATE TABLE IF NOT EXISTS questions \
                    (id INTEGER PRIMARY KEY AUTOINCREMENT,\
                     question TEXT,\
                     category TEXT, \
                     question_text BLOB)')

            db.execute('CREATE TABLE IF NOT EXISTS stud_quest \
                    (stud_iin TEXT, \
                    question_id INTEGER, \
                    timestamp INTEGER, \
                    PRIMARY KEY (stud_iin, question_id, timestamp), \
                    FOREIGN KEY (question_id) REFERENCES questions (id), \
                    FOREIGN KEY (stud_iin) REFERENCES students2 (iin) )')

            db.commit()

        # Получение данных об эдвайзере студента.
        async def get_adviser(message):
            return cur.execute(f'''select a.first_name, a.last_name, a.phone_number
                                from advisers2 a, students2 s
                                where s.telegram_id = {message.from_user.id}
                                and s.adv_iin = a.adviser_iin''').fetchall()

        # Получение данных о задолженности студента.
        async def get_debt(message):
            return cur.execute(f'SELECT debt from students2 \
                                where telegram_id = {message.from_user.id}').fetchall()

        # Получение данных о GPA студента.
        async def get_gpa(message):
            return cur.execute(f'SELECT gpa from students2 where telegram_id =
{message.from_user.id}').fetchall()

        # Получение списка студентов для эдвайзера.
        async def get_students(message):
            return cur.execute(f'''SELECT s.first_name, s.last_name, s.iin,
s.phone_number
                                from students2 s, advisers2 a
                                where a.telegram_id = {message.from_user.id}
                                and s.adv_iin = a.adviser_iin
```

Продолжение приложения Б

```
                    '''').fetchall()

        # Функция для отправки выбранных новостей в общий чат.
        async def send_news_to_chat(data, chat_id):
            news = cur.execute('SELECT title, data, pub_date, deadline FROM news2
WHERE title == ?',(data,)).fetchall()
            for n in news:
                # Указывается идентификатор чата с эдвайзером, куда отправляется
новость.
                await bot.send_message(chat_id, text = f'*{n[0]}*\n\n{n[1]}\n\nДата
публикации: _{n[2]}_', parse_mode='markdown')

        # Добавление новости в БД.
        async def add_news(state):
            async with state.proxy() as data:
                cur.execute('INSERT INTO news2 (title, data, author, pub_date, deadline)
VALUES (?, ?, ?, ?, ?)', tuple(data.values()))
                db.commit()

        # Получение списка актуальных новостей из БД.
        async def get_news(message):
            return cur.execute("SELECT title, \
                                data,\
                                pub_date, \
                                deadline \
                            FROM news2 \
                            WHERE CAST(strftime('%Y%m%d') as INTEGER)
BETWEEN pub_date AND deadline \
                        ORDER BY deadline").fetchall()

        # Запись данных о заданных вопросах.
        async def question_log(message):
            # Извлечение ИИН.
            data = cur.execute(f'SELECT iin FROM students2 WHERE telegram_id =
{message.from_user.id}').fetchall()
            iin = data[0][0]

            # Извлечение ID вопроса.
            data_2 = cur.execute(f'SELECT id FROM questions WHERE question =
"{message.text[1:]}"').fetchall()
            id = data_2[0][0]
```

```
            curr_dt = datetime.now()
            timestamp = int(round(curr_dt.timestamp()))

            cur.execute(f'INSERT INTO stud_quest VALUES ("{(iin)}", {id},
{timestamp})')
            db.commit()

        # Получение названия категории вопроса, который наиболее часто
задавался студентом.
        async def get_questions_cat(message):
            data_iin = cur.execute(f'select iin from students2 where telegram_id =
{message.from_user.id}').fetchall()
            iin = data_iin[0][0]

            data_id = cur.execute(f'select question_id, \
                                    count(stud_iin) as cnt \
                                  from stud_quest \
                                  where stud_iin = "{iin}" \
                                  group by question_id, stud_iin \
                                  order by cnt desc \
                                  limit 1').fetchall()
            if len(data_id) == 0:
                return None
            else:
                id = data_id[0][0]
                category_data = cur.execute(f'select category from questions where id =
{id}').fetchall()
                return category_data[0][0]

