Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Сибирский государственный университет телекоммуникаций и
информатики»
(СибГУТИ)

09.03.01 Информатика и вычислительная техника
Профиль: Программное обеспечение средств вычислительной техники и автоматизированных систем
(очная форма обучения)

ОТЧЕТ ПО ПРОИЗВОДСТВЕННОЙ ПРАКТИКЕ
на Кафедре вычислительных систем

Разработка Telegram-бота для перевода слов

Выполнил:
студент института ИВТ
гр. ИП-015 _____/Костыль В.Ю./
«27» мая 2023г. (подпись)

Проверил:
Руководитель от СибГУТИ _____/Приставка П.А./
«27» мая 2023г. (подпись)

Новосибирск 2023

План-график проведения производственной **практики**

Вид практики

Костыль Валерии Юрьевны

Фамилия Имя Отчество студента

института Информатика и вычислительная техника, 3 курса, гр. ИП-015

Направление: 09.03.01 – Информатика и вычислительная техника

Код – Наименование направления (специальности)

Профиль: Программное обеспечение средств вычислительной техники и автоматизированных систем

Место прохождения практики Кафедра вычислительных систем

Объем практики: **360/10** часов/ЗЕ

Вид практики ***производственная***

Тип практики ***Технологическая (проектно-технологическая) практика***

Срок практики с "30" января 2023 г.
по "27" мая 2023 г.

Содержание практики*:

Наименование видов деятельности	Дата (начало – окончание)
1. Общее ознакомление со структурным подразделением предприятия, вводный инструктаж по технике безопасности	30.01.2023–02.02.2023
2. Выдача задания на практику, деление студентов на группы (если необходимо), определение конкретной индивидуальной темы, формирование плана работ	03.02.2023–05.02.2023
3. Работа с библиотечными фондами структурного подразделения или предприятия, сбор и анализ материалов по теме практики	07.02.2023–12.02.2023
4. Выполнение работ в соответствии с составленным планом: – Определение технологий, с которыми будет вестись работа – Изучение информации об API Telegram, в частности библиотеки aiogram и других – Создание бота – Формирование сообщения и его отправка	14.02.2023 – 21.05.2023
5. Анализ полученных результатов и произведенной работы Составление отчета по практике, защита отчета	23.05.2023–27.05.2023

*В соответствии с программой практики

Руководитель от СибГУТИ _____/_____/

« » 2023г. (подпись)

ЗАДАНИЕ НА ПРАКТИКУ

Разработка Telegram-бота для перевода песен по названию.

Работу можно разбить на несколько этапов:

1) Определение технологий, с которыми будет вестись работа.
2) Изучение информации об API Telegram, в частности библиотеки aiogram и других.
3) Создание бота перевода песен.
4) Формирование сообщения и его отправка.

ВВЕДЕНИЕ

Бот — это небольшое приложение, которое самостоятельно выполняет заранее созданные задачи без участия пользователя. Telegram-бот умеет делать всё, что мог бы делать человек в чате: отвечать на вопросы, присылать ссылки на сайты или создавать мемы. Автоматически или по запросу он может отправлять:

- текстовые сообщения;
- картинки;
- видео;
- файлы.

Боты умеют:

● Выполнять действия, которые нельзя настроить на канале. Например, продавать товары и принимать оплату, общаться с пользователями, скрывать личные данные. Боты для Телеграма могут собирать потенциальных клиентов, подключая их к CRM, системе продажи билетов или платформе обмена сообщениями.

● Выполнять несколько разных команд одновременно.

Важная функция ботов — возможность запускать цепочку действий, постепенно запрашивая у пользователя новую информацию. Если отправить боту команду /start, на экране появятся кнопки.

● Интегрироваться с другими сервисами.

Например, можно настроить бота, чтобы пользователи могли выполнять быстрый поиск GIF-анимации во встроенном режиме. Если встроенные запросы будут включены, пользователи смогут ввести имя бота и свой запрос в поле ввода текста в любом чате и мгновенно получить нужную гифку.

ПОСТАНОВКА ЗАДАЧИ

Для реализации бота для перевода слов потребуется получить источник откуда будем брать перевод. Для этого использую сайт lyrsense.com. Для того, чтобы взять с сайта данные, нам потребуется подключить соответствующую библиотеку. Задача – разобраться с тем, как правильно использовать библиотеку, и как запрограммировать нашего бота, по требованию пользователя, эту информацию предоставить.

ВЫБОР ПРОГРАММНЫХ СРЕДСТВ

В качестве языка программирования был выбран Python версии (3.11). Он располагает всем необходимым инструментарием для написания данной программы.

