Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «ИНЖЕНЕРНЫЙ БИЗНЕС И МЕНЕДЖМЕНТ»

КАФЕДРА «ПРОМЫШЛЕННАЯ ЛОГИСТИКА» (ИБМ-3)

Лабораторные работы

«Парадигмы и конструкции языков программирования»

38.03.05 «Бизнес-информатика» (бакалавриат)

Студент ИБМ3-34Б _____ Сергеева М. А.
(Подпись, дата)

2024 г.

Лабораторная работа № 7

Тема: Разработка Telegram-бота с конечным автоматом из трех состояний

Цель работы

Разработать Telegram-бота с использованием конечного автомата (FSM) на языке программирования Python. Бот должен переходить между тремя состояниями в зависимости от ввода пользователя.

Ход выполнения работы

1. Подготовка среды

- Установлен Python (версия 3.8+).
- Установлены необходимые библиотеки:
  pip install aiogram
- Получен токен Telegram-бота через @BotFather.

2. Разработка бота

- Создан Python-скрипт и подключена библиотека aiogram.
- Определена модель конечного автомата с тремя состояниями:
  1. **Ожидание ввода (waiting_for_input)** – бот ждет, когда пользователь введет текст.
  2. **Обработка данных (processing_data)** – бот выполняет обработку введенного текста.
  3. **Возврат в начальное состояние** – бот завершает работу и предлагает ввести новое сообщение.
- Реализованы переходы между состояниями с помощью FSMContext.
- Добавлены обработчики команд /start и ввода текста.
- Реализована проверка, чтобы бот не ломался при отправке стикеров, фото и других типов сообщений.

3. Тестирование бота

- Проверены все состояния и переходы.
- Улучшен ответ бота для обработки неожиданных сообщений.

Вывод

В ходе работы был разработан Telegram-бот с конечным автоматом, который корректно обрабатывает пользовательский ввод. Реализованы три состояния, обеспечены плавные переходы между ними. Код соответствует требованиям задания и успешно протестирован.

Приложение

Код бота прилагается к отчету.

```
# ============================================================
#         TELEGRAM BOT ver 2
# ============================================================

# token = ""

from aiogram import Bot, Dispatcher, types, F
from aiogram.types import Message
from aiogram.fsm.context import FSMContext
from aiogram.fsm.state import State, StatesGroup
from aiogram.fsm.storage.memory import MemoryStorage
from aiogram.filters import Command
import asyncio
import logging

TOKEN = "" # здесь должен быть токен

# Настройка логирования
logging.basicConfig(level=logging.INFO)

# Создаем бота и диспетчер
bot = Bot(token=TOKEN)
storage = MemoryStorage()
dp = Dispatcher(storage=storage)


# Определяем конечный автомат с тремя состояниями
class Form(StatesGroup):
    waiting_for_input = State()  # Ожидание ввода
    processing_data = State()  # Обработка данных


# Обработчик команды /start
@dp.message(Command("start"))
```

```
async def start_command(message: Message, state: FSMContext):
    await state.set_state(Form.waiting_for_input)
    await message.answer("Привет! Введи любое слово, и я обработаю его!")


# Ожидание ввода от пользователя
@dp.message(Form.waiting_for_input)
async def get_user_input(message: Message, state: FSMContext):
    user_text = message.text
    await state.update_data(user_text=user_text)
    await state.set_state(Form.processing_data)
    await message.answer("Спасибо! Я обрабатываю твои данные...")

    # Переход в следующее состояние
    await process_data(message, state)


# Обработка данных и возврат в начальное состояние
async def process_data(message: Message, state: FSMContext):
    data = await state.get_data()
    user_text = data.get("user_text", "")
    response = f"Ты ввел: {user_text.upper()}"

    await message.answer(response)
    await state.clear()  # Завершаем состояние
    await message.answer("Можешь ввести новое слово или снова написать /start.")


# Реакция на ввод непонятного сообщения
@dp.message()
async def unknown_message(message: Message):
    await message.answer("Я не понимаю тебя. Напиши /start, чтобы начать.")

# Реакция на картинку, стикер или голосовое сообщение
@dp.message(Form.waiting_for_input, F.text)
async def get_user_input(message: Message, state: FSMContext):
    user_text = message.text
    if not user_text:  # Проверяем, что это именно текст
        await message.answer("Отправь мне текстовое сообщение.")
        return

    await state.update_data(user_text=user_text)
    await state.set_state(Form.processing_data)
    await message.answer("Спасибо! Я обрабатываю твои данные...")

    await process_data(message, state)


# Функция запуска бота
async def main():
    await dp.start_polling(bot)


if __name__ == "__main__":
    asyncio.run(main())
```