** 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 «Информатика и системы управления» Кафедра «Системы обработки информации и управления»**

Лабораторная работа №7

**«Разработка бота на основе конечного автомата для Telegram с использованием языка Python»**

по предмету

«Базовые компоненты интернет-технологий»

Выполнил:

студент группы № ИУ5-31Б

Михалёв Ярослав

Проверил:

Преподаватель кафедры ИУ-5

Гапанюк Юрий

2022 г.

Задание

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

Текст программы

**config.py**

BOT\_TOKEN = ""  
ADMIN\_ID = ""

**loader.py**

*import* logging  
  
*from* aiogram *import* Bot, Dispatcher, types  
*from* aiogram.contrib.fsm\_storage.memory *import* MemoryStorage  
  
*from* config *import* BOT\_TOKEN  
  
  
bot = Bot(token=BOT\_TOKEN, parse\_mode=types.ParseMode.HTML)  
storage = MemoryStorage()  
dp = Dispatcher(bot, storage=storage)  
  
logging.basicConfig(format=u'%(filename)s [LINE:%(lineno)d] #%(levelname)-8s [%(asctime)s] %(message)s', level=logging.INFO)

**app.py**

*from* aiogram *import* executor  
*from* handlers *import* dp  
  
*from* loader *import* bot, storage  
  
*from* utils.notify\_admins *import* on\_startup\_notify  
  
  
*async def* on\_startup(*dp*):  
 *await* on\_startup\_notify(*dp*)  
  
*async def* on\_shutdown(dp):  
 *await* bot.close()  
 *await* storage.close()  
  
  
*if* \_\_name\_\_ == '\_\_main\_\_':  
 executor.start\_polling(dp, on\_startup=on\_startup\_notify, on\_shutdown=on\_shutdown)

**handlers/operations.py**

*import* aiogram.utils.markdown *as* md  
*from* aiogram *import* types  
*from* aiogram.dispatcher *import* FSMContext  
*from* aiogram.dispatcher.filters *import* Command  
*from* aiogram.types *import* Message, ContentType  
  
*from* loader *import* dp  
  
*from* states.form *import* Form  
  
  
@dp.message\_handler(Command("start"))  
*async def* show\_menu(message: Message):  
 *await* Form.name.set()  
  
 *await* message.answer(text="Привет! Давай знакомится, как тебя зовут?")  
  
  
@dp.message\_handler(state=Form.name)  
*async def* process\_deposit\_money(message: Message, state: FSMContext):  
 *async with* state.proxy() *as* data:  
 data['name'] = message.text  
  
 *await* Form.next()  
  
 *await* message.answer("Сколько тебе лет?")  
  
  
@dp.message\_handler(state=Form.age)  
*async def* on\_derive\_money\_pressed(message: Message, state: FSMContext):  
 *async with* state.proxy() *as* data:  
 data['age'] = message.text  
  
 *await* Form.next()  
  
 markup = types.ReplyKeyboardMarkup(resize\_keyboard=*True*, selective=*True*)  
 markup.add("Я парень", "Я девушка")  
  
 *await* message.reply("Теперь определимся с полом", reply\_markup=markup)  
  
  
@dp.message\_handler(state=Form.gender)  
*async def* process\_gender(message: types.Message, state: FSMContext):  
 *async with* state.proxy() *as* data:  
 data['gender'] = message.text  
  
 types.ReplyKeyboardRemove()  
  
 *await* Form.next()  
  
 markup = types.ReplyKeyboardMarkup(resize\_keyboard=*True*, selective=*True*)  
 markup.add("Парни", "Девушки")  
  
 *await* message.answer("Кто тебе интересен?", reply\_markup=markup)  
  
  
@dp.message\_handler(state=Form.interestGender)  
*async def* process\_gender(message: types.Message, state: FSMContext):  
 *async with* state.proxy() *as* data:  
 data['interestGender'] = message.text  
  
 markup = types.ReplyKeyboardRemove()  
  
 *await* Form.next()  
  
 *await* message.answer("Из какого ты города?", reply\_markup=markup)  
  
  
@dp.message\_handler(state=Form.city)  
*async def* process\_gender(message: types.Message, state: FSMContext):  
 *async with* state.proxy() *as* data:  
 data['city'] = message.text  
  
 *await* Form.next()  
  
 *await* message.answer("Расскажите о себе")  
  
  
@dp.message\_handler(state=Form.description)  
*async def* process\_gender(*message*: types.Message, *state*: FSMContext):  
 *async with state*.proxy() *as* data:  
 data['description'] = *message*.text  
  
 *await* Form.next()  
  
 *await message*.answer("Теперь пришли своё фото")  
  
  
@dp.message\_handler(content\_types=ContentType.PHOTO, state=Form.photo)  
*async def* send\_photo\_file\_id(*message*: types.Message, *state*: FSMContext):  
 *async with state*.proxy() *as* data:  
 data['photo\_id'] = *message*.photo[-1].file\_id  
  
