**Московский государственный технический**

**университет им. Н.Э. Баумана**

Факультет «Информатика и системы управления»

Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Программирование на основе классов и шаблонов»

Отчет по лабораторной работе №6

«Телеграм бот»

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Выполнил: | |  | | Проверил: | |
|  | студент группы ИУ5-31Б | |  | | преподаватель каф. ИУ5 | |
|  | Искорнев Иван | |  | |  | |
|  | Подпись и дата: | |  | | Подпись и дата: | |
|  | |  | |  | |  |

Москва, 2023 г.

**Задание:**

Разработайте бота для Telegram. Бот должен реализовывать конечный автомат из трех состояний

**Текст программы:**

lab\_06.py

from typing import Final  
from telegram import Update, InlineKeyboardButton, InlineKeyboardMarkup, InputFile  
from telegram.ext import Application, CommandHandler, MessageHandler, filters, ContextTypes, ConversationHandler  
import three\_steps  
  
TOKEN: Final = '6626051720:AAEkF94oDKNyfiCWxDHXzZtJkTbXvxJ1FsE'  
BOT\_USERNAME: Final ='@doperner\_bot'  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 app = Application.builder().token(TOKEN).build()  
  
 conv\_handler = ConversationHandler(  
 entry\_points=[CommandHandler('start', three\_steps.start\_command),  
 # CommandHandler('help', help\_command),  
 # CommandHandler('custom', custom\_command),  
 # CommandHandler('sendbuttons', send\_buttons)  
 ],  
 states={  
 three\_steps.FIRST\_NAME: [MessageHandler(filters.TEXT & ~filters.COMMAND, three\_steps.set\_first\_name)],  
 three\_steps.LAST\_NAME: [MessageHandler(filters.TEXT & ~filters.COMMAND, three\_steps.set\_last\_name)],  
 three\_steps.AGE: [MessageHandler(filters.TEXT & ~filters.COMMAND, three\_steps.set\_age)],  
 },  
 fallbacks=[],  
 )  
  
 app.add\_handler(conv\_handler)  
  
 # Start polling  
 print('Polling...')  
 app.run\_polling(poll\_interval=3)

three\_steps.py

import asyncio  
  
from typing import Final  
from telegram import Update  
from telegram.ext import Application, CommandHandler, ConversationHandler, MessageHandler, filters, ContextTypes  
  
FIRST\_NAME, LAST\_NAME, AGE = range(3)  
  
# Dictionary to store user information  
user\_info = {}  
  
async def start\_command(update: Update, context: ContextTypes.DEFAULT\_TYPE):  
 user\_id = update.message.from\_user.id  
 if user\_id not in user\_info:  
 user\_info[user\_id] = {}  
  
 await update.message.reply\_text('Привет! Введи свое имя')  
  
  
 return FIRST\_NAME  
  
async def set\_first\_name(update: Update, context: ContextTypes.DEFAULT\_TYPE):  
 user\_id = update.message.from\_user.id  
 user\_info[user\_id]['first\_name'] = update.message.text  
  
 await update.message.reply\_text("Теперь введи свою фамилию")  
  
 return LAST\_NAME  
  
async def set\_last\_name(update: Update, context: ContextTypes.DEFAULT\_TYPE):  
 user\_id = update.message.from\_user.id  
 user\_info[user\_id]['last\_name'] = update.message.text  
  
 await update.message.reply\_text(f"Отлично, {user\_info[user\_id]['first\_name']} {user\_info[user\_id]['last\_name']}! "  
 "Теперь введи свой возраст")  
  
 # Set the state to AGE  
 return AGE  
  
async def set\_age(update: Update, context: ContextTypes.DEFAULT\_TYPE):  
 user\_id = update.message.from\_user.id  
 try:  
 user\_info[user\_id]['age'] = int(update.message.text)  
 await update.message.reply\_text('Вот информация о тебе:\n'  
 f"Имя: {user\_info[user\_id]['first\_name']}\n"  
 f"Фамилия: {user\_info[user\_id]['last\_name']}\n"  
 f"Возраст: {user\_info[user\_id]['age']}")  
 except ValueError:  
 await update.message.reply\_text('Вы неправильно ввели возраст. Начните заново, написав /start.')  
  
 # End the conversation  
 return ConversationHandler.END

**Пример выполнения программы:**