        # Получение ответов на вопросы по названию вопроса
        async def get_question(q):
            data = cur.execute(f'SELECT question_text FROM questions WHERE
question = "{q}"').fetchall()
            return data[0][0]
```

2. Admin.py – разделение ролей пользователей:

```
        ID = None # Глобальная переменная для хранения ID Администратора
(эдвайзера).
        GROUP_ID = None

        # Функция для входа в панель админа через общий чат.
```

Продолжение приложения Б

```
async def admin_panel(message: types.Message):
    global ID
    global GROUP_ID
    ID = message.from_user.id
    GROUP_ID = message.chat.id
    await bot.send_message(message.from_user.id, 'Добро пожаловать в
панель администрации.',\
        reply_markup = admin_kb.btns_admin_panel)
    await message.delete()
```

3. Shared.py – фильтр нецензурной лексики:

```
# Фильтр мата. Сканирует все сообщения.
async def delete_profane(message: types.Message):
    # Явно указан путь к файлу с базой запрещенных слов.
    if {i.lower().translate(str.maketrans("", string.punctuation)) for i in
message.text.split(' ')}\
    .intersection(set(json.load(open('C:/Users/konst/Desktop/diploma/profanity_
filter/profane_words.json')))) != set():
        await message.reply(f'@{message.from_user.username} \nМат
запрещен!')
        await message.delete()
```

4. Student.py – выделение ключевых слов, подсказки, перевод голосовых сообщений в текст:

```
# Выделение ключевого слова из вопроса.
async def extract_term(text):
    term_extractor = TermExtractor()
    for t in term_extractor(text, nested=True):
        if t.normalized in student_kb.suggestions.keys():
            return t.normalized
        #else: return 'NULL'

async def questions_custom(message:types.Message, speech=''):
    # Если функция вызвалась из чата.
    if speech == '':
        term = await extract_term(message.text)
        if term == None:
            await bot.send_message(message.from_user.id, 'Попробуйте
перефразировать.')
        else:
```

Продолжение приложения Б

```
                await bot.send_message(message.from_user.id,
                                       'Возможно, Вы искали:',
                                       reply_markup=await create_keyboard(term, 2))

            # Если функция вызвалась через распознавание речи.
            else:
                term = await extract_term(speech)
                if term == None:
                    await bot.send_message(message.from_user.id, 'Попробуйте
перефразировать.')
                else:
                    await bot.send_message(message.from_user.id,
                                           'Возможно, Вы искали:',
                                           reply_markup=await create_keyboard(term, 2))

        # Перевод голосового сообщения в текст.
        async def speech_to_text(message: types.Message):
            # Скачивание и конвертация файла.
            file_name = r'C:\Users\konst\Desktop\diploma\downloads\speech'
            file = await bot.get_file(message.voice.file_id)
            file_path = file.file_path
            await bot.download_file(file_path, f'{file_name}.ogg')