Python — высокоуровневый язык программирования общего назначения с динамической строгой типизацией и автоматическим управлением памятью, ориентированный на повышение производительности разработчика, читаемости кода и его качества, а также на обеспечение переносимости написанных на нём программ. Язык является полностью объектно-ориентированным в том плане, что всё является объектами. Необычной особенностью языка является выделение блоков кода пробельными отступами. Синтаксис ядра языка минималистичен, за счёт чего на практике редко возникает необходимость обращаться к документации. Сам же язык известен как интерпретируемый и используется в том числе для написания скриптов. Недостатками языка являются зачастую более низкая скорость работы и более высокое потребление памяти написанных на нём программ по сравнению с аналогичным кодом, написанным на компилируемых языках, таких как C или C++.

Python является мультипарадигменным языком программирования,поддерживающим императивное, процедурное, структурное, объектно-ориентированное программирование, метапрограммирование и функциональное программирование. Задачи обобщённого программирования решаются за счёт динамической типизации. Аспектно-ориентированное программирование частично поддерживается через декораторы, более полноценная поддержка обеспечивается дополнительными фреймворками. Такие методики как контрактное и логическое программирование можно реализовать с помощью библиотек или расширений. Основные архитектурные черты — динамическая типизация, автоматическое управление памятью, полная интроспекция, механизм обработки исключений, поддержка многопоточных вычислений с глобальной блокировкой интерпретатора, высокоуровневые структуры данных. Поддерживается разбиение программ на модули, которые, в свою очередь, могут объединяться в пакеты.

Стандартная библиотека включает большой набор полезных переносимых функций, начиная с возможностей для работы с текстом и заканчивая средствами для написания сетевых приложений. Дополнительные возможности, такие как математическое моделирование, работа с оборудованием, написание веб-приложений или разработка игр, могут реализовываться посредством обширного количества сторонних библиотек, а также интеграцией библиотек, написанных на Си или C++, при этом и сам интерпретатор Python может интегрироваться в проекты, написанные на этих языках. Существует и специализированный репозиторий программного обеспечения, написанного на Python, — PyPI. Данный репозиторий предоставляет средства для простой установки пакетов в операционную систему и стал стандартом де-факто для Python. По состоянию на 2019 год в нём содержалось более 175 тысяч пакетов.

В качестве среды для программирования была выбрана IDE PyCharm от JetBrains, так как она предоставляет удобный инструментарий для работы с данным языком, такой как: удобная установка библиотек, подсветка синтаксиса, рекомендации по написанию кода и т.д.

PyCharm — это кроссплатформенная интегрированная среда разработки для языка программирования Python, разработанная компанией JetBrains на основе IntelliJ IDEA. Предоставляет пользователю комплекс средств для написания кода и визуальный отладчик.

Для создания бота использовалась библиотека aiogram

ВЫПОЛНЕНИЕ РАБОТЫ

Этап 1: Определение технологий, с которыми будет вестись работа

Для написания Telegram ботов существует удобный инструментарий на языке Python. Из библиотек для создания бота был выбран aiogram, так как он представляет из себя простое и понятное решение, полностью покрывающее наши запросы.

Этап 2: Изучение информации об API Telegram, в частности библиотеки aiogram и других.
Были прочитаны документация и статьи на тему библиотек aiogram, dispatcher, executor, types.

Этап 3: Создание бота.

Прежде чем начинать разработку, бота необходимо зарегистрировать и получить его уникальный id, являющийся одновременно и токеном. Для этого в Telegram существует специальный бот — @BotFather.
Через него задаются внешние параметры бота такие как имя, главная иконка и т.д.
Токен который мы получили, в дальнейшем будем использовать для подключения к боту через Python.

```
telegram_bot_token = '61072881:AG0pPj-NbGM-cdA0A4RjQYY3SUGxPgrvtM'
bot = Bot(token=telegram_bot_token)
dp = Dispatcher(bot)
```

Этап 4: Формирование сообщения и его отправка.

Ниже описаны функции для отправки сообщений на определенные команды, введенные пользователем.

```
 @dp.message_handler(commands="Start")
async def start(message: types.Message):
    await message.answer("Здравствуйте, введите песню, для поиска перевода!")
```

Следующий handler обрабатывает любое сообщение от пользователя (кроме приведенных выше), и запускает весь алгоритм перевода слов.

```
@dp.message_handler()
async def search(message: types.Message):
    query = message.text.strip()
    url = f"https://lyrsense.com/search?s={query}"
    response = requests.get(url)
    soup = BeautifulSoup(response.text, 'html.parser')
    find_url = soup.find(class_="myModerList").find('a')
    find_url = "https://lyrsense.com" + find_url.get('href')
    response = requests.get(find_url)

    soup = BeautifulSoup(response.text, 'html.parser')
    name_song = soup.find("h1", {"style": "padding-bottom:0;"}).text
    song_translate = soup.find("p", {"id": "ru_text"}).find_all('span')
    song_original = soup.find("p", {"id": "fr_text"}).find_all('span')
    res =name_song + '\n\n'
    for i in song_translate:
        res+=i.text + '\n'
    res += "\nОригинал песни:\n\n"
    for i in song_original:
        res+=i.text + '\n'
    await message.reply(res)
```