 *await message*.answer\_photo(  
 photo=data['photo\_id'],  
 caption=md.text(  
 md.text('Так выглядит твоя анкета:'),  
 md.text(""),  
 md.text(data['name'] + ", " + data['age'] + ", " + data['city']),  
 md.text(data['description']),  
 sep='\n',  
 )  
 )  
  
 *await state*.finish()

**states/form.py**

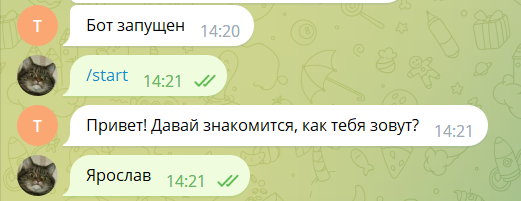
*from* aiogram.dispatcher.filters.state *import* State, StatesGroup  
  
*class* Form(StatesGroup):  
 name = State()  
 age = State()  
 gender = State()  
 interestGender = State()  
 city = State()  
 description = State()  
 photo = State()

**utils/notify\_admins.py**

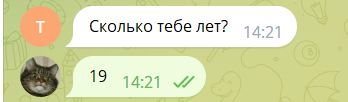
*import* logging  
  
*from* aiogram *import* Dispatcher  
  
*from* config *import* ADMIN\_ID  
  
  
*async def* on\_startup\_notify(*dp*: Dispatcher):  
 logging.info("Бот запущен")  
 *await dp*.bot.send\_message(chat\_id=ADMIN\_ID, text="Бот запущен")

Возможности бота

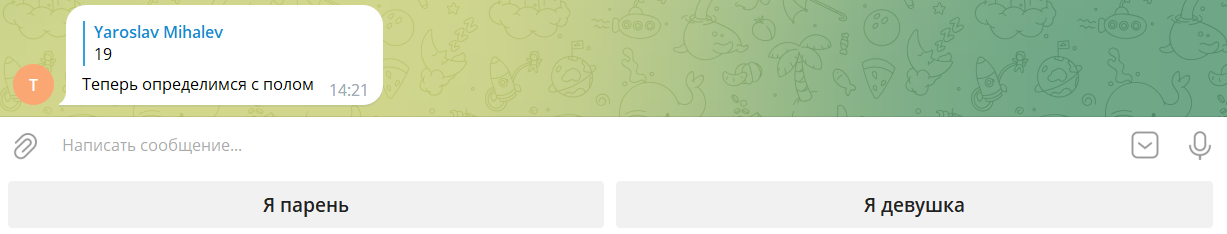
Запрос имени



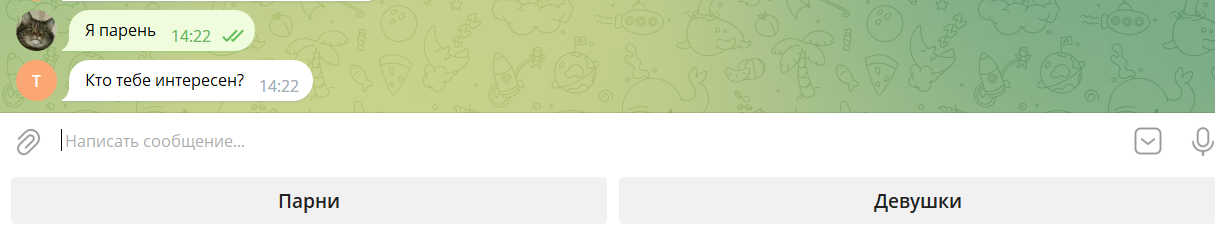
Запрос возраста



Запрос пола



Запрос пола партнера



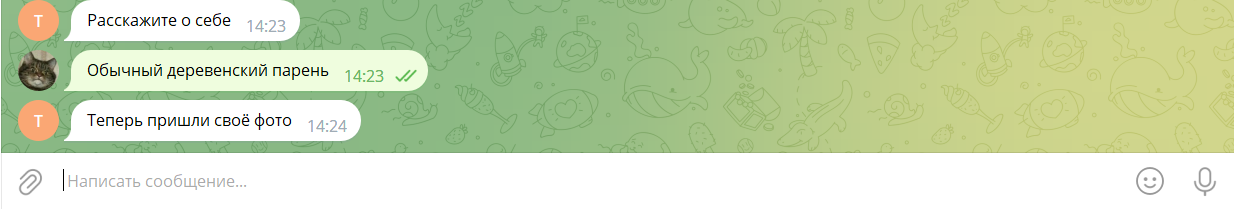
Запрос города



Запрос информации о себе



Запрос фото



После получения всей необходимой информации бот высылает сообщение с заполненной анкетой пользователя