            # Преобразование аудио файла в текст.
            try:
                voice = AudioSegment.from_ogg(f'{file_name}.ogg')
                voice.export(f'{file_name}.wav', format='wav')
                r = sr.Recognizer()
                with sr.AudioFile(f'{file_name}.wav') as source:
                    audio_data = r.record(source)
                    text = r.recognize_google(audio_data, language='ru-RU')
                    # print(text)
                    # return text
                    await questions_custom(message, text)
            except sr.UnknownValueError:
                await bot.send_message(message.from_user.id, f'Не удалось распознать
голосовое сообщение.')
```

Протокол

о проверке на наличие неавторизованных заимствований (плагиата)

Автор: Бородкин Константин Евгеньевич

Соавтор (если имеется):

Тип работы: Дипломная работа

Название работы: Создание системы ChatBot для автоматизации взаимоотношений «эдвайзер-студент» в университете

Научный руководитель: Максатбек Сатымбеков

Коэффициент Подобия 1: 2.7

Коэффициент Подобия 2: 0.4

Микропробелы: 0

Знаки из здругих алфавитов: 1

Интервалы: 0

Белые Знаки: 5

После проверки Отчета Подобия было сделано следующее заключение:

☐ Заимствования, выявленные в работе, является законным и не является плагиатом. Уровень подобия не превышает допустимого предела. Таким образом работа независима и принимается.

☐ Заимствование не является плагиатом, но превышено пороговое значение уровня подобия. Таким образом работа возвращается на доработку.

☐ Выявлены заимствования и плагиат или преднамеренные текстовые искажения (манипуляции), как предполагаемые попытки укрытия плагиата, которые делают работу противоречащей требованиям приложения 5 приказа 595 МОН РК, закону об авторских и смежных правах РК, а также кодексу этики и процедурам. Таким образом работа не принимается.

☐ Обоснование:

Дата 19.05.2022

Заведующий кафедрой

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

Казахский национальный исследовательский технический университет имени К.И.Сатпаева

Институт автоматики и информационных технологий

Бородкин Константин Евгеньевич
5В070400 – Вычислительная техника и программное обеспечение

ОТЗЫВ НАУЧНОГО РУКОВОДИТЕЛЯ

к дипломному проекту

Тема: «Создание системы ChatBot для автоматизации взаимоотношений «эдвайзер-студент» в университете»

Представленный дипломный проект соответствует выданному заданию. Бородкин Константин проявил свои навыки, инициативность, при разработке данной системы. Все поставленные задачи были выполнены.

Структура дипломной работы включает в себя: введение, четыре раздела, заключение, список используемых источников литературы и приложений, а также 22 рисунка.

Во введении определяется актуальность выбранной темы, цели и задачи исследования, обосновывается цель разработки проекта.

В первой главе исследуется понятие ChatBot, проводится анализ по аналоговым системам, проведен анализ рынка и доказана актуальность системы.

Во второй главе представлен ыобзор по средам разработки и технологиям проектирование системы.

В третьей главе определяются функциональные требования к системе, приведены диаграммы базы данных, диаграмма использования, описывается разработка системы.

В четвертой главе описаны экспериментальные функции, которые были добавлены в систему.

В заключении приведены выводы о проделанной работе.

Пояснительная записка соответствует требованиям, предъявленным к данным видам работ. Считаю, что работа выполнена успешно и может быть оценена на «отлично», а при успешной защите Бородкин К. достоин присвоения степени бакалавра техники и технологий.

Научный руководитель:

Сениор-лектор, доктор Ph.D. _____ Сатымбеков М.

«18» 05 2022 г.

РЕЦЕНЗИЯ

На дипломный проект студента 4 курса Satbayev University, Института Автоматики и Информационных Технологий, кафедры Программной Инженерии, специальности «Вычислительная техника и программное обеспечение» Бородкина Константина Евгеньевича, выполненный на тему «Создание системы Chatbot для автоматизации взаимоотношений «эдвайзер-студент» в университете»

В работе исследованы современные проблемы, проанализированы существующие системы в сфере автоматизации отношений между компаниями и клиентами, проведены обзоры исследований на эту тему, доказана актуальность данной проблемы и обоснована востребованность поиска решения для автоматизации взаимоотношений между эдвайзерами и студентами в области высшего образования.

Корректно и точно определена цель системы, также в полном объеме выделен список задач, которые должна решать система для достижения своей главной цели.

Проведен анализ существующих технологий, которые наиболее удачно подходят для создания Chatbot. Достоверно описана информация касательно выбранных технологий, их способов взаимодействия между собой, грамотно обоснован выбор той или иной технологии. В решении задач использованы современные методы и программные средства. Все перечисленные задачи выполнены.

В результате написания дипломного проекта, представлен рабочий вариант чатбота, размещенный на хостинге, который успешно решает существующую проблему в области взаимодействия студентов и эдвайзеров.

В целом работа выглядит готовой и завершенной, рекомендуемая оценка «Отлично», студент Бородкин К.Е. достоин присвоения академической степени бакалавра.

Рецензент:

Казахский национальный женский педагогический университет, PhD, и. о. ассоциированного профессора Н.О. Мекебаев