Подключение и создание базы данных.

```
conn = sqlite3.connect('search_history.db')
cursor = conn.cursor()


cursor.execute('''
    CREATE TABLE IF NOT EXISTS search_history (
        id INTEGER PRIMARY KEY AUTOINCREMENT,
        user_id INTEGER,
        query TEXT
    )
''')
conn.commit()
```

Команда help отправляет пользователю сообщение о доступных командах.

```
@dp.message_handler(commands='help')
async def clear_history(message: types.Message):
    await message.reply("Для поиска перевода введите название песни" + "\n Команда /history
покажет историю запросов."+ "\n Команда /clearhistory очистит историю запросов.")
```

Команда history отправляет пользователю сообщение с историей его запросов.

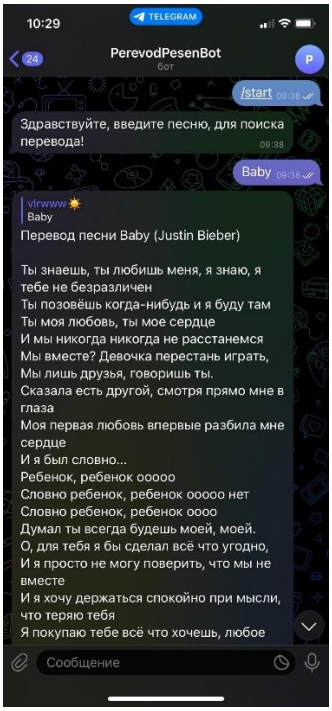
```
@dp.message_handler(commands="history")
async def history(message: types.Message):
    user_id = message.from_user.id
    cursor.execute('SELECT query FROM search_history WHERE user_id=?', (user_id,))
    search_history = cursor.fetchall()

    if search_history:
        history_text = "История поиска:\n\n"
        for query in search_history:
            history_text += f"- {query[0]}\n"
    else:
        history_text = "История поиска пуста."

    await message.reply(history_text)
```

Команда clearhistory удаляет историю запросов.

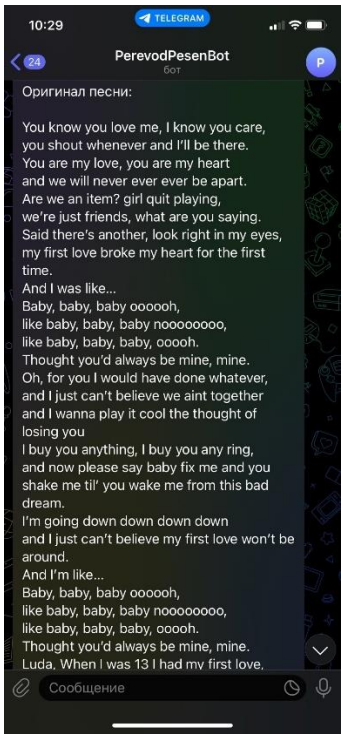
```
@dp.message_handler(commands="clearhistory")
async def clear_history(message: types.Message):
    user_id = message.from_user.id
    cursor.execute('DELETE FROM search_history WHERE user_id=?', (user_id,))
    conn.commit()
    await message.reply("История поиска очищена.")
```

ОПИСАНИЕ ИНТЕРФЕЙСА ПОЛЬЗОВАТЕЛЯ

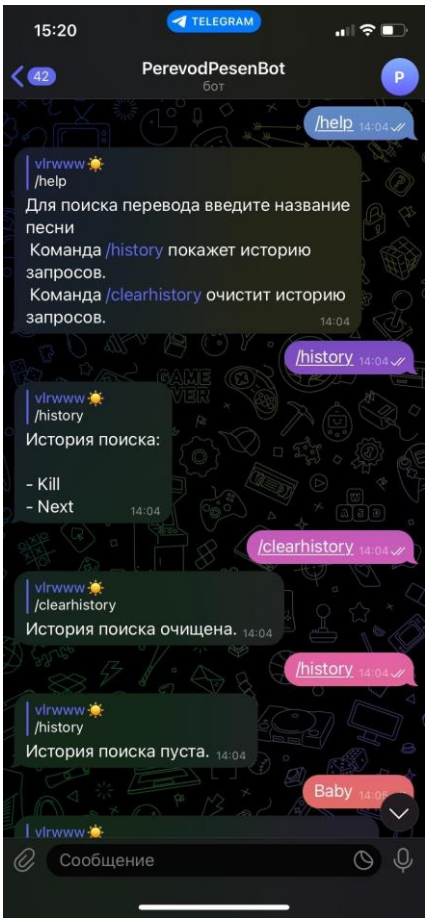
Для запуска работы телеграмм-бота нужно прописать команду /start, после чего ввести название нужной песни.



После чего бот выводит перевод песни на русский и оригинал на иностранном языке.



На скриншоте представлены команды и реакции бота на них.



ЗАКЛЮЧЕНИЕ

Был создан бот для быстрого перевода песен по её названию с использованием библиотеки aiogram. Библиотека Aiogram Python – это мощный инструмент для создания ботов для Telegram на языке Python. Она предоставляет обширный набор функций и возможностей, которые позволяют создавать ботов любой сложности.

Получила опыт работы с различными библиотеками для создания ботов и работы с ними на языке Python.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Ю.М. Шыырап, Л.Ю. Забелин, М.Ю. Шыырап Основы программирования Python: Учебное пособие / Сибирский государственный университет телекоммуникаций и информатики; каф. систем автоматизированного проектирования. – Новосибирск, 2017. – 197 с. №164 от 20.12.18.
2. Документация по pyTelegramBotAPI [Электронный ресурс]: PyTelegramBot.URL: https://pytba.readthedocs.io/ru/latest/index.html (Дата обращения 10.05.2023)

ПРИЛОЖЕНИЯ

```
from aiogram import Bot, Dispatcher, executor, types
import requests
from bs4 import BeautifulSoup
import sqlite3


telegram_bot_token = '6107281:G0Pj-NbGM-cdA0A4RjQYY3SUGxPgrvtM'
bot = Bot(token=telegram_bot_token)
dp = Dispatcher(bot)


conn = sqlite3.connect('search_history.db')
cursor = conn.cursor()


cursor.execute('''
    CREATE TABLE IF NOT EXISTS search_history (
        id INTEGER PRIMARY KEY AUTOINCREMENT,
        user_id INTEGER,
        query TEXT
    )
''')
conn.commit()

@dp.message_handler(commands="start")
async def start(message: types.Message):
    await message.answer("Здравствуйте, введите песню, для поиска перевода!")

@dp.message_handler(commands="history")
async def history(message: types.Message):
    user_id = message.from_user.id
    cursor.execute('SELECT query FROM search_history WHERE user_id=?', (user_id,))
    search_history = cursor.fetchall()

    if search_history:
        history_text = "История поиска:\n\n"
        for query in search_history:
            history_text += f"- {query[0]}\n"
    else:
        history_text = "История поиска пуста."

    await message.reply(history_text)

@dp.message_handler(commands="clearhistory")
async def clear_history(message: types.Message):
    user_id = message.from_user.id
    cursor.execute('DELETE FROM search_history WHERE user_id=?', (user_id,))
    conn.commit()
    await message.reply("История поиска очищена.")

@dp.message_handler(commands='help')
async def clear_history(message: types.Message):
    await message.reply("Для поиска перевода введите название песни" + "\n Команда /history покажет историю запросов."+ "\n Команда /clearhistory очистит историю запросов.")
```

```
@dp.message_handler()
async def search(message: types.Message):
    query = message.text.strip()
    url = f"https://lyrsense.com/search?s={query}"
    response = requests.get(url)
    soup = BeautifulSoup(response.text, 'html.parser')
    find_url = soup.find(class_="myModerList").find('a')
    find_url = "https://lyrsense.com" + find_url.get('href')
    response = requests.get(find_url)
    soup = BeautifulSoup(response.text, 'html.parser')
    name_song = soup.find("h1", {"style": "padding-bottom:0;"}).text
    song_translate = soup.find("p", {"id": "ru_text"}).find_all('span')
    song_original = soup.find("p", {"id": "fr_text"}).find_all('span')
    res = name_song + '\n\n'
    for i in song_translate:
        res += i.text + '\n'
    res += "\nОригинал песни:\n\n"
    for i in song_original:
        res += i.text + '\n'

    await message.reply(res)


    user_id = message.from_user.id
    cursor.execute('INSERT INTO search_history (user_id, query) VALUES (?, ?)', (user_id, query))
    conn.commit()

if __name__ == "__main__":
    executor.start_polling(dp)
```

Отзыв о работе студента

Костыль Валерия Юрьевна

(ФИО студента)

Уровень освоения компетенций

Костыль Валерия Юрьевна

(ФИО студента)

Компетенции	Уровень сформированности компетенций
ПК-1 - Способен разрабатывать требования и проектировать программное обеспечение	

отметка о зачете _____

Руководитель практики от СибГУТИ:

_____ _____ _____

Должность руководителя подпись ФИО руководителя

"___"_____20_____г